

EVENTUAL CONSISTENCY: ORIGIN AND SUPPORT

Francesc D. MUÑOZ-ESCOÍ, José-Ramón GARCÍA-ESCRIVÁ
Juan Salvador SENDRA-ROIG, José M. BERNABÉU-AUBÁN

Instituto Universitario Mixto Tecnológico de Informática
Universitat Politècnica de València
46022 Valencia, Spain
e-mail: {fmunyo, rgarcia, jsendra, josep}@iti.upv.es

José Ramón GONZÁLEZ DE MENDÍVIL

Departamento de Ingeniería Matemática e Informática
Universidad Pública de Navarra
31006 Pamplona, Spain
e-mail: mendivil@unavarra.es

Abstract. Eventual consistency is demanded nowadays in geo-replicated services that need to be highly scalable and available. According to the CAP constraints, when network partitions may arise, a distributed service should choose between being strongly consistent or being highly available. Since scalable services should be available, a relaxed consistency (while the network is partitioned) is the preferred choice. Eventual consistency is not a common data-centric consistency model, but only a state convergence condition to be added to a relaxed consistency model. There are still several aspects of eventual consistency that have not been analysed in depth in previous works: 1. which are the oldest replication proposals providing eventual consistency, 2. which replica consistency models provide the best basis for building eventually consistent services, 3. which mechanisms should be considered for implementing an eventually consistent service, and 4. which are the best combinations of those mechanisms for achieving different concrete goals. This paper provides some notes on these important topics.

Keywords: Eventual consistency, consistency model, CAP theorem, data replication

Mathematics Subject Classification 2010: 68-03, 68M14, 68N01, 68U35, 68W15

1 INTRODUCTION

Eventual consistency [64] has received a lot of attention in the last decade due to the emergence of elastic distributed services. Elastic services [31] need to be both scalable and adaptive, ensuring good levels of functionality, performance and responsiveness (i.e., QoS) – combined with a low cost – to their users, and of economical profit to their providers. In order to reach those levels of performance and responsiveness when the incoming workload being supported is high, the consistency among server replicas might be relaxed and this explains why eventual consistency has become so popular.

Elastic services are commonly deployed onto multiple datacentres and they may use thousands of computers. In those environments network partitions may arise. According to the CAP theorem, that was first stated by Fox and Brewer (1999) [26] and later proven by Gilbert and Lynch [28], when a network partition happens, there is a trade-off between strong consistency and service availability. Since elastic services must guarantee availability in order to comply with their QoS requirements, consistency needs to be relaxed in those situations. This is another reason for the success of eventually consistent services. However, as we will see in Section 2, the compromises stated in the CAP theorem were already known 40 years ago.

There have been many recent research works about eventual consistency [56, 64, 60, 8, 13, 54], but some aspects of this concept have not yet been discussed in depth. Therefore, in order to provide the missing pieces for building a complete picture on this subject, our paper is focused on those other aspects. They will be thoroughly analysed in the following sections that are introduced hereafter.

The first point of interest is a historical review. Although recent research works cite a paper from Werner Vogels [64] as the most well-known reference explaining this kind of consistency, there are many papers older than [64] which have either explained this same concept or implemented this consistency. Indeed, the oldest eventually consistent systems were proposed in the 70s of the past century. Those proposals tried to solve some problems that are close to those being solved nowadays: how to improve the performance and availability of scalable services, providing an acceptable level of consistency among their server instances. Some of these systems and solutions are revised in Section 2.

Section 3 sets the border between models that are inherently convergent and those others that are too much relaxed to be convergent *per se*. Inherently convergent replication models cannot be globally maintained when the network is partitioned. Eventual consistency requires that replica convergence is achieved when no new updates are received for a sufficiently long interval. This implies that *relaxed* models (e.g., FIFO/PRAM [45] or causal [1]) must be taken as a basis to develop eventually consistent services.

Section 4 explains how to implement an eventually consistent replicated service. Four complementary aspects are considered:

1. replication protocol,
2. operation ordering,
3. synchrony in agent interaction, and
4. state merging strategy for reaching convergence.

There are multiple alternatives in each aspect and, not surprisingly, some of the best combinations regarding performance and convergence were already known long time ago. However, depending on the concrete goals of an eventually consistent service, other new approaches may be needed and Section 4.2 revises those requirements.

The principles that are needed to manage eventually consistent services had been already used in several of the oldest distributed services. The challenges at that time were centred in providing acceptable response times and consistency using very limited computers and networks. Current hardware resources are much more powerful, but current services have also much more demanding requirements; e.g., they should be immediately adaptive [50]. So, although those old principles may guide our research efforts in this kind of dynamic consistency, other new contributions are still required in this area.

2 HISTORICAL REVIEW

Quoting Vogels [64], eventual consistency is “. . . a specific form of weak consistency; the storage system guarantees that if no new updates are made to the object, eventually all accesses will return the last updated value. If no failures occur, the maximum size of the inconsistency window can be determined based on factors such as communication delays, the load on the system, and the number of replicas involved in the replication scheme.”

That definition is informal, but clear and concise. Indeed, eventual consistency cannot be defined in a formal way as a regular consistency condition, since it is a liveness condition (eventual state convergence) that may be added to other consistency models. Besides defining it, Vogels also mentions a widely known service that implements eventual consistency: the domain name system [47, 48], proposed in 1983. Because of this, the reader may realise that traditional scalable distributed services have usually been eventually consistent. Vogels provides another pointer to a former publication on this subject: Lindsay et al. (1979) [44]. Therefore, it seems that eventual consistency has been a *classical* mechanism for achieving high performance in the distributed systems arena.

Taking a look at Section 1.4 from [44], the reader may observe that it describes a distributed relational database system that may use different replication protocols (primary-backup or majority voting) where two kinds of consistency may be managed. In the regular case, providing one-copy equivalence, transaction updates are forwarded and applied to all database replicas before that transaction is ended. On the other hand, with relaxed consistency (i.e., with eventual consistency), those updates may be forwarded afterwards, in a lazy way. This reduces the degree of syn-

chronisation being demanded by transactions, allowing their fast completion. Thus, this is one of the first examples of consistency-performance trade-off.

Assuming that the usage of relaxed consistency has been a common solution for improving the scalability of distributed services, it will be difficult to find the oldest research work that provided the first example of service or replication protocol specifically intended for ensuring that kind of consistency. There had been many old systems using replicated components and not all of them described their replication approaches in detail. Any way, let us go on in this backward look for that possible first eventually consistent service, analysing the challenges that were solved by each one of those proposals.

To this end, we may start considering the first reference on primary-backup replication [3]. In that paper, strong consistency is assumed. Both read and write requests are served by the primary replica. Write operations, once processed, forward their updates to the first backup replica. When this backup applies those updates, it sends the reply to the client process plus an acknowledgement to the primary and an update forwarding message to the next backup replica. When this algorithm is followed, the consistency being perceived is strong (indeed, linearisable [32]). In spite of this, Alsberg and Day also outline some variations of their basic algorithm. Thus, they also explain what can be done in multi-master scenarios. The general rule is to reach a consensus on the requests service order among all those master replicas before processing their incoming requests. But that general rule admits an exception that is described in this way in [3]: *“There may be specific applications where the nature of the service permits the out of order processing of requests. An example is an inventory system where only increments and decrements to data fields are permitted and where instantaneous consistency of the data base is not a requirement.”*

It assumes that some applications may provide an updating interface consisting of multiple commutative operations (e.g., increments and decrements in this example). In that case, multiple master replicas are allowed, serving their requests concurrently. Consistency is eventually achieved when every replica receives and applies all the updates generated (in any order) in the remaining replicas. This is a valid sample of an eventually consistent service and it was described in 1976, three years before the relaxed algorithm found in [44].

Moreover, Alsberg and Day [3] cite two related papers to look for additional information [12, 34]. Bunch [12] describes a preliminary version of the primary-backup algorithm discussed in [3]. In that version, backup replicas do not need to be linearly ordered and they do not propagate the updates following a chain forwarding approach. Instead, updates are logically multicast to all backups, allowing any kind of multicast implementation. Besides this first difference, there is another one: there are two classes of read operations referred to as critical and non-critical. Critical reads are directly managed by the primary replica. Non-critical read requests are forwarded to the *cheapest* replica (e.g., the one minimising transmission delay). In spite of their name, both read classes are strongly consistent since write operations do not return control to their client until all existing copies have been updated and have acknowledged the update completion. However, these non-critical reads

settled the basis for the relaxed consistency algorithm described in [44], once the synchronous update propagation was replaced with a lazy forwarding.

On the other hand, Johnson and Thomas [34] propose a replication algorithm that is more general than that of [3]. It allows multi-master replication for a given kind of database (a key-value store that maintains users data in a user authentication and accounting system [16]). Each database copy is held by a *database management process* (DBMP). The updates applied in a given master replica should be transferred to the remaining replicas. A list of pending replicas is maintained and the algorithm tolerates lazy propagation. Since multiple writers may exist and they all may concurrently apply conflicting updates in different replicas, some rules were needed to reach a convergent state once those updates were forwarded to the remaining replicas. To this end, Johnson and Thomas designed a solution based on update timestamping. In order to define that timestamping approach some local clock is used in every server, combined with node identifiers to break ties, defining a total order on all system events. The authors assume that those clocks could be sufficiently synchronised by default; otherwise, they suggest the usage of event counters in every node. Indeed, this was a solution that inspired the definition of logical clocks [41], as Lamport acknowledges at the end of his paper. This conflict resolution rule (i.e., “the last writer wins”) was also applicable in case of network partitions. Indeed, Johnson and Thomas mention the following regarding service continuity in case of network partitions: “. . . a completely general system must deal with the possibility of communication failures which cause the network to become partitioned into two or more sub-networks. Any solution which relies on locking an element of the database for synchronized modification must cope with the possibility of processes in non-communicating sub-networks attempting to lock the same element. Either they both must be allowed to do so (which violates the lock discipline), or they both must wait till the partition ceases (which may take arbitrarily long), or some form of centralized or hierarchical control must be used, with a resulting dependency of some DBMPs on others for all modifications and perhaps accesses as well.” Thus, they already identified in 1975 that in case of network partitions there is a trade-off between service availability and service consistency (since locking was assumed in that paper as a means for ensuring strong consistency); i.e., part of what is known nowadays as the CAP theorem [26, 28].

In our humble opinion, the paper from Johnson and Thomas can be considered the first key reference about eventual consistency. It was able to describe an efficient way to implement that kind of consistency (combining lazy update propagation with a general rule to reach convergence in case of conflicting updates, tolerating disconnected operation). Besides their data convergence rule, Johnson and Thomas defined specific mechanisms for detecting and managing delete-update and delete-create conflicts that might be hard to manage in a distributed deployment.

Let us now come back to our days, following a chronological order, to find additional contributions from other relevant papers in this historical review.

A first example is the LOCUS [65] distributed file system described by Parker et al. [52] in 1981. Its designers took care of handling network partitions, allowing

progress in disconnected subgroups of nodes. They also mentioned the trade-off between strong consistency and service availability when network partitions arise. In LOCUS, consistency was relaxed while disconnected nodes went on and *version vectors* were proposed in order to detect state conflicts, applying *reconciliation protocols* at reconnection time. Those reconciliation protocols depend on the semantics of the operations being applied. Note that the maintenance of version vectors at each replica implies that the updates being propagated comply with causal consistency.

A second example is the Grapevine system developed at Xerox by Birrell et al. [9]. Grapevine was an electronic mail service that also provided support for resource location, authentication and access control. The communication mechanisms being managed by the Grapevine servers were asynchronous (i.e., the sender was able to continue once the message was sent, without waiting for any kind of acknowledgement) and persistent (i.e., the communication servers were able to maintain the messages until their intended receivers were ready to get them). Grapevine needed a registration database where it maintained data about its users and its groups. A *group* maintained a collection of users addresses, thus allowing that a single e-mail message could be delivered to a set of users specifying their group name. In the Grapevine deployment (1981) described in [9], this system was spread through the Xerox sites at USA, Canada and United Kingdom. There were five registration (and message) servers and around 1500 users defining 500 user groups. The registration database was fully replicated in those registration servers using a multi-master approach. Database updates were forwarded in a lazy way through the asynchronous and persistent communication channels regularly used for electronic mail propagation. The replication algorithms tolerated network partitions, merging any conflicting updates using timestamps and the “last writer wins” principle already described in [34].

Fischer and Michael [25] describe an evolution of the algorithms presented in [34] for managing a distributed directory service. In this new solution, no explicit update operation is provided. Instead, the programmer should apply first a delete operation followed by a new insert. Additionally, the system remembers which objects have been inserted and which others have been deleted. With those sets, it is able to find out whether a given object is still active or not. A criterion for purging removed elements from both sets is also given. It is based on how many servers have already known that information. With all these variations on the Johnson and Thomas algorithm, the result is much simpler (indeed, no delete-update nor delete-create conflicts may arise) and it is still able to tolerate network partitions and unreliable communication.

Davidson (1984) [17] provides some rules for allowing service continuity in case of network partitions in a replicated database system. This means that the consistency among replicas is lost while the network remains partitioned, but service availability is guaranteed. However, once the partitioned groups rejoin, Davidson proposes several criteria for detecting serialisability violations and for choosing the set of transactions to be rolled back in order to build a global history that respects all serialisability requirements.

Apers and Wiederhold (1985) [4] also study the network partition problem in replicated database systems. However, instead of focusing only on serialisable order, they also consider semantic pre- and post-conditions on each kind of transaction. As a result, transactions are classified as:

1. unconditionally committable (UC), when their execution cannot violate any pre-condition of other transactions run in other partitions,
2. conditionally committable (CC), when their acceptance cannot be guaranteed but their possible afterwards rejection will not introduce other consistency problems, and
3. non-committable (NC), when their possible afterwards rejection leads to consistency problems that will not be solvable.

Only UC and CC transactions are accepted in case of a network partition. NC transactions are immediately rejected in that case. Algorithms are presented for merging partitions and for rebuilding their serialisation graphs, applying compensating transactions onto previously accepted CC transactions when needed. This is one of the first examples, when a system is partitioned, of managing semantic correctness criteria at partition reconnection time and of using conditional criteria for accepting some classes of transactions.

Other semantic criteria for managing state merging at partition reconnection time were proposed by Sarin et al. (1985) [57]. They base their solution in time-stamping all operations that modify the application or database state, defining in this way a total order for all those operations. However, no detail is given on how such update propagation should be made nor on how those timestamps are globally generated, allowing multiple kinds of implementations, even lazy propagation. When partitions rejoin, they forward and receive any missed updates. Conceptually, when a missed update is received in this reconnection stage, it leads to the roll back of all the operations that had been previously accepted and applied with a higher timestamp, reapplying them later on, in their appropriate order. However, multiple semantic optimisations are described for avoiding both the operation rollback and its reexecution once the missed update has been applied.

Demers et al. (1987) describe the Clearinghouse system in [21]. In that system “the effect of every update is eventually reflected in all replicas”. The system consists of several thousand nodes where a multi-master replication strategy is used. Each update request may be received and processed by a different replica and its effects will be lazily propagated to the remaining sites. The paper proposes and compares different lazy update propagation mechanisms in order to minimise network traffic: direct mail, anti-entropy and rumour mongering. With the latter two approaches the resulting system becomes highly scalable.

The first *computer-supported cooperative work* (CSCW) applications were developed in the middle eighties. The Lotus/Iris Notes project (Kawell et al. [35]) was based on lazy update propagation and on the “last writer wins” policy for dealing with concurrent updates. The database being managed was unconven-

tional: it was a collection of documents and a “transaction” consists in an update to one of those documents (multiple documents cannot be updated using a single action). On each update, a complete document should be transferred among replicas. Fortunately, documents were small (usually 1 or 2 KB). Those updates were transferred following a pull policy. Notes assumed that computers are not continuously connected to the network. When a computer contacts others, they exchange their documents lists with the versions and IDs for each document. When those lists were compared, the computer that missed any update requested the other to transfer those updates. Following this strategy, all Notes replicas became eventually consistent, but those replicas might had been inconsistent for quite long intervals.

Kumar and Stonebraker (1988) [37] describe how to apply the *escrow* [51] method to replicated databases, assigning complementary parts of the escrow to each replica; i.e., distributing the escrow. The original escrow mechanism allowed the management of concurrent transactions that use commutative operations to update a given relation field even when the value of such field should respect some constraints (e.g., to be positive or exceed some minimal threshold). Escrow distribution allows the management of some concurrent transactions without exchanging messages among replicas in some cases. This enhances performance and increases the tolerable degree of concurrency. As a result of this, inter-replica consistency is relaxed and transactions serialisability is lost, but transaction correctness is still preserved.

In the *lazy replication* (1990) approach [39] proposed by Ladin et al., client requests are forwarded to a single replica that processes the operation and later propagates its updates in a lazy way. The operations being processed may be ordered according to the application semantics, selecting one of these approaches: client ordering, server ordering or global ordering. In the client-order case, the client specifies which previously initiated operations precede the operation to be sent. To this end, each update operation returns an update identifier (uid) when it is completed and both queries and updates may specify as their arguments a set of precedent uids. In the server-order case, every server-ordered operation is totally ordered by the servers against every other server-ordered operation. Finally, in the global-order case, each global-ordered operation must be totally ordered by the servers against every other operation, independently on the type of the latter. This third type may be used in case of system reconfigurations, and it defines a border for ensuring that all server replicas must have delivered the same set of previous requests. Indeed, this is placing a convergence point for eventually consistent replicated services.

The systems to be implemented using the *lazy replication* technique find several advantages when they are compared to previous works. To begin with, they are basing their eventual consistency on an explicit (instead of potential, as when a regular causal multicast mechanism is being used for propagating updates) causal consistency. This reduces the amount of dependencies to be considered among the updates being propagated, enhancing performance and reducing delivery delays. Those de-

lays may arise, e.g., when a precedent causal message is lost and is resent. A second advantage is the careful management of application semantics for specifying how concurrent operations should be observed by every replica.

The Coda distributed file system [58] (1990) is an example of distributed environment allowing disconnected operation. In this case, consistency is relaxed among clients and servers. Clients get a cache image of each demanded file and may operate on them even when no server may be reached. Using version vectors and file update identifiers, servers may later identify and accept the updates applied by those clients while they were disconnected. When clients remain connected to servers, they forward the updates to every reachable server in a synchronous way. Therefore, relaxed (eventual) consistency only arises at disconnection intervals and does not depend on the workload being supported. Note that other systems relied on lazy propagation (and its resulting eventual consistency) in order to shorten regular operation service time, but that was not the case in Coda. If a previously disconnected client introduces conflicting updates at reconnection time, those state divergences are reported to the user and must be manually merged. No automation is provided by default for managing these conflicts. In spite of this, subsequent releases of Coda introduced an automated reconciliation process when the application update semantics allows this. Kumar and Satyanarayanan describe an example of this kind applied to directory management [38].

In the scope of replicated relational databases, Krishnakumar and Bernstein (1991) [36] propose a system with lazy writeset propagation (but respecting causal order) where transactions may be accepted although up to N previous transactions executed in other replicas may be missing in the local node. The resulting system is not serialisable, and the resulting correctness criterion is known as N -ignorance. This eventually consistent system ensures enough guarantees for multiple distributed applications (e.g., a flight reservation system, with N being the overbooking tolerated in that system) and improves concurrency and performance up to N times when it is compared with strictly serialisable systems.

In the same field and year, Pu and Leff [55] proposed the ϵ -serialisability concept based on asynchronous writeset propagation. The resulting executions may be one-copy serialisable for writes but only ϵ -serialisable for reads, being ϵ a bound on data divergence. To this end, and due to the asynchrony in write propagation, a query (i.e., read-only) transaction may tolerate to be overlapped with up to ϵ conflicting concurrent update transactions without becoming aborted. Since the value of ϵ is configurable, this technique ranges from strict one-copy serialisability to a very relaxed system with eventual replica consistency. Four replication protocols are described in [55]: ORDUP (ordered updates, demanding a total order of updates shared by all sites), COMMU (commutative updates), RITU (read-independent timestamped updates, allowing order freedom in non-conflicting updates) and COMPE (optimistic service, looking later for conflicts and using compensating transactions in order to reach convergence). Eventual replica consistency might be implemented using, for instance, large values for ϵ combined with a COMMU or COMPE replication protocol. Note, however, that the ORDUP repli-

cation protocol does not allow any divergence among the states of replicas. Therefore, ORDUP is inherently convergent.

Bayou (1994) [62] was a replicated storage system to be used in a mobile computing environment where disconnections may frequently arise. It uses lazy propagation of updates providing eventual consistency. However, Bayou introduced *sessions* in order to give a better consistency image to its users. At the server domain the consistency is very relaxed and only eventually convergent, but on each user session the consistency could be stronger depending on the properties being enforced. To this end, Bayou proposed four user-centric consistency guarantees:

Read your writes (RYW): read operations reflect previous writes from the same process;

Monotonic reads (MR): successive reads see a non-decreasing collection of writes;

Writes follow reads (WFR): writes are propagated after the reads they depend on; and

Monotonic writes (MW): writes are propagated after writes that precede them.

The combination of WFR and RYW ensures a consistency that is similar to the data-centric causal model. When all these four guarantees are attained, the resulting image perceived by a user session is equivalent to one-copy consistency, but the actual data-centric consistency might still be very relaxed. Note that each operation being executed in a given session may be forwarded to a different server replica. Specific protocols based on version vectors were used in [62] for complying with the consistency guarantees required in sessions.

Fekete et al. (1996) [22] provide the first formal specification of an eventually consistent system. Their proposal formalises the algorithms described in [39]. It tolerates that operations were executed defining a partial order, but that order progressively tends towards a total order that is needed for reaching state convergence; i.e., operations may be reordered once run. Once the total order is decided for a sequence of operations, those operations are considered *stable* and the state in all replicas should have converged.

Yu and Vahdat (2000) [66] describe an implementation of the TACT middleware that is able to measure the current level of divergence among service replicas and to specify replica consistency requirements considering several aspects. This allows a precise control of replica divergence, based on three complementary dimensions:

1. numerical error (limits the total weight of writes applied across all replicas before being propagated to a given replica),
2. order error (limits the amount of tentative writes, subject to reordering, that may be pending at a replica), and
3. staleness (places a real-time bound on the delay of write propagation).

When all three dimensions have a zero bound, the system ensures linearisable consistency. On the other hand, when no bound is set, eventual consistency is used.

TACT may use several algorithms for ensuring that the requested bounds are respected. This defines a continuous space of replica consistency from which the user may choose the adequate level for each deployed replicated service. Indeed, different replicas from the same service may have different bounds depending, for instance, on the characteristics of the hosting computer or on the network bandwidth and delay.

Saito and Shapiro (2005) [56] provide a survey on *optimistic replication*; i.e., replication techniques that relax their concurrency control and consistency in order to achieve greater efficiency since synchronisation is avoided or, at least, minimised. Section 5 of that survey discusses eventual consistency. In that part, Saito and Shapiro provide one of the best definitions of this kind of consistency: “A replicated object is eventually consistent when it meets the following conditions, assuming that all replicas start from the same initial state: 1. at any moment, for each replica, there is a (committed) prefix of the schedule that is equivalent to a prefix of the schedule of every other replica; 2. the committed prefix of each replica grows monotonically over time; 3. all non-aborted operations in the committed prefix satisfy their preconditions; and 4. for every submitted operation α , either α or $\cancel{\alpha}$ will eventually be included in the committed prefix.”

The equivalence of *committed prefixes* among replicas allows the modelling of state convergence when no new updates are received. Their monotonic growth expresses that such convergence is (and will be) reached multiple times but it is not continuously preserved. Precondition accomplishment models the semantic correctness of these eventually consistent systems. The last condition means that in order to reach convergence and reconcile from existing conflicts, some of the submitted operations may be discarded (i.e., $\cancel{\alpha}$); for instance, applying other compensating actions that eliminate their effects.

Besides this, Saito and Shapiro classify eventually consistent systems depending on how they deal with three relevant problems related to the operations to be executed: ordering, conflicts and commitment.

Ordering refers to the scheduling policy being used for ordering the updates that define the committed prefix at each replica. Additionally, operations should be ordered in a way expected by users. Five ordering alternatives exist:

1. syntactic ordering, i.e., all nodes should follow the same operation order independently on the semantics of each operation (easy to implement but rises too many conflicts among nodes),
2. commutative operations, allowing any execution order (no conflict appears but it has a limited applicability),
3. canonical ordering (concurrent operations are ordered following an application-dependent set of rules),
4. operational transformation, implying the transformation of some operations in order to adapt their results for reaching convergence in a committed prefix (complex procedure that depends on the application semantics), and
5. semantic optimisation (again, too complex).

Conflicts refer to how state conflicts are dealt with and resolved. Two alternatives:

1. syntactic (differences – either in the state values or in the operation order – are used for detecting conflicts and a deterministic criterion is used for resolving them), and
2. semantic (conflicts are resolved considering the semantics of the involved operations; this is a complex and application-specific solution).

Finally, *commitment* refers to the protocols being used for deciding when an executed operation can be considered stable (i.e., it has been accepted and belongs to a common committed prefix in all replicas), or for reaching agreement on non-deterministic decisions. There are three alternatives:

1. implicit (no operation is ever explicitly rejected; this may be supported using the “last writer wins” approach in case of conflicts),
2. background agreement (nodes send piggybacked information about their accepted updates in their update propagation messages; e.g., using version vectors), and
3. consensus (this is a complex and potentially blocking solution, usually needed in strongly consistent systems but not recommended in eventual ones).

One of the conclusions that may be extracted from [56] is that concurrency control could be avoided in eventually consistent systems. To this end, convergent and commutative replicated data types (CRDTs) [60] were proposed by Shapiro et al. as a set of replicated data types able to easily converge by themselves without needing any concurrency control mechanism. CRDTs are based either on commutative operations (CmRDTs), giving some rules, advices and examples for transforming regular operations into other commutative variants, or on a *merge* operation for state-based replication protocols (CvRDTs), introducing commutativity at update application time in those protocols. Thus, in both cases, the system achieves convergence ensuring that all incoming client requests are eventually executed by every replica.

This historical review would end here. Its goal has been to show that some research work on eventual consistency existed before Vogels’ paper [64] was written. We have centred our discussion on the papers written before 1994, since subsequent papers have been thoroughly reviewed in other works; e.g., by Saito and Shapiro [56]. A summary of the contributions found in our review is given in Table 1. However, as it has been said in Section 1, there have been many recent papers on this subject and there is still an aspect that had not been completely dealt with before 2008: a formal specification for eventual consistency. There have been a few papers covering that goal [22, 13, 10] and the two latter deserve some comments since they have provided valuable contributions.

Bouajjani et al. [10] and Burckhardt [13] criticise that almost all previous definitions of eventual consistency had been only centred in state convergence at quies-

Reference	Contributions	Year
Johnson and Thomas [34]	Multi-master replication as a way for implementing eventual consistency. Description of a basis for logical clocks. Mechanism for totally ordering the events in a system. Network partition tolerance. “Last writer wins” principle for reaching convergence. Management of delete-update and delete-create conflicts.	1975
Alsberg and Day [3]	Multi-master replication with commutable operations as a way for implementing eventual consistency.	1976
Lindsay et al. [44]	Identification of the level of update propagation synchrony as a key aspect for replica convergence: strong consistency with synchronous propagation and eventual consistency with lazy propagation.	1979
Parker et al. [52]	Version vectors for detecting inconsistencies in disconnected operation. Potential causal consistency as a base for implementing eventual consistency. Need of semantic reconciliation protocols at reconnection time.	1981
Birrell et al. [9]	Deployment of a WAN system supporting eventual consistency.	1982
Fischer and Michael [25]	Avoidance of delete-update and delete-create conflicts in eventually consistent services using multi-master replication.	1982
Davidson [17]	Service continuity in partitioned databases with serialisable histories. Criteria for choosing which transactions to roll back at reconnection time.	1984
Apers and Wiederhold [4]	Service continuity in partitioned relational databases. Consideration of correctness invariants (stated as pre- and post-conditions) at partition reconnection time.	1985
Sarin et al. [57]	Service continuity in partitioned databases. Semantic criteria for reordering or avoiding compensating actions at partition reconnection.	1985
Demers et al. [21]	Analysis of three types of lazy update propagation.	1987
Kawell et al. [35]	Proposal of a CSCW application (Lotus Notes) with eventual consistency. Pull-based strategy for update propagation. Document propagation for reaching convergence.	1988
Kumar and Stonebraker [37]	Extension of the <i>escrow</i> method (management of commutative transactions respecting value constraints) to replicated databases. Serialisability is sacrificed and inter-replica consistency is relaxed, but transaction correctness is maintained.	1988
Satyanarayanan et al. [58]	Support for client disconnected operation in distributed filesystems. Manual multi-conflict resolution may be demanded at reconnection time.	1990
Ladin et al. [39]	Specification of update ordering requirements, depending on application semantics. Explicit causal dependences as a base for building eventual consistency. Global order for reaching convergence on the set of processed operations in every replica.	1990
Krishnakumar and Bernstein [36]	N-ignorance as an eventually consistent example in replicated relational databases. Lazy writeset propagation with causal order.	1991
Pu and Leff [55]	ϵ -serialisability (relaxed consistency for query transactions in relational databases). Proposal of four replication protocols using asynchronous writeset propagation. Some protocols (COMMU and COMPE) may implement eventual consistency.	1991
Terry et al. [62]	Specification of user-centric consistency conditions, instead of the traditional data-centric ones. Sessions for providing user-centric consistency. Users perceive a consistency that is stronger than that maintained by servers.	1994
Fekete et al. [22]	Characterisation of eventual consistency based on stable operations (that force state convergence) and reorderable operations, using I/O automata. First formal specification for eventual consistency. Inspired in the replication protocols proposed in [39].	1996
Yu and Vahdat [66]	Evaluation of replica divergence. Selection of a per-replica level of consistency. Specification of consistency based on three axes: (1) numerical error, (2) order error, and (3) staleness.	2000
Saito and Shapiro [56]	Survey on optimistic replication techniques, including eventual consistency. Thorough classification of eventually consistent systems.	2005
Bouajjani et al. [10]	Complete formal specification of eventual consistency. Definition of weak eventual consistency, centred only in convergence. Consideration of local program correctness in its consistency specification. Proposal of verification tools for eventually consistent systems.	2014

Table 1. Contributions outlined in this historical review

cence intervals. However, those definitions and specifications did not state anything about traces where no quiescent interval exists. In those cases, apparently, no state convergence effort is required. In spite of this, most eventually consistent services actually consider and make those efforts. Therefore, the specification given in [10] considers both:

1. the correctness of the operations being executed by each process, and
2. the conditions to be satisfied when the arrival of new updating requests never stops.

Additionally, that paper qualifies quiescent convergence as *weak eventual consistency* and proposes some verification tools for evaluating the correctness of (both weak and non-weak) eventually consistent systems.

3 EVENTUAL DATA-CENTRIC CONSISTENCY MODELS

Distributed shared memory (DSM) consistency models [49] assume that multiple processors share a given memory and that processes directly run in those processors. Thus, those models are focused on how data is shared by processes. Those classical memory consistency models are known nowadays as data-centric consistency models and they are also used in distributed systems [61]. In this section, the term *consistency model* will always refer to those data-centric models.

The state convergence assumed in eventual consistency is a liveness property [23, 13, 54]. Therefore, the conditions required in several definitions of eventual consistency [56, 64] could be respected adding a convergence property to a relaxed consistency model. However, there is no agreement on where to place the borderline between strong and relaxed consistency models. Depending on the problem being considered, that frontier separates different sets of strong and relaxed models. For instance, the *linearisable* [32] and *sequential* [42] models are in the strong set when database one-copy equivalence with serialisable isolation is being considered [23]. On the other hand, Attiya [6] states that only the linearisable model is strong enough to avoid old-new inversions in read accesses onto shared variables. The sequential model does not avoid that kind of inconsistency in its reads.

In the scope of eventual consistency characterisation, two frontiers should be defined separating these three sets of models [54]:

Strong models: A *strong* model guarantees state convergence among replicas on each write action. This means that write actions cannot overlap. As a result of this, no read old-new inversion is possible. The linearisable [32] (or atomic [43]) model is in this strong group [6].

Convergent models: A model is inherently *convergent* when the conditions that define the model imply state convergence. In this case write actions may overlap and read old-new inversions may happen when those read actions occur in different replicas. However, once all value propagations for concurrent write actions are delivered, the state in all replicas reaches convergence again.

Relaxed models: A model is *relaxed* if it does not guarantee convergence when all its consistency conditions are respected. FIFO consistency is an example of relaxed model since it only requires that the writes of each process are applied in writing order on the other replicas, allowing any interleaving of the writes made by different processes. Because of this, different receivers may see different values on a given set of variables when they have applied all their incoming updates.

Section 2 has shown that eventual consistency was generated as a reaction to the constraints of the CAP theorem [28]. Being eventually consistent, a service may remain available to all its clients even when the network that interconnects its replicas gets partitioned. Thus, inherently convergent models are all those on which the CAP constraints may apply; i.e., those that require consensus on the order of writes among all participating processes [54], since that consensus cannot be reached in an asynchronous distributed system when its processes may fail or get disconnected [24]. That requirement already applies to the *cache* [30] consistency model and all those stricter than cache. However, no need of consensus exists in the *slow* [33], *FIFO/PRAM* [45] and *causal* [1] models.

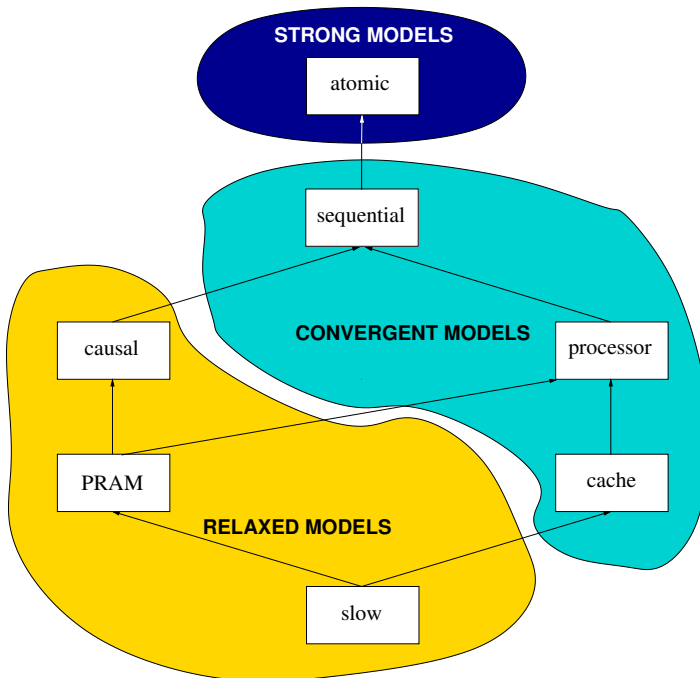


Figure 1. Strong, convergent and relaxed models

Figure 1 shows the *weaker-than* (\longrightarrow) relations among data-centric models. They were already identified by Mosberger [49] in 1993. Taking those relations as a base, it is easy to select which consistency models belong to each set, defining in this way the intended borders.

These identified frontiers have a first consequence: all inherently convergent models (i.e., sequential, processor, cache) are stronger than *eventual consistency*. Note that a model A is stronger than another model B if all executions accepted by model A comply with the constraints of model B and there is some execution that respects model B but does not respect the constraints imposed by model A [63]. Inherently convergent models are eventually consistent, since they reach convergence once a given set of write actions is known and applied by every process. However, not all eventually consistent executions are inherently convergent. For instance, the following execution:

$$\begin{aligned} W2(x)3, W1(x)2, R3(x)3, R4(x)2, R3(x)2, R2(x)2, \\ R4(x)3, R1(x)3, W4(x)5, R2(x)5, R3(x)5, R1(x)5 \end{aligned} \quad (1)$$

... is eventually consistent, since at the end all its four processes converge to value $x = 5$; but it is not sequential, nor processor, nor cache. Its strongest respected model is causal. Processes P2 and P3 have seen the sequence 3, 2, 5 on x and P1 and P4 have seen the sequence 2, 3, 5. When the written values 2 and 3 had been delivered at all processes, before value 5 is written, the state of these four processes did not converge. Therefore, it is not inherently convergent.

Up to our knowledge, this relationship among inherently convergent models and eventual consistency has not been explicitly identified yet. Indeed, Viotti and Vukolić [63] (2016) revise many distributed consistency models, trying to identify their relations in order to build a global hierarchy. In that hierarchy, they distinguish multiple variants of eventual consistency: basic eventual [13], strong eventual [13], eventual serialisability [22] and eventual linearisability. They only state that the *strong eventual* and the *eventual linearisability* models, being the strongest in that set, are weaker than the *linearisable* model. No relation is identified among all those four variants of eventual consistency and other data-centric models (besides *linearisable* consistency). However, our Execution 1 illustrates that the *basic eventual* model is weaker than *cache* consistency. Additionally, *cache* executions comply with the requirements of the *strong eventual* model (to reach convergence once the same set of update operations are delivered to all processes), and Execution 2 is an example of execution based on commutative operations, assuming an initial value of zero for variable x in every process, that is *strong eventual* but not *cache* consistent (P2 and P3 have observed sequence 3, 5 since they have received the operations in the order $+3, +2$; while P1 and P4 have observed sequence 2, 5, since they received those operations in the order $+2, +3$). This means that the *strong eventual* model is weaker than *cache* consistency. Therefore, these relations exist: *basic eventual* \longrightarrow *strong eventual* \longrightarrow *cache* \longrightarrow *processor* \longrightarrow *sequential* \longrightarrow *linearisable*

(*atomic*). Thus, there are several models between the *strong eventual* to *linearisable* weaker-than relation.

$$W2(x) + 3, W1(x) + 2, R3(x)3, R4(x)2, R3(x)5, R2(x)5, R4(x)5, R1(x)5 \quad (2)$$

Regarding network partitions, our proposed frontier among convergent and relaxed models has another interesting consequence. Fox and Brewer [26] stated the CAP theorem assuming that “consistency” was atomic (i.e., strong). Gilbert and Lynch [28] proved that same theorem respecting that assumption. However, Pascual-Miret et al. [53] have proven that even cache consistency cannot be respected by available replicated services while the network is partitioned. Their proofs are quite complex. A more intuitive proving argument of that fact has been already given here for setting the borderline between convergent and relaxed models. *Cache* consistency requires agreement on the order of writes on each variable. That agreement requires consensus among all the processes that replicate shared variables. Consensus cannot be achieved in a system where processes may fail [24] (or where they remain unreachable due to a network partition). Therefore, the CAP theorem not only applies to atomically consistent services but to all replicated services that are strong or convergent according to our classification.

Depending on the consistency requirements of the service, two alternatives exist for implementing eventual consistency in a system that tolerates network partitions:

1. to use slow or PRAM replica consistency (or even no consistency at all) when optimal throughput is the main goal, or
2. to use causal replica consistency when at least a causal behaviour is expected, partially sacrificing performance in this case.

Both approaches should rely on supplementary data convergence mechanisms.

4 IMPLEMENTATION APPROACHES

The problem of guaranteeing state convergence only arises when a replicated service exists. The state being managed by that service may use an optimistic replication technique [56] because in this way it reduces overall overhead and tolerates network partitions. In that case, service replicas become eventually consistent.

At a glance, this means that a replication protocol should be chosen, and its optimism may rely on lazy update propagation, introducing some degree of asynchrony in the interactions among replicas. However, other characteristics of inter-replica management may be relevant. Thus, a careful evaluation of those characteristics is needed.

In order to provide a characterisation of the existing implementation approaches, Section 4.1 analyses which elements should be considered to define an implementation strategy for eventual consistency. With that base, Section 4.2 provides a performance analysis of those implementation strategies. Later, Section 4.3 summarises

which systems were the first using each one of the implementation strategies identified in Section 4.1, complementing in this way the historical review presented in Section 2.

4.1 Four Elements to Implement Eventual Consistency

Several aspects should be considered to implement any replication strategy. Let us revise them, focusing on their effect on performance and replica convergence. In each aspect, several implementation approaches are enumerated. After their name, in parentheses, we show an abbreviation in order to refer to them later on:

Replication protocol. The replication protocol rules the steps to be followed by the service replicas in order to manage a given operation submitted by a client agent. Several types of replication protocols may be found:

Primary copy (PC): One of the server replicas is distinguished as the primary replica. All update operations must be forwarded to that primary, who is the unique replica that may process the updates. Once an operation has been processed by the primary, its effects are collected and forwarded to the remaining replicas that will accept and apply those updates in order to reach convergence with the state of that primary. Primary copy replication protocols follow the *primary-backup* [11] replication model.

Multi-master (MM): Each operation is processed by a single replica (also known as *master* for that operation service) who later propagates the resulting updates (if any) to the remaining replicas. However, in this case, each operation may be forwarded to any replica. No primary exists in this type of protocol. Thus, operations may be served by different masters, increasing in this way the degree of concurrency. This strategy might generate inconsistencies if the operations being served by different masters are conflicting.

Quorum-based (QB): Operations are classified as queries (i.e., read-only operations) or updates (i.e., those operations that create, delete or modify at least one data element). Queries need to be executed at as many replicas as stated in their *read quorum* (RQ) while updates should reach at least a *write quorum* (WQ). Quorum-based protocols were originally defined [27] for achieving strong consistency. To this end, assuming that there are N replicas in the system and that each replica has one vote, two rules should be respected:

1. $WQ > N/2$, and
2. $WQ + RQ > N$ votes.

In this way, it is guaranteed that conflicting operations will have a non-empty intersection of replicas. These traditional quorum-based protocols may be tagged as *strict quorum* protocols (QB_S) [7].

Modern NoSQL scalable datastores (e.g., Cassandra [40]) admit multiple quorum sizes, allowing varying degrees of consistency. Indeed, they are able

to provide relaxed eventual consistency when quorums do not need to intersect for accepting an operation [29]. This second kind of quorum-based protocols can be tagged as *partial quorum* protocols (QB_P) [7].

Operation ordering. Eventually consistent services do not demand state convergence after processing each operation. Instead of this, the state in different replicas diverges at some intervals and will reach again convergence afterwards. This state convergence may depend on the operation semantics and ordering. Let us analyse which alternatives exist in this area:

None (NO). When all the operations in the interface of a given service are commutative, there is a complete freedom on the order of execution of the incoming requests. Once every replica has executed the same set (i.e., unordered collection) of operations, all those replicas will be convergent. This eliminates the need of inter-replica coordination (e.g., in MM replication protocols), being the optimal solution for improving replication throughput. Commutative replicated data types (CRDTs) [59, 60], also known as *conflict-free replicated data types*, have been proposed by Shapiro and Preguiça in order to eliminate operation ordering requirements in replicated data types. They provide a useful guide for designing data types with commutative operations and for avoiding conflicts when non-commutative operations exist.

Partial order (PO). At a glance, non-commutative operations need to be executed in order to ensure replica convergence. However, even in that case, update operations that might seem conflicting may be executed in any order when they are updating disjoint parts of the shared state. As a result, the global order to be considered becomes partial and this means that concurrent (and unordered) service is tolerated for a subset of the operation requests. In spite of this, some degree of coordination among replicas is needed, reducing the service throughput.

Total order (TO). In some cases all updating operations are conflicting and they should be executed in a global total order by every replica. This is the regular behaviour in strong consistency protocols and should be avoided if high performance is the main goal.

Synchronisation degrees in agent interaction. Distributed services usually follow a client/server interaction pattern. In a strongly consistent service, when a client agent requests an updating operation to a replicated server, five interaction steps may be distinguished:

1. the client sends the request to a master replica;
2. the master replica processes the request;
3. the master replica sends the state updates to every slave replica;
4. slave replicas acknowledge the completion of the application of those updates on their local copies; and
5. the master replica replies to the client.

Thus, the client agent remains blocked until the reply is returned to it. The master replica is also waiting for the acknowledgements sent by the slave replicas. This means that strongly synchronous communication is needed. The first primary-backup [3] replication protocols followed that same pattern, ensuring linearisable consistency.

However, in eventually consistent services that level of synchrony is unneeded. For instance, query requests only demand steps 1, 2 and 5, but they can be served by any replica (not necessarily the primary one) and pure updating requests (that return no result) may be completely asynchronous for the client, who will be only involved in step 1. Additionally, in this latter case, a master replica may only execute steps 2 and 3, without being involved in steps 4 and 5.

On the other hand, when clients expect a reply from their updates, eventual consistency admits lazy update propagation. Thus, the serving replica may execute steps 2 and 5 as soon as possible, executing later step 3 in a lazy way (either followed by step 4 or not, depending on the reliability of the communication channels).

Therefore, multiple degrees of agent interaction synchrony are possible in replicated systems. Since query operations always demand a reply, let us centre our attention on how updating operations may be processed. The following alternatives exist:

Asynchronous (A): When updates do not need any reply and server state propagation is done in a lazy way. This is the optimal solution regarding throughput.

One synchronous interaction (1S): When either:

1. the update requires an answer and server state propagation is done in a lazy way, or
2. the update does not need any answer but server state propagation is synchronous.

Two synchronous interactions (2S): When the five steps described above are done in a synchronous way. This only happens in strongly consistent services.

State convergence strategy. The state of different replicas may become divergent from time to time. In those cases, some strategy is needed in order to fix those divergences. That strategy should be able to, first, identify the differences among multiple replicas and, later, decide how to merge those diverging states into a convergent one. The following alternatives exist to this end:

Unneeded (UN). In some applications, state divergences only arise while an updating request that has been processed at a replica is not yet known by the remaining replicas. Once the other replicas receive and process that request, state convergence is restored. This happens, for instance, when all

service operations are commutative. Note that in that case, the replication protocol being used should be based on “operation transfer” instead of “state transfer”.

In case of network partitions, each subgroup should remember the set of operations that they have processed while the network was partitioned, in order to transfer that set to the remaining subgroups when connectivity is restored. This allows that every operation executed at every subgroup were considered and accepted in the resulting convergent state.

This strategy does not need any divergence detection mechanism.

Overwrite (OW). Some types of simple applications (e.g., directories, calendars, . . .) may use databases that hold a collection of independent elements that are seldom updated. Additionally, the computational effort for those updates is minimal. In those cases, when conflicts arise, the application is interested in the newest updating attempt. All previous ones may be overwritten by that latest one.

These applications only need to tag the updating actions with a (logical) timestamp, as it was suggested in [34]. In case of conflicts, the merged state will only hold the newest update. The detection and resolution of conflicts may be easily automated with this strategy.

Reordering (RO). When the updating operations applied in a concurrent way at different master nodes or partitions depend on the previous state and are conflicting, only one of them might be accepted according to the application semantics. In that scenario, those other conflicting operations should be discarded (using backward recovery in the nodes where they had been previously applied) and restarted. This implies an operation reordering. In some cases, the reordering may be automated if there are deterministic criteria that rule those reordering decisions.

Manual convergence (MC). In some cases, there are no deterministic criteria to schedule conflicting operations that were rejected in the state merging procedure. Therefore a manual merging approach is needed in that case.

From the point of view of performance, the second alternative (OW) is the best one, since it only needs to find the newest state and apply it to the remaining replicas. Additionally, the mechanisms needed in that case may be simple (logical timestamps or version vectors). Unfortunately, that strategy is not generally applicable.

Commutative operations also simplify the convergence approaches. Nothing special is needed, although missing requests should be run in those nodes that had not seen them. This might take a long time in case of prolonged network partitions.

4.2 A Guide for Performance Analysis

Considering only performance, the MM-NO-A-OW (multi-master, with no ordering requirements, asynchronous interactions and overwrite-based merging) or MM-NO-A-UN combinations of strategies seem to be the best ones. Up to our knowledge, no complete performance comparison, including all identified strategies, has been made yet. In spite of this, some research papers have analysed and compared some of the alternatives discussed in any of those aspects.

For instance, de Juan-Marín et al. (2007) [19] presents a performance evaluation of different primary-copy algorithms depending on their degree of communication synchrony. A completely asynchronous interaction (i.e., using the A case) was able to complete an update operation in less than 2 ms (the processing time at the server side was around 1 ms), while the same request demanded at least 25 ms in the 2S case. In the scenarios considered in [19], synchronous interactions may worsen communication time up to 20 times.

Golab et al. (2014) [29] propose the *gamma* client-centric metric for benchmarking consistency. To this end, the Cassandra scalable datastore is used in [29]. It uses a QB replication protocol. In their tested configurations with a “hotspot” distribution, only a “read one-write one” (RQ:1, WQ:1) quorum provided a higher proportion of consistency anomalies than strict quorums (e.g., “read all-write all” or “read a majority-write a majority”); 1.3% vs. 0.6%. On the other hand, with its “latest” distribution, the “read one-write one” quorum and the “read one-write a majority” (RQ:1, WQ>N/2) provided more anomalies than strongly consistent quorums (1.3% and 0.6%, respectively, versus 0.1%). This means that the most relaxed QB protocols (RQ:1, WQ:1), as expected, introduce more inconsistencies than the traditional strongly consistent QB protocols. Such difference is up to 13 times greater in the worst case, but it is only twice greater in the common case.

Bailis et al. (2014) [7] propose eventual consistency with probabilistic bounded staleness (PBS). Like [29], PBS uses QB_P protocols. Since quorum-based protocols use a read quorum for handling their read accesses, they may obtain up to RQ different values in each read. So, PBS provides (K, Δ, p) -regular semantics [63]. Bailis et al. evaluate the effects on both response time and consistency of different system configurations. Thus, when response time is assessed using a workload model that represents a peak interval in the LinkedIn servers (an example of IO-bound workload) in a system with three replicas with $p = 99.9\%$, the following values are obtained for different quorum sizes:

- $RQ = 1, WQ = 1$: Both read and write latencies of 0.66 ms, but requiring 1.85 ms for ensuring the intended (Δ, p) -regular semantics; i.e., this introduces a non-negligible staleness interval.
- $RQ = 2, WQ = 1$: Read latency of 1.63 ms, with write latency of 0.65 ms, but without any staleness interval in spite of being a QB_P configuration.
- $RQ = 2, WQ = 2$: Read latency of 1.62 ms, with write latency of 1.64 ms, and without staleness, since it is the smallest QB_S configuration.

- $RQ = 1$, $WQ = 3$: Read latency of 0.65 ms, with write latency of 4.09 ms, without staleness. This is the standard ROWA configuration, the recommended QB_S deployment for read-intensive applications.

That evaluation shows that those QB_P configurations which minimise response time are subject to returning stale values. However, there are other QB_P configurations that do not read stale values and still provide better response times than the most efficient QB_S configuration. Let us call *overall latency* the sum of both read and write latencies. QB_P configurations have shown an overall latency between 1.32 and 2.28 ms, while QB_S ones have demanded between 3.26 and 4.74 ms. This shows that QB_P has been, as a minimum, 43% faster than QB_S (and more than N times faster in the best case) and, in some cases, they have avoided staleness in their read values.

These partial evaluations could be extended. Section 4.1 has identified the four aspects to consider in every implementation of eventual consistency. Each aspect provides multiple implementation approaches (3 in the general case, but there are 4 state convergence strategies) and this provides a total of $3^3 \cdot 4 = 108$ combinations.

Assuming this diversity of possible implementation approaches, developers of scalable services need some advice on the best choice in each implementation aspect. Those advices depend on which are the main goals for that service and on the priorities given for each objective. Although each application may have its specific goals, there are some common objectives for all of them. They are:

Minimal response time (mRT). Services that use eventual consistency try to optimise their service time and the response time being perceived by their clients.

Minimal convergence effort (mCE) when no disconnection arises. When no network partition happens, this goal measures the effort being required by the assessed mechanism in order to reach inter-replica convergence.

Minimal convergence recovery time (mCT) once a network partition is healed. This depends on the communication rounds being needed for transferring any missed state, and on the concrete protocol to be used for joining that missed state with the local state on each replica.

Minimal transfer size (mTS) once a network partition is healed. The size of the state to be transferred (or the amount of operations to be propagated) conditions the time being needed for recovering convergence. It is a critical goal for mobile computers; e.g., laptops using CSCW applications [35].

Depending on a given prioritised group of goals, programmers need a guide about which combination of implementation approaches on each aspect is the best one for achieving them. At the moment, that guide is missing but it could be written once some prototypes of the approaches identified in Section 4.1 were deployed and evaluated. In order to partially fill this gap, an overall evaluation of those approaches is given in Table 2. Four levels of achievement are considered: irrelevant (\times), poor ($-$), moderate ($+$), and good ($++$). A mechanism is qualified as *irrelevant* (\times) when

its effect is null on a given objective. This means that it is not directly related with that goal.

Axis and Mechanism	Goals			
	mRT	mCE	mCT	mTS
Replication protocol				
Primary copy	+	++	++	×
Multi-master	++	-/+	+	×
Quorum-based (strict)	-	-	×	×
Quorum-based (partial)	++	-/+	+	×
Operation ordering				
None	++	+	×	×
Partial order	+	+	×	×
Total order	-	+	×	×
Synchronisation degree				
Asynchronous	++	+	×	×
One-synchronous	+	+	×	×
Two-synchronous	-	+	×	×
Convergence strategy				
Unneeded	++	++	+	-
Overwrite	++	+	++	++
Reordering	-	×	-	-
Manual convergence	-	×	-	-

Table 2. Goals achievement level for each implementation approach

Regarding the mCE goal, there is no difference among the operation ordering and synchronisation degree variants because in both cases there are two aspects to consider: the degree of achieved convergence and the effort being needed. Thus, in the operation ordering axis, when the achieved convergence is considered, the marks are: no order (-), partial order (+), total order (++), since when every replica applies all the operations in total order, convergence is trivially achieved; but when the needed effort is taken into account, the marks are just the contrary: no order (++), partial order (+), total order (-), since total order requires multiple rounds of message exchange for being achieved [18], while “no order” does not require any effort. So, all the alternatives receive a moderate (+) global mark. The same happens in the synchronisation degree axis.

Considering what is shown in Table 2, the most recommendable implementation approaches for covering each goal are:

1. *Minimal response time* (mRT): MM-NO-A-UN, MM-NO-A-OW, QB_P-NO-A-UN, QB_P-NO-A-OW.

Regarding the replication protocol, MM and QB_P are the best approaches since they do not require any coordination among concurrently served requests. Thus, they tolerate high concurrency without introducing any inter-request blocking.

Therefore, their mark is good ($++$). To this end [3], the operations being managed should be commutative. PC may be also combined with lazy replication but compel every operation request to be forwarded to the same primary replica. In the end, with heavy workloads, this might collapse such single replica, while that same workload could be shared out among multiple masters in MM. This explains why the PC mark is lower ($+$) than the MM and QB_P ones. QB_S totally order conflicting write operations, using to this end its quorum-based concurrency control algorithm. Those algorithms provide higher response time than QB_P (as shown in [7]) and are prone to deadlocks that may be avoided using some kind of priority management among the competing concurrent lock requestors. However, that management introduces either:

- (a) the need of additional communication [46] for deciding whether a given low priority vote may still be changed or not, or
- (b) the abortion of low priority requests in case of conflict [14].

In the end, this introduces a non-negligible overhead that penalises the response time being perceived by client processes.

The NO and A convenience in the second and third axes has already been explained.

Finally, considering the convergence strategy, both RO and MC have a poor ($-$) mark since the former needs to cancel and reapply operations to implement its reordering and the latter requires human intervention. In both cases, the time interval needed for those tasks becomes long, although RO automates its tasks, being slightly better than MC. On the other hand, UN is based on non-conflicting operations. So, no concurrency control mechanism is needed and no operation abortion or rollback may arise [59, 60, 2]. This deserves a good ($++$) mark. Such mark is shared by OW in case of objects of small size, since it also avoids operation re-execution and immediately determines whether an incoming state propagation should be applied or rejected in a receiver replica. The timestamps needed for handling this can be built without problems [34].

2. *Minimal convergence effort* (mCE): There is a tie between all alternatives in both the *operation ordering* and *synchronisation degree* axes. So, all their alternatives seem to be equally adequate in this goal. The other two axes do not have ties among their best choices, but it is worth noting that *convergence strategy* is the main axis to be considered regarding this mCE goal. The replication protocol to be used has a low impact in the final effort.

PC is able to guarantee convergence without much effort, since all replicas accept what the primary propagates and no conflicts arise. It obtains a good ($++$) mark. QB_S also guarantees convergence, but requires a careful voting management that may lead to deadlocks in case of conflicts. Its mark is poor ($-$). Finally, both MM and QB_P do not take care of conflicts in their default protocols. This guarantees a very good performance and a minimal effort. However,

they do not guarantee convergence by default if any pair of conflicting concurrent update operations exist. Thus, they also receive a poor (–) mark, although it becomes moderate (+) if conflicts are rare.

Regarding convergence strategies, RO and MC are mechanisms needed to recover convergence once it has been lost due to a network partition when every resulting service subgroup may go on, maintaining their availability. Those techniques are not regularly applied when no disconnection has arised. UN is based on defining conflict-free data types. The effort should be applied at object definition time. Once its feasibility is confirmed, there are no implementation nor execution overheads. On the other hand, OW demands operation timestamping. Such timestamping may be based on global logical counters [34] that are trivially built, but introduce a non-negligible overhead on message size. This explains why the UN mark (++) is better than the OW one (+) in this goal. As a result, every *-*-UN combination is worth to be considered here.

3. *Minimal convergence recovery time* (mCT): This goal considers the time spent in replica state reconciliation once a network partition is healed. The synchronisation degree axis is irrelevant since it deals with inter-requests ordering and recovery messages do not participate in that order. For the same reason, operation ordering is also irrelevant.

Regarding replication protocols, QB_S should be discarded, since it was defined for achieving strong consistency and adopts the *primary partition* model [15] in case of disconnection. Apparently, PC shares the same problem, but it has an important difference: the primary copy is an inherent leader process and behaves as a group representative for state transference once the network partition disappears [20], assuming that there is a primary process per partition [5]. Thus, PC becomes the best choice in this regard. On the other hand, MM and QB_P may need some intra-group coordination among their server processes in order to find out which are the set of updates to be transferred. That knowledge is immediately achieved in PC.

Finally, among the existing convergence strategies, OW is the best since it selects the newest value in case of conflict. This means that once the network connectivity is recovered a single version of each updated object needs to be transferred. This minimises the amount of needed messages and their size. UN needs to transfer all missed operations among process groups. This demands more time and space than in OW. So, its mark (+) is worse than that of OW. Finally, RO demands operation cancellation and re-execution for merging the missed updates in order to define an agreed history sequence. This is still more time demanding than in UN. MC demands human intervention and this implies larger intervals than every other (automated) solution.

Taking into account all these aspects, every PC-*-*OW combination is worth considering in this goal.

4. *Minimal transfer size* (mTS): This goal is highly related with the previous one, but the replication protocol is unimportant in this case, since that protocol does not condition what data needs to be transferred. As a result, only the convergence strategy should be assessed.

OW provides an immediate criterion for reducing the size of the data to be transferred in the reconciliation stage. It only propagates the latest known version of each modified object. This deserves a good mark (++).

UN and RO need to transfer to the reconnected system groups either their missed operations or their effects, since convergence is usually recovered once every operation is known by every system process. This usually means a larger transfer size than in OW, since OW has a clear criterion for selecting a single value per element, while these two other approaches have no transfer minimising criterion. This is the worst possible behaviour in this regard and it explains their poor mark (-).

MC requires human intervention to solve conflicts. The amount of data to be applied may be minimal (the user decides, and such decision may be better than other automated ones, since users may consider many more aspects than those included in a program). However, human conflict resolution needs to manage at least a summary of the conflicting operations. That information should be transferred to the computer that manages user interaction for conflict resolution. Thus, in the end, this alternative needs two data transfer phases, since later on, such decision should be announced and the results or the code of the accepted operations should be propagated to the participating nodes that had missed their effects. The overall size and cost of these approaches strongly depend on the concrete mechanisms being used in each phase. Thus, its mark is left blank since it is system-dependent, but it could not be better than moderate (+) since it requires two data transfer phases.

The best combinations in this goal are those that match the *-*-*OW pattern. Although UN, RO and MC are not recommended, they only need some criterion to reduce the size of missed updates before transferring them in order to improve their mark. Some criteria examples have been proposed elsewhere. For instance, delta-state CRDTs [2] reduce the amount of elements to be modified by each pending operation, improving in this way the behaviour of UN. Also, they transfer only the latest value of the updated elements in case of long lasting partitions. With those extensions, the mark differences between OW and other mechanisms may vanish.

An exhaustive experimental evaluation of all existing alternatives for implementing eventual consistency is unapproachable, since there are too many combinations to be considered. Table 2 has shown a high level assessment of all those alternatives, providing a first guide for choosing the best basic approaches in each concrete goal.

No single combination is the best for all possible goals. In some proposals [2], this has led to the implementation of several mechanisms in a particular axis, using the best of them in each particular scenario.

4.3 Strategies along Time

With the aim of complementing our historical revision started in Section 2, Table 3 shows which papers were the first using some combinations of implementation strategies discussed in this section. In order to have a wider variety, some strongly consistent protocols are also in the table.

The table shows how each protocol deals with the CAP theorem when a network partition occurs. In that case, each protocol needs to decide whether consistency is relaxed or availability is sacrificed. In the A column, a Y (yes) means that availability is maintained in every replica, while an N (no) means that some replicas remain unavailable. The C column shows which is the strongest consistency model being supported by the available replicas while the network remains partitioned. The “relaxed” value means that any of the relaxed models identified in Section 3, or even none at all, is enough in that proposal since it is assuming commutative operations.

The entry for *Alsberg and Day* shows the information about the variant of their protocol that admits multi-master management with commutative operations. The 2PC abbreviation represents the two phase commit protocol explained in [44] in order to decide the fate of distributed transactions. That final protocol introduces two synchronous communication stages at the end of each distributed transaction. Despite tolerating lazy propagation, the algorithms described by Lindsay et al. do not admit progress in all subgroups when the network is partitioned.

Paper	Repl. prot.	Order	Sync.	Converg.	CAP mgmnt.		Year
					C	A	
Johnson and Thomas [34]	MM	NO, PO	A, 1S	OW	FIFO	Y	1975
Bunch [12]	PC	TO	2S	N/A	strong	N	1975
Alsberg and Day [3]	MM	NO	1S	UN	relaxed	Y	1976
Lindsay et al. [44]	PC, QB	PO	2S	2PC	strong	N	1979
Parker et al. [52]	MM	PO	1S	MC	causal	Y	1981
Apers and Wiederhold [4]	–	PO	1S	RO	strong	N	1985
Ladin et al. [39]	MM	PO, TO	1S	UN	causal	Y	1990
Shapiro and Preguiça [59]	MM	NO,PO	A	UN	relaxed	Y	2007
Almeida et al. [2]	MM	NO	A	UN	relaxed	Y	2015

Table 3. Implementation strategies used in several proposals

Table 3 shows that one of the first implementations of eventual consistency, that of Johnson and Thomas (1975) [34], already provided one of the best combinations of implementation approaches regarding response time: MM-NO-A-OW.

However, its OW strategy for state convergence cannot be used in every application; it is adequate for small objects like those assumed in [34]. Until 2007, with the proposal of CRDTs [59], we were not able to find a solution of similar quality (MM-NO-A-UN). However, its NO approach for operation ordering and UN strategy for state convergence is based on commutative operations, that require their re-execution at every replica. Besides, CRDTs require partial order (causal propagation) in case of using their state-based variant. Delta-based CRDTs [2] fix that problem, since they use the same solution (MM-NO-A-UN) but relying on partial delta state transfers, avoiding in this way potentially costly re-executions at reception time. This guarantees an excellent performance. Due to this, delta-CRDTs are used in the implementation of some modern NoSQL databases, like Riak and Cassandra.

5 CONCLUSIONS

A discussion on several aspects of eventual consistency has been provided. Eventual consistency is basically a liveness property (data convergence) to be added to relaxed consistency models. Due to this, it had not had a formal specification similar to other consistency conditions since they have been usually specified as safety constraints. However, Bouajjani et al. [10] and Burckhardt [13] have recently proposed specifications that carefully characterise safety (program correctness) and liveness (eventual state convergence) correctness conditions for eventually consistent services.

Although eventual consistency has received a lot of attention nowadays when elastic and geo-replicated distributed services have been developed, it was already suggested in several papers 40 years ago. Therefore, it is not a new concept. A short historical review has been presented, describing some of the oldest although relevant works in this subject.

The border between inherently convergent and relaxed models has been set. Those relaxed models (e.g., PRAM and causal) may be taken as a basis for implementing eventually consistent services. This shows that eventual consistency is quite a relaxed condition, and allows us to extend the classical CAP theorem constraints, since CAP's consistency not only encompasses the atomic model but also every consistency based on consensus.

Finally, four complementary aspects have been identified for implementing eventual consistency. In each aspect, several alternatives exist, and this means that there are many implementation strategies for developing eventually consistent services. A performance evaluation guide for assessing those strategies has been given, identifying the best combination of mechanisms for achieving some concrete application goals.

REFERENCES

- [1] AHAMAD, M.—BURNS, J. E.—HUTTO, P. W.—NEIGER, G.: Causal Memory. Proceedings of the 5th International Workshop on Distributed Algorithms and Graphs (WDAG '91), Delphi, Greece, 1991, pp. 9–30.
- [2] ALMEIDA, P. S.—SHOKER, A.—BAQUERO, C.: Efficient State-Based CRDTs by Delta-Mutation. In: Bouajjani, A., Fauconnier, H. (Eds.): Networked Systems (NETYS 2015). Springer, Cham, Lecture Notes in Computer Science, Vol. 9466, 2015, pp. 62–76.
- [3] ALSBERG, P.—DAY, J. D.: A Principle for Resilient Sharing of Distributed Resources. Proceedings of the 2nd International Conference on Software Engineering (ICSE '76), San Francisco, CA, USA, 1976, pp. 562–570.
- [4] APERS, P. M. G.—WIEDERHOLD, G.: Transaction Classification to Survive a Network Partition. Technical report, STAN-CS-85-1053, Department of Computer Science, Stanford University, Stanford, CA, USA, 1985.
- [5] ASPLUND, M.—NADJM-TEHRANI, S.—BEYER, S.—GALDÁMEZ, P.: Measuring Availability in Optimistic Partition-Tolerant Systems with Data Constraints. 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN '07), Edinburgh, UK, 2007, pp. 656–665, doi: 10.1109/DSN.2007.62.
- [6] ATTIYA, H.: Robust Simulation of Shared Memory: 20 Years After. Bulletin of the EATCS, Vol. 100, 2010, pp. 99–113.
- [7] BAILIS, P.—VENKATARAMAN, S.—FRANKLIN, M. J.—HELLERSTEIN, J. M.—STOICA, I.: Quantifying Eventual Consistency with PBS. The VLDB Journal, Vol. 23, 2014, No. 2, pp. 279–302.
- [8] BERNSTEIN, P. A.—DAS, S.: Rethinking Eventual Consistency. Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (SIGMOD '13), New York, NY, USA, 2013, pp. 923–928, doi: 10.1145/2463676.2465339.
- [9] BIRRELL, A. D.—LEVIN, R.—NEEDHAM, R. M.—SCHROEDER, M. D.: Grapevine: An Exercise in Distributed Computing. Communications of the ACM, Vol. 25, 1982, No. 4, pp. 260–274, doi: 10.1145/358468.358487.
- [10] BOUAJJANI, A.—Enea, C.—HAMZA, J.: Verifying Eventual Consistency of Optimistic Replication Systems. Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '14), San Diego, CA, USA, 2014, pp. 285–296, doi: 10.1145/2535838.2535877.
- [11] BUDHIRAJA, N.—MARZULLO, K.—SCHNEIDER, F. B.—TOUEG, S.: Optimal Primary-Backup Protocols. In: Segall, A., Zaks, S. (Eds.): Distributed Algorithms (WDAG '92). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 647, 1992, pp. 362–378.
- [12] BUNCH, S. R.: Automated Backup. In: Alsberg, P. (Ed.): Research in Network Data Management and Resource Sharing. Preliminary Research Study Report, CAC Document Number 162, University of Illinois at Urbana-Champaign, USA, 1975, pp. 71–106.
- [13] BURCKHARDT, S.: Principles of Eventual Consistency. Foundations and Trends in Programming Languages, Vol. 1, 2014, No. 1–2, pp. 1–150, doi: 10.1561/25000000011.

- [14] CAREY, M. J.—LIVNY, M.: Distributed Concurrency Control Performance: A Study of Algorithms, Distribution, and Replication. Proceedings of the 14th International Conference on Very Large Data Bases (VLDB '88), Los Angeles, CA, USA, 1988, pp. 13–25.
- [15] CHOCKLER, G. V.—KEIDAR, I.—VITENBERG, R.: Group Communication Specifications: A Comprehensive Study. *ACM Computing Surveys*, Vol. 33, 2001, No. 4, pp. 427–469, doi: 10.1145/503112.503113.
- [16] COSELL, B. P.—JOHNSON, P. R.—MALMAN, J. H.—SCHANTZ, R. E.—SUSSMAN, J.—THOMAS, R. H.—WALDEN, D. C.: An Operational System for Computer Resource Sharing. Proceedings of the 5th ACM Symposium on Operating Systems Principles (SOSP '75), Austin, Texas, USA, 1975, pp. 75–81, doi: 10.1145/800213.806524.
- [17] DAVIDSON, S. B.: Optimism and Consistency in Partitioned Distributed Database Systems. *ACM Transactions on Database Systems*, Vol. 9, 1984, No. 3, pp. 456–481, doi: 10.1145/1270.1499.
- [18] DÉFAGO, X.—SCHIPER, A.—URBÁN, P.: Total Order Broadcast and Multicast Algorithms: Taxonomy and Survey. *ACM Computing Surveys*, Vol. 36, 2004, No. 4, pp. 372–421, doi: 10.1145/1041680.1041682.
- [19] DE JUAN-MARÍN, R.—DECKER, H.—MUÑOZ-ESCOÍ, F. D.: Revisiting Hot Passive Replication. Proceedings of the 2nd International Conference on Availability, Reliability and Security (ARES '07), Vienna, Austria, 2007, pp. 93–102.
- [20] DE JUAN-MARÍN, R.—DECKER, H.—ARMENDÁRIZ-ÍÑIGO, J. E.—BERNABÉU-AUBÁN, J. M.—MUÑOZ-ESCOÍ, F. D.: Scalability Approaches for Causal Multicast: A Survey. *Computing*, Vol. 98, 2016, No. 9, pp. 923–947.
- [21] DEMERS, A. J.—GREENE, D. H.—HAUSER, C.—IRISH, W.—LARSON, J.—SHENKER, S.—STURGIS, H. E.—SWINEHART, D. C.—TERRY, D. B.: Epidemic Algorithms for Replicated Database Maintenance. Proceedings of the 6th Annual ACM Symposium on Principles of Distributed Computing (PODC '87), Vancouver, BC, Canada, 1987, pp. 1–12, doi: 10.1145/41840.41841.
- [22] FEKETE, A.—GUPTA, D.—LUCHANGCO, V.—LYNCH, N. A.—SHVARTSMAN, A. A.: Eventually-Serializable Data Services. Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing (PODC '96), Philadelphia, PA, USA, 1996, pp. 300–309, doi: 10.1145/248052.248113.
- [23] FEKETE, A. D.—RAMAMRITHAM, K.: Consistency Models for Replicated Data. In: Charron-Bost, B., Pedone, F., Schiper, A. (Eds.): *Replication: Theory and Practice*. Chapter 1. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 5959, 2010, pp. 1–17.
- [24] FISCHER, M. J.—LYNCH, N. A.—PATTERSON, M. S.: Impossibility of Distributed Consensus with One Faulty Process. *Journal of the ACM*, Vol. 32, 1985, No. 2, pp. 374–382, doi: 10.1145/3149.214121.
- [25] FISCHER, M. J.—MICHAEL, A.: Sacrificing Serializability to Attain High Availability of Data. Proceedings of the 1st ACM SIGACT-SIGMOD Symposium on Principles of Database Systems (PODS '82), Los Angeles, CA, USA, 1982, pp. 70–75, doi: 10.1145/588111.588124.

- [26] FOX, A.—BREWER, E. A.: Harvest, Yield and Scalable Tolerant Systems. Proceedings of The Seventh Workshop on Hot Topics in Operating Systems (HotOS '99), Rio Rico, Arizona, USA, 1999, pp. 174–178, doi: 10.1109/HOTOS.1999.798396.
- [27] GIFFORD, D. K.: Weighted Voting for Replicated Data. Proceedings of the 7th ACM Symposium on Operating Systems Principles (SOSP '79), Pacific Grove, CA, USA, 1979, pp. 150–162, doi: 10.1145/800215.806583.
- [28] GILBERT, S.—LYNCH, N.: Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services. ACM SIGACT News, Vol. 33, 2002, No. 2, pp. 51–59, doi: 10.1145/564585.564601.
- [29] GOLAB, W. M.—RAHMAN, M. R.—AUYOUNG, A.—KEETON, K.—GUPTA, I.: Client-Centric Benchmarking of Eventual Consistency for Cloud Storage Systems. 2014 34th IEEE International Conference on Distributed Computing Systems (ICDCS), Madrid, Spain, 2014, pp. 493–502.
- [30] GOODMAN, J. R.: Cache Consistency and Sequential Consistency. Technical report, No. 61, IEEE Scalable Coherent Interface Working Group, 1989.
- [31] HERBST, N. R.—KOUNEV, S.—REUSSNER, R. H.: Elasticity in Cloud Computing: What It Is, and What It Is Not. 10th International Conference on Autonomic Computing (ICAC), San Jose, CA, USA, 2013, pp. 23–27.
- [32] HERLIHY, M. P.—WING, J. M.: Linearizability: A Correctness Condition for Concurrent Objects. ACM Transactions on Programming Languages and Systems, Vol. 12, 1990, No. 3, pp. 463–492, doi: 10.1145/78969.78972.
- [33] HUTTO, P. W.—AHAMAD, M.: Slow Memory: Weakening Consistency to Enhance Concurrency in Distributed Shared Memories. Proceedings of the 10th IEEE International Conference on Distributed Computing Systems (ICDCS), Paris, France, 1990, pp. 302–309, doi: 10.1109/ICDCS.1990.89297.
- [34] JOHNSON, P. R.—THOMAS, R. H.: The Maintenance of Duplicate Databases. RFC 677, Network Working Group, Internet Engineering Task Force, 1975, doi: 10.17487/rfc677.
- [35] KAWELL JR., L.—BECKHARDT, S.—HALVORSEN, T.—OZZIE, R.—GREIF, I.: Replicated Document Management in a Group Communication System. Proceedings of the 1988 ACM Conference on Computer-Supported Cooperative Work (CSCW), Portland, Oregon, USA, 1988, p. 395, doi: 10.1145/62266.1024798.
- [36] KRISHNAKUMAR, N.—BERNSTEIN, A. J.: Bounded Ignorance in Replicated Systems. Proceedings of the 10th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS '91), Denver, Colorado, USA, 1991, pp. 63–74, doi: 10.1145/113413.113419.
- [37] KUMAR, A.—STONEBRAKER, M.: Semantics Based Transaction Management Techniques for Replicated Data. Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data (SIGMOD '88), Chicago, Illinois, USA, 1988, pp. 117–125, doi: 10.1145/50202.50215.
- [38] KUMAR, P.—SATYANARAYANAN, M.: Log-Based Directory Resolution in the Coda File System. Proceedings of the 2nd International Conference on Parallel and Distributed Information Systems (PDIS '93), San Diego, CA, USA, 1993, pp. 202–213, doi: 10.1109/PDIS.1993.253092.

- [39] LADIN, R.—LISKOV, B.—SHRIRA, L.: Lazy Replication: Exploiting the Semantics of Distributed Services. Proceedings of the 9th Annual ACM Symposium on Principles of Distributed Computing (PODC '90), Quebec City, Quebec, Canada, 1990, pp. 43–57, doi: 10.1145/93385.93399.
- [40] LAKSHMAN, A.—MALIK, P.: Cassandra: A Decentralized Structured Storage System. ACM SIGOPS Operating Systems Review, Vol. 44, 2010, No. 2, pp. 35–40, doi: 10.1145/1773912.1773922.
- [41] LAMPORT, L.: Time, Clocks, and the Ordering of Events in a Distributed System. Communications of the ACM, Vol. 21, 1978, No. 7, pp. 558–565, doi: 10.1145/359545.359563.
- [42] LAMPORT, L.: How to Make a Multiprocessor Computer That Correctly Executes Multiprocess Programs. IEEE Transactions on Computers, Vol. 28, 1979, No. 9, pp. 690–691, doi: 10.1109/TC.1979.1675439.
- [43] LAMPORT, L.: On Interprocess Communication. Distributed Computing, Vol. 1, 1986, No. 2, pp. 77–101.
- [44] LINDSAY, B. G.—SELINGER, P. G.—GALTIERI, C. A.—GRAY, J. N.—LORIE, R. A.—PRICE, T. G.—PUTZOLU, F.—TRAIGER, I. L.—WADE, B. W.: Notes on Distributed Databases. Technical report, RJ2571 (33471), IBM Research Laboratory, San Jose, CA, USA, 1979.
- [45] LIPTON, R. J.—SANDBERG, J. S.: PRAM: A Scalable Shared Memory. Technical report, CS-TR-180-88, Princeton University, USA, 1988.
- [46] MAEKAWA, M.: A \sqrt{N} Algorithm for Mutual Exclusion in Decentralized Systems. ACM Transactions on Computer Systems, Vol. 3, 1985, No. 2, pp. 145–159.
- [47] MOCKAPETRIS, P. V.: Domain Names – Concepts and Facilities. RFC 882, Network Working Group, Internet Engineering Task Force, 1983, doi: 10.17487/rfc0882.
- [48] MOCKAPETRIS, P. V.: Domain Names – Implementation and Specification. RFC 883, Network Working Group, Internet Engineering Task Force, 1983, doi: 10.17487/rfc0883.
- [49] MOSBERGER, D.: Memory Consistency Models. ACM SIGOPS Operating Systems Review, Vol. 27, 1993, No. 1, pp. 18–26, doi: 10.1145/160551.160553.
- [50] MUÑOZ-ESCOÍ, F. D.—BERNABÉU-AUBÁN, J. M.: A Survey on Elasticity Management in PaaS Systems. Computing, Vol. 99, 2017, No. 7, pp. 617–656.
- [51] O'NEIL, P. E.: The Escrow Transactional Method. ACM Transactions on Database Systems, Vol. 11, 1986, No. 4, pp. 405–430.
- [52] PARKER, D. S.—POPEK, G. J.—RUDISIN, G.—STOUGHTON, A.—WALKER, B. J.—WALTON, E.—CHOW, J. M.—EDWARDS, D. A.—KISER, S.—KLINE, C. S.: Detection of Mutual Inconsistency in Distributed Systems. In: Berkeley Workshop, 1981, pp. 172–184.
- [53] PASCUAL-MIRET, L.—GONZÁLEZ DE MENDÍVIL, J. R.—BERNABÉU-AUBÁN, J. M.—MUÑOZ-ESCOÍ, F. D.: Widening CAP Consistency. Technical report, IUMTI-SIDI-2015/003, Universitat Politècnica de València, Valencia, Spain, 2015.

- [54] PASCUAL-MIRET, L.—MUÑOZ-ESCOÍ, F. D.: Replica Divergence in Data-Centric Consistency Models. 2016 27th International Workshop on Database and Expert Systems Applications (DEXA Workshops), Porto, Portugal, 2016, pp. 109–112.
- [55] PU, C.—LEFF, A.: Replica Control in Distributed Systems: An Asynchronous Approach. Proceedings of the 1991 ACM SIGMOD International Conference on Management of Data (SIGMOD '91), Denver, Colorado, USA, 1991, pp. 377–386, doi: 10.1145/115790.115856.
- [56] SAITO, Y.—SHAPIRO, M.: Optimistic Replication. ACM Computing Surveys, Vol. 37, 2005, No. 1, pp. 42–81.
- [57] SARIN, S. K.—BLAUSTEIN, B. T.—KAUFMAN, C. W.: System Architecture for Partition-Tolerant Distributed Databases. IEEE Transactions on Computers, Vol. C-34, 1985, No. 12, pp. 1158–1163.
- [58] SATYANARAYANAN, M.—KISTLER, J. J.—KUMAR, P.—OKASAKI, M. E.—SIEGEL, E. H.—STEERE, D. C.: Coda: A Highly Available File System for a Distributed Workstation Environment. IEEE Transactions on Computers, Vol. 39, 1990, No. 4, pp. 447–459, doi: 10.1109/12.54838.
- [59] SHAPIRO, M.—PREGUIÇA, N. M.: Designing a Commutative Replicated Data Type. Technical report RR-6320, INRIA, Rocquencourt, France, 2007.
- [60] SHAPIRO, M.—PREGUIÇA, N. M.—BAQUERO, C.—ZAWIRSKI, M.: Convergent and Commutative Replicated Data Types. Bulletin of the EATCS, Vol. 104, 2011, pp. 67–88.
- [61] TANENBAUM, A. S.—VAN STEEN, M.: Distributed Systems – Principles and Paradigms. 2nd edition, Pearson Education, 2007.
- [62] TERRY, D. B.—DEMERS, A. J.—PETERSEN, K.—SPREITZER, M. J.—THEIMER, M. M.—WELCH, B. B.: Session Guarantees for Weakly Consistent Replicated Data. Proceedings of the 3rd International Conference on Parallel and Distributed Information Systems (PDIS '94), Austin, Texas, USA, 1994, Art. No. 19.
- [63] VIOTTI, P.—VUKOLIĆ, M.: Consistency in Non-Transactional Distributed Storage Systems. ACM Computing Surveys, Vol. 49, 2016, No. 1, Art. No. 19.
- [64] VOGELS, W.: Eventually Consistent. ACM Queue, Vol. 6, 2008, No. 6, pp. 14–19, doi: 10.1145/1466443.1466448.
- [65] WALKER, B. J.—POPEK, G. J.—ENGLISH, R.—KLINE, C. S.—THIEL, G.: The LOCUS Distributed Operating System. Proceedings of the 9th ACM Symposium on Operating Systems Principles (SOSP '83), Bretton Woods, New Hampshire, USA, 1983, pp. 49–70, doi: 10.1145/800217.806615.
- [66] YU, H.—VAHDAT, A.: Design and Evaluation of a Continuous Consistency Model for Replicated Services. Proceedings of the 4th Symposium on Operating Systems Design and Implementation (OSDI '00), San Diego, CA, USA, 2000, pp. 305–318.



Francesc D. MUÑOZ-ESCOÍ received his Ph.D. in computer science from Universitat Politècnica de València (UPV) in 2001. He currently works as Associate Professor at UPV. He has published more than 100 papers in international conferences and journals. His research interests cover multiple distributed system areas: group communication services, distributed algorithms, replication protocols, recovery approaches, distributed data management, elastic services and cloud computing.



José-Ramón GARCÍA-ESCRIVÁ received his final degree in computer science from UPV in 1987, where he currently works as Associate Professor. He has been involved as researcher in more than 20 projects. His interests cover both web technologies and distributed system areas.



Juan Salvador SENDRA-ROIG currently works as Associate Professor at UPV. His research interests cover string algorithms and several distributed system areas: distributed algorithms, replication protocols, distributed data management, and indexing and pre-processing of massive text data.



José M. BERNABÉU-AUBÁN received his Ph.D. in computer science from Georgia Institute of Technology (USA) and currently works as Full Professor at UPV where he leads the Distributed Systems Research Group. He has led the Instituto Universitario Mixto Tecnológico de Informática at UPV from 1994 to 2004 and since 2014 up to now. From 2004 to 2011, he joined Microsoft Corp. working actively in the architecture, design and development of the Windows Azure platform, co-authoring some of its patents. He has written multiple papers in journals and conferences in different distributed system areas. He has led more than 30 research projects in those fields.



José Ramón GONZÁLEZ DE MENDÍVIL received his Ph.D. in sciences from the University of the Basque Country (Spain) in 1993 and currently works as Full Professor at Universidad Pública de Navarra where he leads the Distributed Systems research group. He has written several papers in journals and conferences in different distributed system areas: distributed algorithms, deadlock detection, deadlock resolution, replicated databases, and replication protocols. His current interest is designing elastic services for PaaS using fuzzy performance models.

A NEUROGENETIC ALGORITHM BASED ON RATIONAL AGENTS

Lídio Mauro Lima DE CAMPOS

Faculty of Computing/ICEN

*Universidade Federal do Pará, Rua Augusto Corrêa 01, Guamá
CEP 66075-110, Caixa postal 479, Belém, Pará, Brasil*

e-mail: limadecampos@gmail.com

Roberto Célio Limão DE OLIVEIRA

Faculty of Computer Engineering

*Universidade Federal do Pará, Rua Augusto Corrêa 01, Guamá
CEP 66075-110, Caixa postal 479, Belém, Pará, Brasil*

e-mail: limao@ufpa.br

Gustavo Augusto Lima DE CAMPOS

Faculty of Computing

*Universidade Estadual do Ceará, Centro de Ciências e Tecnologia
Av. Paranjana 1700, Itaperi, CEP 60740-000, Fortaleza, Ceará, Brasil*

e-mail: gustavo@larces.uece.br

Abstract. Lately, a lot of research has been conducted on the automatic design of artificial neural networks (ADANNs) using evolutionary algorithms, in the so-called neuro-evolutive algorithms (NEAs). Many of the presented proposals are not biologically inspired and are not able to generate modular, hierarchical and recurrent neural structures, such as those often found in living beings capable of solving intricate survival problems. Bearing in mind the idea that a nervous system's design and organization is a constructive process carried out by genetic information encoded in DNA, this paper proposes a biologically inspired NEA that evolves ANNs using these ideas as computational design techniques. In order to do this,

we propose a Lindenmayer System with memory that implements the principles of organization, modularity, repetition (multiple use of the same sub-structure), hierarchy (recursive composition of sub-structures), minimizing the scalability problem of other methods. In our method, the basic neural codification is integrated to a genetic algorithm (GA) that implements the constructive approach found in the evolutionary process, making it closest to biological processes. Thus, the proposed method is a decision-making (DM) process, the fitness function of the NEA rewards economical artificial neural networks (ANNs) that are easily implemented. In other words, the penalty approach implemented through the fitness function (Equation (5)) automatically rewards the economical ANNs with stronger generalization and extrapolation capacities. Our method was initially tested on a simple, but non-trivial, XOR problem. We also submit our method to two other problems of increasing complexity: time series prediction that represents consumer price index and prediction of the effect of a new drug on breast cancer. In most cases, our NEA outperformed the other methods, delivering the most accurate classification. These superior results are attributed to the improved effectiveness and efficiency of NEA in the decision-making process. The result is an optimized neural network architecture for solving classification problems.

Keywords: Evolutionary computation, neural networks, grammatical evolution, hybrid intelligent systems

1 INTRODUCTION

The design of an Artificial Neural Network (ANN) can be considered a complex decision making process that frequently relies on the user experience. In general, this ANN design task is a trial and error process, where a number of different transfer functions and amount of hidden neurons should be adjusted in order to solve a specific problem. As the design of ANN is still being done, manually, by human experts, the automation of this design process will benefit the decision-making process done by human experts [2].

Bio-inspired algorithms have shown efficient in different nonlinear optimization problems [3, 5, 13, 44, 30]. Due to their efficiency and adaptability, the interest in research in the field of neuroevolution (i.e. evolutionary approach to design ANNs) has increased lately. The core issue in neuroevolution is to build an efficient indirect encoding scheme [6, 7, 8, 9, 10]. Which means that the evolutionary algorithm evolves a compressed description of the ANN rather than the ANN itself. [8].

These approaches are biologically motivated by the fact, that in case of the human brain, there are more neurons than nucleotides in the genome. Moreover, in biological genetic encoding the mapping between genotype and phenotype is indirect. The phenotype typically contains orders of magnitude more structural components than the genotype contains genes. Nevertheless, neuroevolution has not so far addressed that the development and evolution of neurons is carried out by genetic

information encoded in DNA and when followed will generate the final shape of the organs including the brain.

ADEANN is inspired by two natural biological mechanisms: genetic encoding and the evolution of genetic coding. As is well-known, neuron development is governed by the genetic information encoded in deoxyribonucleic acid (DNA), and ultimately generates the final shape of the brain. During biological development, in the complex process that links the DNA code to its phenotype, the same gene is used in many contexts. This compactness minimizes the information required to describe complex individuals. On the other hand, evolution describes the temporal changes in the genetic code (DNA). Among the several mechanisms underlying these evolutionary changes, natural selection is very important.

The two natural processes described above are hybridized such that the DNA contained in cells can also spawn cells. On the other hand, the changes in DNA are passed onto later generations. Motivated by these natural processes, we propose an artificial hybrid system that abstracts these natural mechanisms at an acceptable level of complexity.

To this end, we propose a new NEA, a biologically inspired artificial hybrid system [45, 46, 47, 49, 50] called Artificial Development and Evolution of ANNs (ADEANN). The ADEANN integrates two components. The first is a generative representation that represents genotypes (a set of production rules of a Lindenmayer system) by a compact indirect encoding scheme (IES). The IES also conducts and controls the process of mapping the genotypes to the phenotypes (complex neural morphologies). To mimic the DNA encoding scheme and enable scalability, our IES leverages the phenotype representation to a smaller genotype. Thus, the search process is carried out in a lower-dimensional solution space. In addition, our IES implements the organizational principles of hierarchy, modularity, and gene reuse (allowing compact representation of complex phenotypes). The second component is a genetic algorithm (GA), a simplified representation of natural evolution. In local search problems based on GAs, a bit string is called a chromosome (the genotype). Each bit on the chromosome is a gene, and a gene set represents the parameters of a function to be optimized. Each string is assigned a fitness that indicates the quality of its encoded solution (the phenotype). To improve the biological realism of GA, the GA in our approach evolves the generative representation. The evolutionary process can be regarded as the temporal genetic changes in the hypothetical DNAs of a population of individuals, regulated by an artificial selection mechanism. The above biological inspiration underlies the originality of our approach. To our knowledge, we report the first attempt to generate recurrent neural networks from combined metaphors.

The main contribution of our method is the genotype representation by our proposed IES. Using a compact DNA encoding, we codify a parametric Lindenmayer system (L-system) with memory, which implements the principles of organization, modularity, repetition, and hierarchy to achieve complex neural architectures (multi-layer and recurrent networks). [34, 35] adopted L-systems, although [35], used DNA encoding, their study was restricted to feedforward neural networks, whereas our

approach is extended to recurrent networks. In the IES used by [34], the genotypes encode twenty rewrite rules of an L-system. Our DNA encoding system encodes a parametric L-system with memory using 10 production rules. Therefore, our IES is more compact than [34]'s method, and reduces the search space of all feasible solutions. In addition, the memory mechanism in our approach enables the reuse of phenotypic structures (rewrite of nodes and connections) at different stages of development. Such reuse is an important capability of NEAs.

Our ADEANN also utilizes expert knowledge of the problem to more efficiently search the infinite space of topologies, thus minimizing the expert's effort in the optimization. The penalty approach implemented by the fitness function (Equation (5)) automatically rewards the economical ANNs with stronger generalization and extrapolation capacities. Our L-system generates ANN topologies without requiring additional substrate configurations for the given problem.

This paper is organized as follows. Section 2 discusses the state of the art. In Section 3 we describe a new approach to formalize the problem of ADANNs (artificial development and evolution of ANNs) as a local search based on rational agents. Section 4 introduces a biologically inspired method for automatic design of ANNs. In Section 5, the experimental setup and results are presented. Lastly, a discussion and conclusions are presented in Sections 6 and 7, respectively.

2 STATE OF THE ART

This section discusses existing research on NEAs. Most NEAs use direct encoding systems (DESSs) [12], which specify every connection and node in the genotype that will appear in the phenotype (ANN). These methods are simply implemented, but the size of the connectivity matrix scales as the square of the number of nodes. In NEAT [6], the DES incorporates a few biologically plausible entities, and alters both the weighting parameters and structures of the networks. [29] developed a novel multi-objective optimization for a hierarchical genetic algorithm (MOHGA) based on the micro-GA approach. However, this method was not tested in other applications such as temporal series forecasting.

As discussed by [1], the increasing complexity of evolutionary computation demands more sophisticated methods than direct mapping from genotype to phenotype. At the other extreme are indirect encoding systems (IESs) [7, 28, 27, 8, 9, 10, 16, 30]. In an IES, the network generation is indirectly specified by the genotype. The solution description is compressed, enabling complex topologies of ANNs.

ES-HyperNEAT [10, 19] has shown promising results enabling the development of complex regular plastic ANNs. The encoding scheme adopted by HyperNEAT [8, 20] and ES-HyperNEAT [10] does not shape itself to the biological development process, including the nervous system. The CPPN (Compositional pattern-producing networks) in HyperNEAT plays the role of DNA in nature, but at a much higher level of abstraction; in effect, it encodes a pattern of weights that is painted across the geometry of a network. Furthermore, in these methods both

the genotype and phenotype are neural networks, which is not biologically plausible.

The ES-HyperNEAT [10] is an improvement over HyperNEAT [8]. While the location of the hidden nodes in the substrate had to be decided by the designer on the original HyperNEAT, ES-HyperNEAT showed that the decision might in fact be automated. In the HyperNEAT, CPPN encodes an infinite number of weights within the Hypercube, from which a subset should be chosen to be incorporated into the ANN. [10] consider other important information is the density of nodes from which an additional increase of the same does not offer any advantage. For more details about state of the art, query the authors' research [21].

The following methods [7, 9, 30], belong to the class of GDSs (generative and developmental systems) using grammatical evolution (GE). The methodologies took a step towards of biologically inspired approaches. Lee et al. [9] sought inspiration from the DNA, in their research, information is coded using the symbols A, G, T and C. A sequence of three of these symbols is known as a codon. The sequence of codons between these delimiters is translated into a production rule for developing a neural controller. The production rule of the L-System used by [9] is context-free and does not allow to generate recurrent networks. A similar idea [7] using binary strings also exists in which the process of reading and translating bits can be repeated from different bits in the string to produce different production rules. The ANE proposed by [7] has one disadvantage that is the difficulty to set the parameters of the fitness function of Genetic Algorithm to direct the search for minimum architectures.

The data structure of NEAT [6], which represents the genotype, grows linearly with the number of edges between two nodes. (MOHGA) [29] cannot yield the recurrent neural networks (RNNs). Despite the novel capabilities of HyperNEAT, the user must manually place the hidden nodes at the substrate, which substantially disadvantages the original HyperNEAT. The applicability of the method proposed by [30] to other applications, such as temporal series-forecasting, was not tested, and the model cannot yield RNNs.

3 OUTLINE OF THE APPROACH

Our approach involves the formulation of an artificial neural network design problem (ANNNDP) as an optimization problem, that is: given a set of L observations on the behavior of a particular process, $\Psi = \{(xd^l, yd^l)\}$, $l = 1, \dots, L$, where xd^l represents a numeric vector defined in R^n and yd^l is a numeric vector defined in R^m , the task is to find a back-propagation ANN's topology, $yc^l = \text{ANN}(w^*, xd^l)$, which minimizes the mean square error between yd^l and yc^l , this is, between the desired values in the observations set and the computed values in the neurons' outputs situated in the ANN's output layer.

An ANN's topology can be described as a finite set of neurons, that is, nodes in graphs notation $\text{Nodes} = \{n_1, n_2, \dots, n_k\}$, and a finite set $H \subseteq N \times N$ of connections between neurons, meaning, directed edges in graphs notation. From the point of

view of graph theory, feed-forward ANNs (FANN) are acyclic graphs while recurrent ANNs (RANN) are cyclic graphs. An input layer is a set of input units, that is, a subset of n nodes whereas an output layer is a set of output units, namely a subset of m nodes. In FANNs, the k^{th} layer ($k > 1$) is the set of all nodes $n_i \in \text{Nodes}$. This type of nodes have an edge path of length $k - 1$ between some input unit and u . In fully connected RANNs, all units have connections to all non-input units.

We approach the ANNDP solving based on the problem-solving-agent proposed by Russell and Norvig [23], whose agent is called ADEANN (artificial development and evolution of ANNs), which encapsulates a special scheme of solutions representation as well as a local search strategy based on genetic algorithms to solve the problem. Regarding the representation scheme, the approach adopts a generative representation, that is, instead of an encoded ANN topology, each chromosome stores a set of production rules of a Lindenmayer system, which, in turn, generates ANNs topologies. With regard to the solution process, the SEARCH-ANN function outlined below illustrates the structure of the program in the ADEANN agent.

Function SEARCH-ANN(SearchParameters, TransitionModel,
FitnessFunction) **return** an ANN topology

```

1: inputs: SearchParameters, TransitionModel, FitnessFunction
2: vars: Pop, t, PopPerformances;
3:
4:  $k \leftarrow 0$ 
5: ANNsPopk  $\leftarrow$  Generate-ANNs(SearchParameters)
6: ANNsPerformance  $\leftarrow$  Evaluate-ANNs(ANNsPopk, FitnessFunction)
7: loop do
8: if StopConditionTest( $k$ , ANNsPerformance, SearchParameters)
9: then return solution(Best-ANN(ANNsPopk, ANNsPerformance)
10: ANNsPop(k+1)  $\leftarrow$  (ANNsPop(k+1), ANNsPerformance, TransitionModel,
SearchParameters)
11: ANNsPerformance  $\leftarrow$  Evaluate-ANNs(ANNsPop(k+1), FitnessFunction)
12:  $k \leftarrow k + 1$ 
13: end

```

The SEARCH-ANN function starts the local search process aiming to achieve an artificial neural network topology $yc^l = \text{ANN}\{(w^*, xd^l)\}$, which minimizes the mean square error between yd^l and yc^l , for $l = 1, \dots, L$ in the ANNDP's formulation. This function employs information on the search parameters (SearchParameters input term) as well as a transition model (TransitionModel input term) to describe how to modify current populations of ANNs and generate a new population, in addition to an evaluation function (FitnessFunction input term) to measure the value of each ANN in a current population.

Firstly, in the beginning of the process, Generate-ANNs function generates an initial population of ANNs, where each ANN is represented by a set of production rules of a Lindenmayer system codified in a chromosome. This function considers the information in the SearchParameters input term on the desired number of ANNs in the populations as well as on the desired length for the chromosomes in the population. Evaluate-ANNs function stores in ANNsPerformace the computed performance value of each ANN topology in the current population based on the mean square error computed in the output layer of the ANN. SEARCH-ANN function employs an iteration counter (k) and a condition named StopConditionTest Boolean function to decide when to stop the local search process and return a solution to a problem. Stop condition is described by means of a proposition relating the information on the current iteration counter k and the information available in SearchParameters input term, that is, regarding the max number of loops in its repetition scheme, which is central to the local search strategy in the approach, as well as on an ideal performance value such that for an ANN to be considered a solution.

Modify-ANN function is executed repeatedly seeking to transform a current population of ANN's in a new population of ANNs. In our approach, this function encapsulates the evolutionary principles of pairs selection and crossing over pairs and individual mutation. Central to the approach, compact indirect encoding scheme (IES) conducts and controls the process of mapping a set of production rules of a Lindenmayer system codified in a chromosome to an associated ANN topology.

It is worth mentioning that the local search strategy in the SEARCH-ANN function is an adaptation of the generate-and-test programming technique in order to find the best ANN topology. More specifically, in this case, the strategy can be said to consisting in an adaptation of the modify-test programming technique, such that the goal is to find the best ANN, according to the values of FitnessFunction, as well as that the next solution set is the transformation of a given current solution set.

4 BIOLOGICALLY INSPIRED NEA

The general structure of ADEANN is shown in Figure 1. The optimization process of ADEANN proceeds through several stages. The GA starts with a population of individuals randomly initialized with 0s and 1s (Figure 1 a)). Second, the bits of each individual of the population are subjected to transcription (Figure 1 b)) and translation (Figure 1 c)), following valid production rules. Both processes are performed by the function Rule-Extraction-with-GA, presented in Section 4.2. After finding the appropriate production rules (Table 1), the rewriting system generates the genotypes (Figure 1 d)). All of the genotypes are mapped to phenotypes (ANN architectures) (Figure 1 e)). The ANNs are then trained (Figure 1 f)) and validated, and tests are carried out. The classification accuracy of each ANN is measured from its fitness (Figure 1 g)), calculated by Equation (4) or Equation (5). The latter equation implements a penalty approach that rewards economical ANNs with

stronger generalization capacity and greater ease of implementation. The genotypes are classified by the performances of their ANNs (Figure 1h)). The GA selects the best individuals (Figure 1i)) for mutation and crossover operations (Figure 1j)), which provide the new population (Figure 1k)). The previous steps are repeated through n generations.

4.1 Neural Structure Codification with L-System

To mimic the mechanism of grown structures, including neurons, we adopt a parametric L-system with memory. It comprises a set of rules created from an alphabet. This system can be described as a grammar $G = \{\Sigma, \Pi, \alpha\}$, where the alphabet consists of the elements of the set $\Sigma = \{., f, F, n, [,], *, B\}$, and the production rules (Π) described in Table 1. The axiom $\alpha = .$ is the starting point of the developmental process, where f denotes a neuron and F is a connection between neurons, $[$ and $]$ indicate storage and recovery, respectively, of the current state of the development, $*$ denotes that the string is recovered from storage, and B is the connection of a neuron with a block of neurons. The second rule $. \rightarrow (f \dots f)n$ means replace the start point by the neurons of the input layer. Rule 3.1 ($f \rightarrow [f$) means to store the position of the current neuron, so as to start a new ramification from it. Rule 3.2 ($f \rightarrow fFf$) means establish a connection between two neurons. Rule 3.3 ($f \rightarrow fF$) means establishing a connection from a specific neuron. Rule 3.4 ($f \rightarrow n$) means replace a provisional neuron with a permanent neuron. Rule 3.5 ($f \rightarrow f$) means to maintain a specific neuron during development. Rule 3.6 ($f \rightarrow fB$) means connect a neuron to a block of neurons. Rule 4 ($[\rightarrow [Ff$) means start the development of a new ramification from a specific neuron and recover the previous state. Rule 5 ($f \rightarrow f*$) means recover a previous ramification stored for use.

Our approach is biologically motivated by the human brain, which contains many more neurons than nucleotides in the genome [4].

To paraphrase [4], part of the genetic code (genetic information) is converted into the nervous system during biological development. The genetic information drives the cell division, enabling to encode highly complex systems with a compact code. For example, a human brain contains approximately 10^{11} neurons, each connected to 10^5 other neurons (on average). If the connectivity graph was encoded by a destination list for each neuron, it would consume $10^{11} * 10^5 * \ln_2 10^{11} = 1.7 * 10^{17}$ bits of storage, whereas the number of genes in the human genome is of the order of $2 * 10^9$. [4] mentioned that the two numbers differ by more than 10 orders of magnitude, and marveled at the degree of compression achieved by the neuron developmental process.

As a simple example, Figure 2 illustrates the construction process of one branch of an ANN. Suppose that starting with the axiom ($\alpha = .$) and applying the second production rule $. \rightarrow f$ to the axiom, the resulting string is f . Applying the third rule (3.1) $f \rightarrow [f$ to string f yields a new string, $[f$. After one, two, and three applications of the fourth rule $[\rightarrow [Ff$ on string $[f$, the strings become $[Ff]f$, $[Ff]Ff]f$, and $[Ff]Ff]Ff]f$, respectively. The last string, which is stored for later

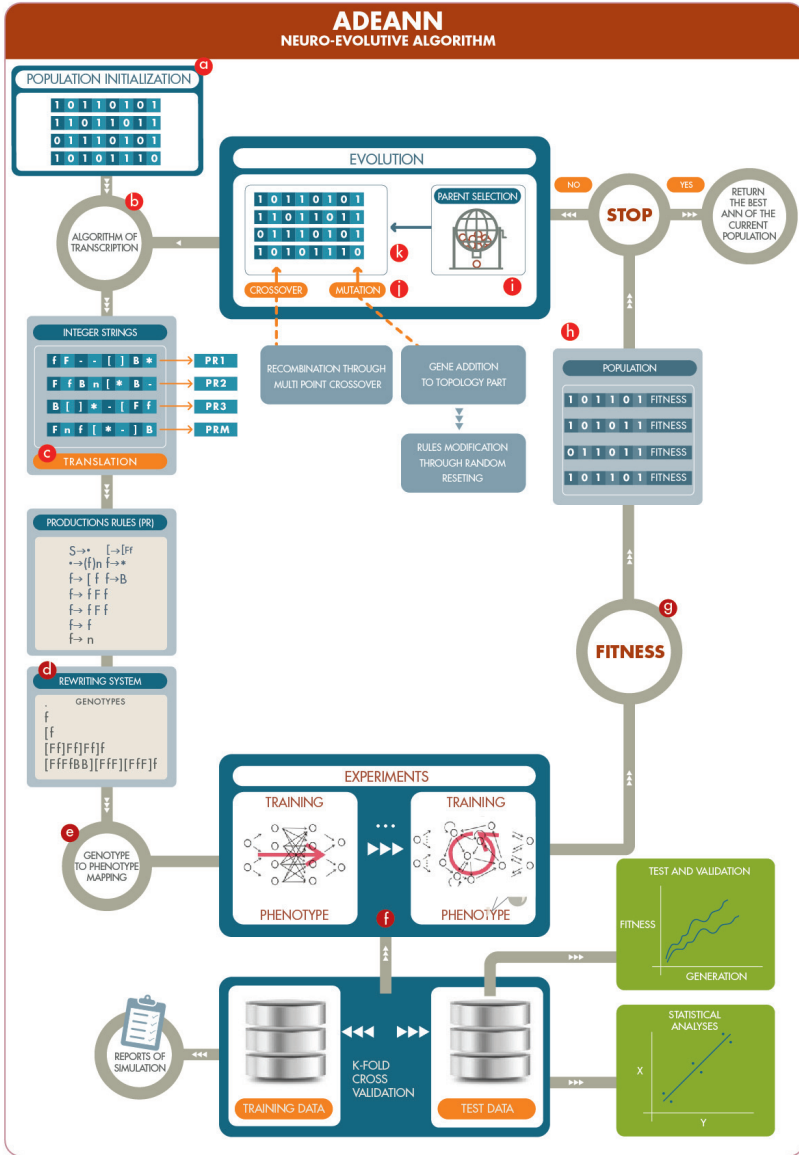


Figure 1. ADEANN links three computational paradigms: GA, an L-system, and an ANN. The evolutionary processes are defined as temporal genetic changes in the hypothetical DNAs, regulated by a selection mechanism. The hypothetical DNA of each individual encodes a set of production rules of an L-system, which coordinates the growth of artificial neurons. Each ANN is trained to solve a given problem and is thereafter tested to check its generalization or prediction capability.

Rule Identifier	Rule
1, 2	$S \rightarrow \cdot$ (axiom) (2) $\cdot \rightarrow (f \dots f)n$
3	(3.1) $f \rightarrow [f$ (3.2) $f \rightarrow fFf$ (3.3) $f \rightarrow fF$ (3.4) $f \rightarrow n$
3, 4, 5	(3.5) $f \rightarrow f$ (3.6) $f \rightarrow fB$ (4) $[\rightarrow [Ff]$ (5) $f \rightarrow f*$

Table 1. The production rules of the parametric L-System with memory

use, represents a branch with three neurons (see last row, third column of Figure 2). Rule (5), $f \rightarrow f*$ (which means $f \rightarrow f[Ff]Ff[Ff]$, where $*$ denotes that the string is recovered from storage), is applied to the first f of the previous string $[Ff]Ff[Ff]f$. The resulting string is $[Ff[Ff]Ff[Ff]]Ff[Ff]f$, representing the phenotype shown in the fourth row and third column of Figure 2. This phenotype begins a new branch from N3. Here, we have used the principle of *reuse phenotypic structure*, which produces a variation of an existing structural theme.

Applying Rule (3.2) to the second f of the previous string, the resulting string is $[Ff[FfFf]Ff[Ff]]Ff[Ff]f$, with phenotype shown in the third row and third column of Figure 2. This phenotype creates a new neuron N8. The development continues by recursively iterating Rules (5) and (3.2). First, Rule (5) is applied to the sixth f of the previous string $[Ff[FfFf]Ff[Ff]]Ff[Ff]f$, yielding $[Ff[FfFf]Ff[Ff]]Ff[Ff[Ff]]Ff[Ff]f$. Consequently, a new branch begins from N4. Here, we have used the second principle of *reuse*, which creates separate developmental pathways from the same gene. Second, Rule (3.2) is applied to the seventh f of the previous string, yielding $[Ff[FfFf]Ff[Ff]]Ff[FfFf]Ff[Ff]f$. This step creates a synapse from N41 to N8. Figure 2 also shows the mapping between genotypes and phenotypes. Our indirect encoding method, presented in Section 4.2, generates ANNs with variable numbers of neurons in the range $[X, Y]$, where X and Y are computed by Equations (1) and (2), respectively. The classes of admissible ANNs generated by the L-Systems are direct and recurrent neural networks, with the recurrent output layer. The encoding rule is described by the rules presented in Table 1.

$$X = [N_{inputs} + N_{inputs} * random(NR_{min}) + N_{outputs}], \tag{1}$$

$$Y = [N_{inputs} + N_{inputs} * random(NR_{max}) + N_{outputs}]. \tag{2}$$

Here, N_{inputs} and $N_{outputs}$ are the numbers of neurons in the input and output layers of the ANN, respectively, and NR is the number of neurons at each ramification.

In a dismembered way, Figure 3 a) illustrates the iterative generation of ANN at each stage of the development process. The algorithm starts by interactively finding the placements of the hidden neurons from the inputs and their connections. It then determines the placements of the other hidden neurons and their connections. Finally, it determines the placements of the output neurons from the hidden neurons and their connections. The ANN construction process is illustrated in Figure 3 b). The entire developmental process follows an important biological property called the

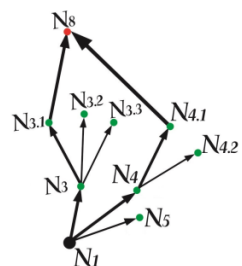
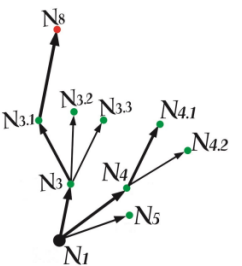
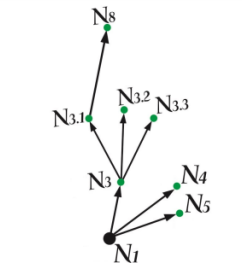
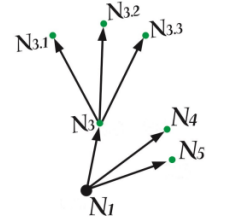
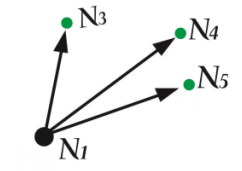
RULES	GENOTYPE	PHENOTYPE	IT
R3.2	$\begin{array}{cccccccccccc} \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ [Ff & [Ff & Ff] & Ff] & Ff] & Ff] & [Ff & F] & Ff] & Ff] & f \\ E_1 & E_1 & E_3 & & E_3 & E_3E_1 & E_1 & E_1 & E_1E_1 & E_1 & E_1 & E_1 \end{array}$		IT.10
R5	$\begin{array}{cccccccccccc} \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ [Ff & [Ff & Ff] & Ff] & Ff] & Ff] & [Ff & F] & Ff] & Ff] & f \\ E_1 & E_1 & E_1 & E_1 & E_1E_1 & E_1 & E_1 & E_1 & E_1E_1 & E_1 & E_1 & E_1 \end{array}$		IT.9
R3.2	$\begin{array}{cccccccccccc} \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ [Ff & [Ff & Ff] & Ff] & Ff] & Ff] & [Ff & F] & Ff] & Ff] & f \\ E_1 & E_1 & E_3 & E_3 & E_3E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 \end{array}$		IT.8
R5	$\begin{array}{cccccccccccc} \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ [Ff & [Ff & Ff] & Ff] & Ff] & Ff] & [Ff & F] & Ff] & Ff] & f \\ E_1 & E_1 & E_3 & E_3 & E_3E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 \end{array}$		IT.7
R4(3x) R3.1 R2 axiom	$\begin{array}{cccccccccccc} \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ [Ff & [Ff & Ff] & Ff] & Ff] & Ff] & [Ff & F] & Ff] & Ff] & f \\ E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 & E_1 \end{array}$		IT.6

Figure 2. A simple example of the construction process of a branch of an iterated ANN using the rules of the L-system is illustrated in Table 1

hierarchical principle (recursive composition of sub-structures). In the topologies generated at each developmental stage, only those nodes forming paths to an input and output neuron are retained. The search through the search space is restricted to functional ANN topologies.

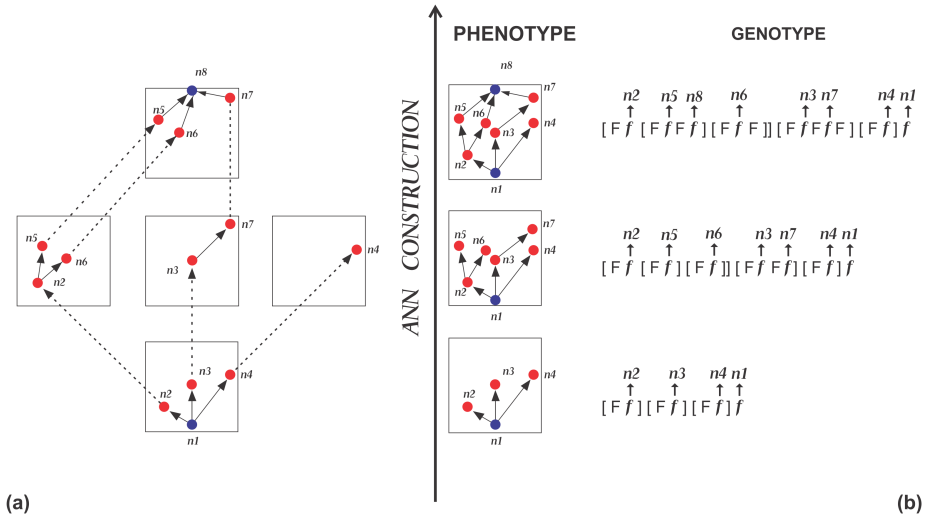


Figure 3. Construction of an iterated network

4.2 Rule Extraction by Genetic Algorithms

This subsection is dedicated to rule extraction by GAs. The neurons generated in the previous subsection are developed after the following process. To formulate a biologically realistic GA, we let the genes of the chromosomes (sequences of hypothetical DNA) encode a recipe (the production rules of the L-system described in Section 4.1 and illustrated in Table 1). The recursive rules in Table 1 drive the developmental stages of the neurons (Figure 2).

[40] argued that biological genes are meaningful only when translated into proteins following certain rules of growth for organ development. He emphasized that the complete genetic formula of neuronal development is unknown to geneticists.

In biological genetic processing (Figure 4 b)), DNA is transcribed into ribonucleic acid (RNA), and the RNA is translated into proteins. The proteins are derived from linear sequences of amino acids encoded by codons (groups of three nucleotides selected among U, C, A, and G) of the genetic code (Table 2). Table 2 specifies the translation products of different codons; for example, the codon UUU is translated into phenylalanine (Phe), and UUA is translated into leucine (Leu). In Figure 4 b), the protein is formed by a sequence of amino acids starting with methionine (Met)

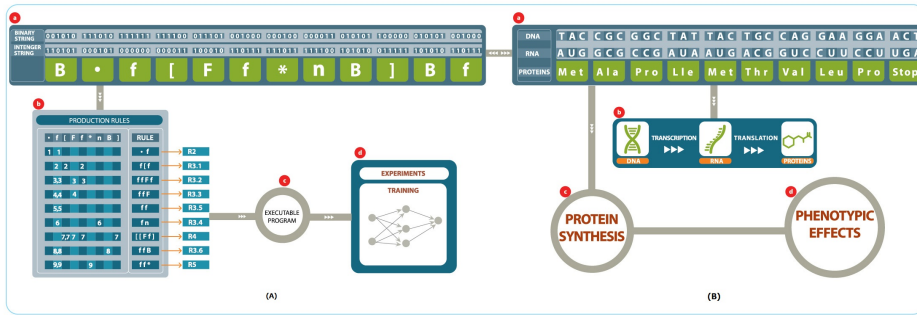


Figure 4. b) DNA transcription into RNA and translation of RNA into protein. a) In the analogous artificial process, a binary string is transcribed into an integer string and the string is translated into the production rules of the L-system.

and ending with proline (Pro). Such protein synthesis triggers all stages of the neuronal development (phenotypic effects), as shown in Figure 4 b).

The elements of the alphabet $\Sigma = \{., f, F, n, [, *, B, \}$ of the L-system, described in Section 4.1 and displayed in bold font in Table 2, are a metaphor of the genetic code. Each two-bit sequence represents one nucleotide; for example, the set (00, 01, 10, 11) symbolizes (U, C, A, G) in the original genetic code. Accordingly, six bits represent three nucleotides; that is, (000000, 011111) symbolizes (UUU, CGG). Below we present the function Rule-Extraction-with-GA, which attempts to mimic DNA transcription and RNA translation as discussed in the previous paragraph.

Steps 1 and 2 of the function Rule-extraction-with-GA mimic the DNA transcription process into RNA, as shown in Figure 4 b). These steps are repeated for all individuals in the population. All the integer strings obtained after step 4 of this process see Figure 4 a) are stored in the array 'D' and evaluated in step 5. Step 5 translates the integer string to valid production rules of the L-system (Table 1) proposed in Section 4.1. Steps 5, 5.1 and 5.2 mimic the translation process of RNA into protein. The above steps are repeated for all rows in table 'D'.

Figure 4 a) illustrates the rule extraction for a single individual of the population. In this figure, the transcription string yields the integer string B.f[Ff.nB]Bf. We seek the shortest string containing all valid rules; in this case, .f[Ff * nB]. Steps 5, 5.1, and 5.2 identify the minimum substring .f[Ff * nB] within the larger string B.f[Ff.nB]Bf. After finding the minimum string, we identify the positions at which the rules were found (see Figure 4 a)). For example, Rule 2 ($\cdot \rightarrow f$), symbolically represented by (\cdot, f), is found at positions 1 and 2 of the string .f[Ff * nB]. Rule 3.1 $f \rightarrow [f$ is found at positions 1 and 2, and 3 and 5.

Collectively, these rules form a kind of executable program that prepares the artificial neurons for development. Once a set of valid production rules has been determined for each line of table 'D', the neuronal development presented in Section 4.1 is initiated. The development process is illustrated in Figure 2. The resulting

	00 (U)	01 (C)	10 (G)	11 (A)	
00 (U)	<i>f</i> (UUU)	<i>F</i> (UCU)	<i>n</i> (UAU)	<i>.</i> (UGU)	00 (U)
00 (U)	<i>n</i> (UUC)	<i>.</i> (UCC)	<i>f</i> (UAC)	<i>F</i> (UGC)	01 (C)
00 (U)	<i>F</i> (UUA)	<i>f</i> (UCA)	<i>B</i> (UAA)	<i>f</i> (UGA)	10 (A)
00 (U)	[(UUG)	<i>n</i> (UCG)	[(UAG)	* (UGG)	11 (G)
01 (C)	<i>f</i> (CUU)] (CCU)	<i>n</i> (CAU)	* (CGU)	00 (U)
01 (C)	* (CUC)	<i>F</i> (CCC)	<i>f</i> (CAC)	<i>F</i> (CGC)	01 (C)
01 (C)] (CUA)	<i>f</i> (CCA)	* (CAA)	[(CGA)	10 (A)
01 (C)	<i>f</i> (CUG)	* (CCG)	<i>B</i> (CAG)] (CGG)	11 (G)
10 (A)	* (AUU)] (ACU)	<i>n</i> (AAU)	<i>f</i> (AGU)	00 (U)
10 (A)	<i>f</i> (AUC)	<i>B</i> (ACC)	<i>f</i> (AAC)	<i>B</i> (AGC)	01 (C)
10 (A)	<i>F</i> (AUA)	[(ACA)	<i>B</i> (AAA)	<i>n</i> (AGA)	10 (A)
10 (A)	* (AUG)	<i>f</i> (ACG)	* (AAG)] (AGG)	11 (G)
11 (G)] (GUU)	[(GCU)	<i>F</i> (GAU)	<i>n</i> (GGU)	00 (U)
11 (G)	<i>n</i> (GUC)	<i>B</i> (GCC)	[(GAC)	<i>.</i> (GGC)	01 (C)
11 (G)	<i>f</i> (GUA)] (CGA)	<i>B</i> (GAA)	<i>F</i> (GGA)	10 (A)
11 (G)	<i>B</i> (GUG)	<i>f</i> (GCG)	* (GAG)	[(GGG)	11 (G)

Table 2. The genetic code from the perspective of mRNA, translated as in Figure 4b). In the same table, the DNA's metaphor.

population of ANNs is trained and the fitness of each ANN is calculated by Equation (5). The ordered sequences of alphabetical characters (valid production rule: $.f[Ff * nB]$) are analogous to the sequences of amino acids that join into a protein (Figure 4b)). Protein synthesis triggers all stages of biological development. As the function Rule-Extraction-with-GA begins the string reading from different positions and in both directions, the implicit parallelism level of the GA increases, alleviating the scalability problem.

4.3 Fitness Evaluation and Selection Mechanism

The fitness of an ANN can be appropriately measured by the mean square error (MSE) given by Equation (3). The MSE measures the expected squared distance between the predicted and true values. As such, it measures the quality of a predictor. The system adjusts the weights of its neural networks to minimize the MSE in the training and test sets. ADEANN aims not only to minimize the MSE of the ANN in the training set but also to generalize and properly predict in the test set. In Equation (3), q is the number of test samples. The fitness function selects the most suitable neural networks (direct or recurrent) for a specific class of problem, taking into consideration, that dynamical systems are more precisely estimated by recurrent neural networks (RNNs). In other words, depending on the simulated problem class, the fitness function, selects a direct or recurrent ANN most suitable for a specific task. The fitness function Equation (5) automatically implements a penalty approach that rewards economical ANNs with better generalization capacities and

ease of implementation (see the last fifteen lines and second column of Table 5). This function selects networks by two criteria; the number of hidden-layer neurons and the MSE. Therefore, smaller networks score higher fitness values than larger networks with the same performance. RNN-evolving NEAs are rarely reported in the literature. Our ADEANN system partially fills this gap, and provides accurate, automatic direct and recurrent ANNs for pattern recognition problems and dynamical systems simulations.

Function Rule-Extraction-with-GA

Step 1 – Dynamically allocate an array of integers ‘B’ with dimension $[I \times G]$, where I is the number of individuals in the population and G is the number of desired genes. Randomly assign 0 or 1 to each position (i, g) in the array.

Step 2 – Obtain the complementary array ‘B’.

For $i = 1$ to G :

For $g = 1$ to i

if $B[i, g] == 0$ $B[i, g] = 1$ else $B[i, g] = 0$;

Step 3 – Dynamically allocate an array of characters ‘D’ with dimension $[I \times (G/6)]$, where I is the number of individuals in the population and G is the number of desired genes.

Step 4 – For each chromosome (individual) of the population (each line of the binary array ‘B’), read six-bit sequences, starting from the first one. Each group of six bits must be converted to a symbol of the alphabet defined by the grammar (L-system) described in Section 4.1. This conversion, which must obey Table 2, generates a string. To determine the character encoded by each six-bit sequence, Table 2 is read in the following order:

1. Determine which of the four rows on the left corresponds to the first two bits of the string;
2. choose the column matching the central two bits;
3. choose the row on the right corresponding to the last two bits.

For example, the strings 001000, and 111111 translate into characters * and [, respectively. To each position (i', g') of the array ‘D’, assign the values converted from the binary characters in this step. Each line in array ‘D’ is represented by an integer string, as illustrated in Figure 4a).

Step 5 – For each (row) of Table ‘D’ obtained in Step 4, find the substring encoding the valid production rules, reading each row from the first character and continuing along contiguous or non-contiguous positions.

Step 5.1 – Discard unnecessary characters until the substring $.f[Ff * nB]$ is obtained.

Step 5.2 – Repeat Step 5, beginning the read of the current line of array ‘D’ from other positions; for example, from the start to the end and vice versa. Individuals in the population with invalid rules are assigned zero fitness.

The fitness of an ANN can be appropriately measured by the mean square error (MSE) given by Equation (3). The MSE measures the expected squared distance between the predicted and true values. As such, it measures the quality of a predictor. The system adjusts the weights of its neural networks to minimize the MSE in the training and test sets. ADEANN aims not only to minimize the MSE of the ANN in the training set but also to generalize and properly predict in the test set. In Equation (3), q is the number of test samples. The fitness function selects the most suitable neural networks (direct or recurrent) for a specific class of problem. In other words, depending on the simulated problem class, the fitness function, selects a direct or recurrent ANN most suitable for a specific task.

The fitness function (Equation (5)) automatically implements a penalty approach that rewards economical ANNs with better generalization capacities and ease of implementation (see the last fifteen lines and second column of Table 5). This function selects networks by two criteria; the number of hidden-layer neurons and the MSE. Therefore, smaller networks score higher fitness values than larger networks with the same performance. RNN-evolving NEAs are rarely reported in the literature. Our ADEANN system partially fills this gap, and provides accurate, automatic direct and recurrent ANNs for pattern recognition problems and dynamical systems simulations.

In this paper, the reproducing individuals are selected by tournament selection, one of the most widely used selection strategies in GAs [12]. In tournament selection, the mating pool consists of tournament winners. The average fitness is usually higher in the mating pool than in the population. The fitness difference between the mating pool and the population reflects the selection intensity (I), given by Equation (6). The approximate solution is given by Equation (7), where x and y are the fitness values of the population and t is the tournament size. Tournament selection is expected to improve the fitness of each subsequent generation [31].

$$MSE(k) = \frac{\sum_{p=1}^q (d_p - o_{kp})^2}{q} \quad (3)$$

where d_p and o_{kp} are the desired output and the output of individual k for pattern p , respectively. In our system, we propose a penalty approach that prevents the algorithm from unnecessarily producing ANNs with a large number of hidden neurons. To this end, our system compares two fitness functions, given by Equations (4) and (5):

$$f1 = \left(\frac{A1}{MSE} + \frac{A2}{NNHL} \right), \tag{4}$$

$$f2 = \left[\exp(-MSE) \times \exp(-NNHL) + \frac{1}{(MSE \times NNHL)} \right] \tag{5}$$

where MSE is the mean squared error and NNHL is the number of neurons in the hidden layer.

$$I = \int_{-\infty}^{\infty} tx \frac{1}{2\pi} e^{-\frac{x^2}{2}} \left(\int_{-x}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} dy \right)^{t-1} dx, \tag{6}$$

$$I \approx \sqrt{2 \left(\ln(t) - \ln \left(\sqrt{4.44 \ln(t)} \right) \right)}, \tag{7}$$

$$LD(t) = t^{\frac{-1}{t-1}} - t^{\frac{-t}{t-1}}. \tag{8}$$

Table 3 shows how the selection intensity calculated by Equation (8) varies with tournament size.

Tournament Size (<i>t</i>)	1	2	3	5	10	30
Selection Intensity (I)	0	0.56	0.85	1.15	1.53	2.04

Table 3. Relation between tournament size and selection intensity

In tournament selection, larger tournament size *t* is lead to higher expected loss of diversity (LD) [31]. As evident from Equation (8), approximately 50% of the population is lost at tournament size *t* = 5. Diversity is important in GAs because crossing over a homogeneous population does not yield new solutions.

In tournament selection, larger tournament size *t* is lead to higher expected loss of diversity (LD) [31]. Diversity is important in GAs because crossing over a homogeneous population does not yield new solutions. Accordingly, the tournament size *t* of ADEANN is set to 3, meaning that three chromosomes compete with each other, and the best chromosome among these three is selected for reproduction.

By recombining the genetic codes of two parents, the crossover operator produces two solutions in a yet unvisited region of the search space. For each pair of parents, the crossover operator is applied with a certain crossover probability (*Pc*). The benefits of increasing the number of crossover points in the crossover operation have been reported in many (largely empirical) studies [36]. ADEANN crosses multiple randomly selected points in the chromosome. The number of crossover points is proportional to the chromosome length divided by six and multiplied by the crossover rate (*Cr*). After applying the crossover operator, the genes of each resulting offspring are altered by the mutation operator. The mutation operator allows the GA to explore new solution space avoiding trapping in local optima. The mutation points in

ADEANN are randomly chosen with probability Pm . Their number is proportional to the chromosome length divided by six and multiplied by the mutation rate (Mr).

4.4 Network Models and Their Biological Relevance

The first network model generated by ADEANN in the classification experiment was a single hidden-layer feed-forward network. This model, characterized by a three-layer network of simple processing units connected by acyclic links, is commonly employed in systems biology studies [37], including evolutionary studies. Information flows through the ANN in discrete time. The output o_j of node j is calculated by Equation (9):

$$\vartheta(o_j) = \frac{1}{1 + \exp^{-k \cdot \sum_{i \in I_j} (w_{ij}x_i + b_j)}}. \quad (9)$$

Here, I_j is the set of nodes connected to node j , w_{ij} is the strength of the connection between node i and node j , o is the output value of node i , and b_j is the bias. The parameter k measures the steepness of the sigmoidal function (Equation (9)). As k is positive, the sigmoidal function is monotonically increasing, continuous and differentiable over the whole domain. In our experiments, we assigned the typical value for the training of neural networks with BP (namely, $k = 1$).

The parameters of the neural network w_{ij} are changed by an amount Δw_{ij} , which is calculated by Equation (10):

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} \quad (10)$$

where the parameter η is the learning rate and E is the error in the output layer. The δ term in Equation (11) is a momentum term, introduced by [38] to accelerate the learning process while avoiding instability in the algorithm.

The second network model generated by ADEANN, which was employed in the time-series forecasting experiments, was the single hidden-layer recurrent network. [39] discussed the nature of recurrence within the neocortex, a part of the brain that processes sensory stimuli. Such a recurrent structure of biological neuronal circuits is relevant to the field of artificial intelligence.

$$\Delta w_{ij}(t+1) = -\eta \frac{\partial E}{\partial w_{ij}} + \delta \Delta w_{ij}(t). \quad (11)$$

5 EXPERIMENTAL SETUP AND RESULTS WITH ADEANN

In this section we evaluate the performance of the ADEANN in evolving the architectures of ANNs.

5.1 Experimental Setup

Before conducting the formal experiments, we must set the ADEANN parameters. The parameter values were empirically determined from the results of several XOR experiments. Experiments were run on a V14T-5470-A60 Core i7 computer with 8 GB memory, operating at 3.2 GHz. The various combinations of the parameter values are listed in Table 5. The first column specifies the numbers of the experiments, generations, and individuals per population. The second column states the neural network solution returned by the search process. The third, fourth, and fifth columns, respectively, specify the chromosome length, the number of crossover and mutation points, and the crossover rate. The last four columns present the simulations results: the average (μ) and standard deviation (σ) of the number of neurons in the hidden layer, the MSE in the generalization phase, and the percentage of production rules. The superior settings are given in Table 4.

Pop. size (v)	Generations	C_r	P_m	P_{elit}
[30–100]	[50–500]	[0.3, 0.9]	0.1	2%

Table 4. Parameters used in the experiments

The results of the XOR problem after twenty experiments are summarized in Table 5. In the last fifteen simulations, the fitness given by Equation (5) automatically prevents the algorithm from unnecessarily producing complex ANNs with a large number of hidden neurons, and encourages smaller topologies. This constraint improves the generalization capacity of the ANNs. All networks returned by the search process contained two neurons in the input layer, two in the hidden layer, and one in the output layer, and can be described by ANN = (2, 2, 1). This structure is not guaranteed by the fitness function Equation (4), because the appropriate coefficients for this equation are determined by a problem dependent, time-consuming process that obtains A1 and A2 by trial-and-error. Furthermore, there is no guarantee that the search will return a small network. In the first five experiments (summarized in the first five lines of Table 5), all of the ANNs returned by the search process contained 8, 7, 6, or 5 neurons in the hidden layer. The results of the best (19th) and worst (first) experiments, in terms of small MSE and minimum number of hidden neurons, are highlighted in bold in Table 5.

From Table 6, we observe that in 100 runs NEAT [6] finds a structure for XOR in 32 generations (4 755 networks evaluated, $\sigma = 2.553$). The average number of hidden nodes in the solution network was $\mu = 2.35$. The 13th experiment (Table 6) yielded the closest solution, with $\mu = 3.31$ and $\sigma = 1.65$ in 50 generations. ADEANN has a similar average classification accuracy to DENN [32], but more accurately estimates the best network in the search process ($MSE \leq 0.000024$). ADEANN easily solves the XOR problem while maintaining a small architecture. The numbers of nodes and connections were close to optimal, considering that the smallest ANN had a single hidden unit. Moreover, the ADEANN always found a solution. The total number of ANNs trained in the experiments was 30 100.

GxP	ANN	CL	(c, m)	Cr	μ	σ	MSE	PR [%]
E1-50x30	(2, 8, 1)	180	(9, 3)	0.6	6.50	1.50	0.000046	6.67
E2-50x30	(2, 7, 1)	216	(10, 3)	0.6	4.80	2.32	0.000049	16.67
E3-50x30	(2, 5, 1)	252	(12, 4)	0.6	4.80	1.72	0.000048	25
E4-60x30	(2, 6, 1)	252	(12, 4)	0.9	3.86	1.81	0.000032	23.33
E5-50x20	(2, 6, 1)	276	(13, 5)	0.6	6.00	1.94	0.000029	20.28
E6-50x30	(2, 2, 1)	180	(18, 3)	0.3	5.67	2.62	0.000076	9
E7-50x30	(2, 2, 1)	180	(18, 3)	0.6	3.20	0.75	0.000075	13.40
E8-50x30	(2, 2, 1)	180	(18, 3)	0.9	4.50	2.18	0.000051	11.33
E9-50x20	(2, 2, 1)	216	(32, 28)	0.6	6.00	2.07	0.000056	22.33
E10-50x20	(2, 2, 1)	216	(32, 28)	0.9	3.71	2.43	0.000056	21.53
E11-50x30	(2, 2, 1)	252	(37, 33)	0.6	3.89	3.65	0.000056	28.67
E12-60x30	(2, 2, 1)	252	(37, 33)	0.9	4.36	1.72	0.000055	26.73
E13-50x30	(2, 2, 1)	300	(45, 40)	0.6	3.31	1.65	0.000053	51.20
E14-60x30	(2, 2, 1)	300	(45, 40)	0.9	3.77	2.01	0.000053	41.93
E15-50x30	(2, 2, 1)	360	(54, 48)	0.6	3.69	2.14	0.000056	50.73
E16-50x30	(2, 2, 1)	360	(54, 48)	0.9	3.71	1.87	0.000056	54.33
E17-50x30	(2, 2, 1)	390	(58, 52)	0.6	3.79	1.61	0.000053	63.33
E18-50x30	(2, 2, 1)	390	(58, 52)	0.9	5.10	2.05	0.000053	63.40
E19-100x30	(2, 2, 1)	516	(77, 68)	0.9	5.28	2.03	0.000024	96.67
E20-100x30	(2, 2, 1)	516	(77, 68)	0.6	4.15	2.17	0.000025	90

Table 5. Genetic algorithm parameters and simulation results of 20 experiments of the XOR problem. ANN = (X, Y, Z) , where X , Y , and Z are the number of neurons in the input, hidden and output layers, respectively.

Problem	NEAT	ADEANN	DENN
XOR	– 2.35 2.553 –	98.84% ^a 3.31 ^b 1.65^c 0.000024^d	98.97% – – 0.004865

Table 6. Comparison between ADEANN and other methods in the XOR problem. ^a Average classification accuracy, ^b Average number of hidden neurons, ^c Standard deviation of the number of hidden neurons, ^d MSE, DENN [32].

5.2 Consumer Price Index – CPI Problem

To verify the viability of the method for large problems, we carried out experiments for the prediction of the time series of the consumer price index (CPI), for the period of January 1998 to May 2010 (source: Central Bank of Brazil, as seen in Table 7). The CPI quantifies the costs of products at different moments. In other words, it measures changes through time in the price level of consumer goods and services purchased by households and is useful for the calculation of the inflation rate.

In five runs, the best experiment shows that the ADEANN system finds an ANN for CPI in 50 generations (1500 networks evaluated, std $\sigma = 4.39$) and MSE =

	Jan	Feb	Mar	Apr	May	Jun
1998	0.0126	0.0014	0.0033	0.0023	0.0014	0.0041
1999	0.0064	0.0141	0.0095	0.0052	0.0008	0.0065
2000	0.0101	0.0005	0.0051	0.0025	0.004	-0.0001
2001	0.0064	0.004	0.0056	0.0086	0.0041	0.0052
2002	0.0079	0.0014	0.0042	0.0071	0.0028	0.0055
2003	0.0232	0.0137	0.0106	0.0112	0.0069	-0.0016
2004	0.0108	0.0028	0.0046	0.0031	0.0071	0.0078
2005	0.0085	0.0043	0.007	0.0088	0.0079	-0.0005
2006	0.0065	0.0001	0.0022	0.0034	-0.0019	-0.004
2007	0.0069	0.0034	0.0048	0.0031	0.0025	0.0042
2008	0.0097	0.0056	0.0045	0.0072	0.0087	0.0077
2009	0.0083	0.0021	0.0061	0.0047	0.0039	0.0012
2010	0.0129	0.0068	0.0086	0.0076	0.0021	-

	Jul	Aug	Sep	Oct	Nov	Dec
1998	-0.0025	-0.0052	-0.0017	0.002	-0.0019	0.0009
1999	0.012	0.0048	0.0019	0.0092	0.0112	0.006
2000	0.0191	0.0086	0.0004	0.0002	0.004	0.0062
2001	0.0136	0.0054	0.0012	0.0071	0.0085	0.007
2002	0.0103	0.0076	0.0066	0.0114	0.0314	0.0194
2003	0.0034	0.0013	0.0076	0.0021	0.0033	0.0043
2004	0.0059	0.0079	0.0001	0.001	0.0037	0.0063
2005	0.0013	-0.0044	0.0009	0.0042	0.0057	0.0046
2006	0.0006	0.0016	0.0019	0.0014	0.0024	0.0063
2007	0.0028	0.0042	0.0023	0.0013	0.0027	0.007
2008	0.0053	0.0014	-0.0009	0.0047	0.0056	0.0052
2009	0.0034	0.002	0.0018	0.0001	0.0026	0.0024

Table 7. Normalized values for CPI in the period of January 1998 to May 2010. Source: Central Bank of Brazil.

0.000015972. On average, a solution network has $\mu = 5.21$ hidden nodes. The worst performance takes 1500 evaluations, or about 50 generations, the standard deviation was $\sigma = 5.42$, the average $\mu = 9.78$ for the number of hidden nodes and $MSE = 0.0000159912$. Since the fitness function (5) takes into account the minimal number of neurons in the hidden layer (NNHL) and a minimal mean squared error (MSE). ADEANN solves the CPI problem without trouble. The lowest ANN for that problem has the following specification: (1, 3, 1). The largest ANN obtained for the CPI problem has the specification (1, 18, 1), i.e., with 18 neurons in the hidden layer. It had an $MSE = 0.000015733$ and good generalizability, with $MSE \leq 0.0001$ and is better than the ANN (1, 3, 1). However, as the fitness, as given by Equation (5), favors the smaller ANNs with good generalization capacity, the ANN (1, 3, 1) is ranked the best.

5.3 Prediction of the Effect of a New Drug on Breast Cancer (BC)

The problem of predicting the effect of a new drug on breast cancer [26] is to find an analogous mapping between the set of 15 experimentally induced tumors and a clinical tumor (malignant breast cancer), based on their known reactions to the same drugs, so that we are able to predict the effect of a new drug on the clinical tumor after having studied the experimental tumors. The solution is based on the presumed similarity between this tumor and a set of experimental tumors. Table 8 illustrates the dataset used for training, which contains the known effect of 26 established drugs on 15 experimental tumors and on a clinical tumor.

	T1	2	3	4	5	6	7	8	9	T0	11	12	13	14	15	TC
D1	1	1	0.5	0.5	1	0.5	1	1	0.5	0.5	0.5	1	1	1	0.5	1
2	1	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1
3	1	1	1	1	0	0	1	1	0	1	0	1	1	1	1	0
4	1	1	0.5	0	0	0.5	1	0.5	0.5	0.5	0.5	1	1	0.5	0.5	1
5	1	1	1	0	1	0	1	1	0	1	0	0	1	0	1	1
6	1	1	0	1	0	0.5	1	1	1	0	0	0	1	1	0.5	1
7	1	1	0	1	0	0.5	1	1	1	0	0	0	1	1	0.5	1
8	0	1	0	1	0	0.5	0	1	1	0	0	1	0	0.5	1	1
9	1	1	1	0	0	0	0	0	1	0	0.5	1	0	0	1	1
10	1	1	1	1	1	0	1	1	1	0	1	0	0	0	1	1
11	1	1	1	0	0	0.5	0	0	0.5	0	0.5	1	1	0	0.5	1
12	1	0.5	1	0.5	0	0.5	0	0	0.5	0	1	0.5	1	0.5	0.5	1
13	0	0	1	0	1	0	0.5	0.5	0.5	0	0.5	1	1	0.5	0.5	0
14	1	1	1	0	0.5	0.5	0	0.5	0	0	0	0.5	0.5	0.5	0	0
15	1	0.5	1	1	0	0	1	1	0	0	0	0	0	0	0	0
16	1	0.5	0.5	1	1	1	0	0.5	1	1	0.5	0.5	1	0.5	1	1
17	1	0	0.5	1	1	0.5	1	0.5	0.5	1	0.5	0.5	0	0.5	0.5	1
18	1	0.5	1	0.5	0	0.5	1	1	0.5	0	0.5	1	1	0.5	0.5	1
19	0	0	1	1	0.5	1	1	1	0.5	0	0.5	0.5	1	0.5	0.5	1
20	1	1	1	0	0	0.5	0	1	0.5	0.5	0.5	0.5	1	1	1	0
21	1	1	1	1	0	0	1	0	0	1	1	0	1	1	1	1
22	1	0	0.5	1	0.5	1	0	0.5	1	1	0.5	0.5	1	0	1	1
23	1	0	0	1	1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0	0.5	0.5	1
24	1	0.5	0.5	0.5	0	0.5	1	0.5	0.5	0	0.5	1	0.5	0.5	0.5	1
25	0	0	0.5	0.5	0	0	1	1	1	0.5	0.5	1	0	1	1	1
26	0	0.5	0.5	0	0	0.5	1	0.5	1	0.5	0.5	0	0.5	0.5	1	0.5

Table 8. The effect of 26 drugs on 15 experimental tumors and on a tumor in a patient with breast cancer

In five runs, the best experiment shows that the ADEANN system finds an ANN for BC in 100 generations (3000 networks evaluated, std $\sigma = 1.206$) and MSE = 0.000005. On average, a solution network has $\mu = 6.83$ hidden nodes. The

minimal neural network which was found has the following specification (neurons in the input layer, neurons in the hidden layer, neurons in the output layer) = (15, 6, 1). The testing results for that network are shown in Table 9. In the method of [26], on average a solution network had $\mu = 3.6$ neurons in the hidden layer, $\text{std } \sigma = 1.1$ and $\text{MSE} = 0.015$. Table 10 illustrates a comparison with a wider variety of methods.

Obtained Output	Desired Output	Error
0.999700	1.0	0.000000045
0.998717	1.0	0.000000823
0.999973	1.0	0.000000000
0.999963	1.0	0.000000001
0.999546	1.0	0.000000103
0.998482	1.0	0.000001153
0.996649	1.0	0.000005616
0.997020	1.0	0.000004441
0.007759	0.0	0.000030104
0.999862	1.0	0.000000000
0.999046	1.0	0.000000455

Table 9. Testing results for the ANN for predicting the effect of 15 new drugs on malignant breast cancer. Eleven test examples are used. If a threshold of 1.0 is chosen to decide on an active neuron, the effects of all the new drugs tested are correctly predicted.

Dataset	ADEANN	G3PCX [48]	GA [46]
BC	99.11 % ^a 14.72 ^b 2.54 ^c 0.000115 % ^d	98.94 % 5 0 -	98.88 % 5 0 -

Table 10. Comparison between ADEANN and other methods for the Breast Cancer Dataset. ^a Average classification accuracy, ^b Average number of hidden neurons, ^c Standard deviation of the number of hidden neurons, ^d MSE.

One of the major problems facing researchers is the selection of hidden neurons using neural networks. There is no theory to tell us how to choose the optimal number of hidden units, moreover, it cannot be generalized that accuracy will increase proportionally with increase in number of hidden neurons or vice versa. [43] suggests that the optimal number of neurons in the hidden layer for a 3-layer neural network is given by Equation (10), where N = number of inputs, m = number of outputs and NNHL is the number of neurons in the hidden layer. The proposed model improves the accuracy and minimal error. For this problem $m = 1$ and $N = 15$, replacing these values in Equation (12) yields $\text{NNHL} = 11.2$. Regarding Table 10, ADEANN indicates that the average number of hidden neurons for BC problem is 14.72. Which shows that ADEANN yielded the closest solution among all three methods.

$$\text{NNHL} = \sqrt{(m + 2)N} + 2 * \sqrt{\frac{N}{m + 2}}. \tag{12}$$

6 DISCUSSION

The first challenge that we have addressed was the scalability problem. Our approach includes improvements to our previous proposal [7]. First, the chromosome used by the GA in our research has a variable length, from 180 to 516 bits, in contrast to before, when it had a fixed length of 1 024 bits. For this reason, our current approach evolves a compressed description of the ANN that reduces the search space of the GA. Furthermore, it reduces the scalability problem of other methods [27, 28]. In the section about the experiments we present results that confirm these improvements.

The method of [28], as described in [7], improved the scalability problem presented by the method of [27], since an ANN with N neurons will result in a binary string of $(N(N + 1)/2)$ in contrast to the method of [27], which has a quadratic behavior N^2 . If the two methods [27, 28] had used the chromosome length that we used in the experiments described in Section 5, which are in the interval [180, 516], the methods of [27] and [28] would generate ANNs with a maximal number of neurons in the intervals [13, 22] and [18, 31], respectively. Depending on the value of NR , our L-System can generate larger ANNs than the methods of [27, 28].

Our method also enables network optimization. As discussed in Section 4.3, the fitness function (Equation (5)) automatically implements a penalty approach that rewards economical ANNs with better generalization capacities and ease of implementation (see the last fifteen lines and the second column of Table 5). This function selects networks by two criteria: the number of hidden-layer neurons, and the MSE. Therefore, smaller networks score higher fitness values than larger networks with the same performance. RNN-evolving NEAs are rarely reported in the literature. Our ADEANN system partially fills this gap, and provides accurate, automatic direct and recurrent ANNs for pattern recognition problems and dynamical systems simulations. Many methods do not worry, as stated by [24], about avoiding optimizing overly complex topologies. Thus, at least part of the evolutionary search will be conducted in an unnecessarily high dimensional space, which is not desirable, as in some situations, less complex topologies can simulate the same task with the same precision.

Our method also enables network optimization. As discussed in Section 5.1, the fitness function Equation (5) automatically implements a penalty approach that rewards economical ANNs with better generalization capacities and ease of implementation (see the last fifteen lines and second column of Table 5). This function selects networks by two criteria; the number of hidden-layer neurons and the MSE. Therefore, smaller networks score higher fitness values than larger networks with the same performance. The overall topology and the number of hidden neurons are indeed dependent on the configuration of the algorithm.

Methods that increase or decrease progressively the size of the hidden layer are relatively fast, but may generate sub-optimal solutions (all greedy optimisation methods are liable to converge to local optima). Metaheuristics (like genetic algorithms) are more likely to find the global optimum but might be too slow.

However, our proposed approach is not without weaknesses. Our approach is much more sensitive to the initial conditions and constants than existing methods. As the initial population is randomly initialized with 0s and 1s, the generated genotypes are difficult to control. Therefore, when configuring the initial population, we must apply stochastic methods such as the Gaussian distribution. Moreover, as the variable NR in Equations (1) and (2) is randomly generated, the algorithm cannot generate all possible neuron topologies in the interval $[X, Y]$ in a specific generation.

7 CONCLUSION

ADEANN constitutes an automatic system to simulate tasks of pattern recognition, without any previous knowledge from the user. Also, two modifications of an earlier method [7] have been evaluated. The first one involves the fitness function, given by Equation (5). It explicitly rewards smaller solutions. Our approach solves the three problems presented previously, in Section 5, without trouble in finding small topologies. The second, our L-System, presented in Section 4.2, enables generating more complex, direct, recurrent and multilayer networks, than the previous one [7]. Other contributions are: our L-System allows generating ANN architectures without requiring additional configuration of substrates for each particular type of problem, as required by the methods of [8, 10], our model automates this process. Only the method of [15] formalizes the problem of ADANNs as a local search strategy. In Section 3, we described a new approach to formalizing the problem of ADANNs as a local search based on rational agents. Our approach increases the level of implicit parallelism of GA, which evolves a compressed description of the L-rules and enables generating more complex, direct, recurrent and multilayer networks than the previous one [7].

NEAs that evolve recurrent neural networks are rarely found in the literature [34]. To partially fill this gap, we designed ADEANN as an accurate automatic method for designing direct and recurrent ANNs and thereby solving pattern recognition problems and simulating dynamical systems.

Ultimately, the biological inspiration addressed in our research, makes it original. The merits of our bio-inspired algorithm compared to conventional algorithms are: flexibility, performance, scalability, flexibility in decision making and improvement scope. In future work, we will tackle other challenges, such as partially connected networks and ANNs that use Hebbian learning, thus exploiting aspects of neural plasticity.

REFERENCES

- [1] STANLEY, K. O.—MIKKULAINEN, R.: A Taxonomy for Artificial Embryogeny. *Artificial Life*, Vol. 9, 2003, No. 2, pp. 93–130, doi: 10.1162/106454603322221487.
- [2] BUKHTOYAROV, V.—SEMENKIN, E.: Evolutionary Three-Stage Approach for Designing of Neural Networks Ensembles for Classification Problems. In: Tan, Y.,

- Shi, Y., Mo, H. (Eds.): *Advances in Swarm Intelligence (ICSI 2013)*. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7928, 2013, pp. 467–477.
- [3] KRÖMER, P.—PLATOŠ, J.—SNÁŠEL, V.: Nature-Inspired Meta-Heuristics on Modern GPUs: State of the Art and Brief Survey of Selected Algorithms. *International Journal of Parallel Programming*, Vol. 42, 2014, No. 5, pp. 681–709, doi: 10.1007/s10766-013-0292-3.
- [4] GRUAU, F.: *Neural Network Synthesis Using Cellular Encoding and the Genetic Algorithm*. Unpublished doctoral dissertation, Ecole Normale Supérieure de Lyon, l'Université Claude Bernard Lyon 1, 1994.
- [5] PARK, H. S.—TRAN, N. H.: Development of a Biology Inspired Manufacturing System for Machining Transmission Cases. *International Journal of Automotive Technology*, Vol. 14, 2013, No. 2, pp. 233–240, doi: 10.1007/s12239-013-0026-y.
- [6] STANLEY, K. O.—MIKKULAINEN, R.: Evolving Neural Networks Through Augmenting Topologies. *Evolutionary Computation*, Vol. 10, 2002, No. 2, pp. 99–127, doi: 10.1162/106365602320169811.
- [7] DE CAMPOS, L.—ROISENBERG, M.—DE OLIVEIRA, R.: Automatic Design of Neural Networks with L-Systems and Genetic Algorithms – A Biologically Inspired Methodology. *The 2011 International Joint Conference on Neural Networks (IJCNN)*, 2011, pp. 1199–1206, doi: 10.1109/IJCNN.2011.6033360.
- [8] GAUCI, J.—STANLEY, K. O.: Autonomous Evolution of Topographic Regularities in Artificial Neural Networks. *Neural Computation*, Vol. 22, 2010, No. 7, pp. 1860–1898, doi: 10.1162/neco.2010.06-09-1042.
- [9] LEE, D.-W.—KONG, S. G.—SIM, K.-B.: Evolvable Neural Networks Based on Developmental Models for Mobile Robot Navigation. *Proceedings of 2005 IEEE International Joint Conference on Neural Networks (IJCNN '05)*, 2005, Vol. 1, 2005, pp. 337–342.
- [10] RISI, S.—STANLEY, K. O.: An Enhanced Hypercube-Based Encoding for Evolving the Placement, Density, and Connectivity of Neurons. *Artificial Life*, Vol. 18, 2012, No. 4, pp. 331–363.
- [11] BENARDOS, P. G.—VOSNIAKOS, G.-C.: Optimizing Feedforward Artificial Neural Network Architecture. *Engineering Applications of Artificial Intelligence*, Vol. 20, 2007, No. 3, pp. 365–382, doi: 10.1016/j.engappai.2006.06.005.
- [12] MILLER, G. F.—TODD, P. M.—HEGDE, S. U.: *Designing Neural Networks Using Genetic Algorithms*. Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1989, pp. 379–384.
- [13] ZHANG, J.-R.—ZHANG, J.—LOK, T.-M.—LYU, M. R.: A Hybrid Particle Swarm Optimization-Back-Propagation Algorithm for Feedforward Neural Network Training. *Applied Mathematics and Computation*, Vol. 185, 2007, No. 2, pp. 1026–1037, Special Issue on Intelligent Computing Theory and Methodology.
- [14] ZHANG, L.—SUN, Y.: Evolved Neural Network Based Intelligent Trading System for Stock Market. In: Tan, Y., Shi, Y., Mo, H. (Eds.): *Advances in Swarm Intelligence (ICSI 2013)*. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7928, 2013, pp. 478–488.

- [15] DONATE, J. P.—SANCHEZ, G. G.—DE MIGUEL, A. S.: Time Series Forecasting. A Comparative Study Between an Evolving Artificial Neural Networks System and Statistical Methods. *International Journal on Artificial Intelligence Tools*, Vol. 21, 2012, No. 1, Art. No. 1250010.
- [16] CLUNE, J.—MOURET, J.-B.—LIPSON, H.: The Evolutionary Origins of Modularity. *Proceedings of the Royal Society B*, Vol. 280, 2013, No. 1775, Art. No. 20122863, Cite arxiv:1207.2743, doi: 10.1098/rspb.2012.2863.
- [17] CANTU-PAZ, E.—KAMATH, C.: An Empirical Comparison of Combinations of Evolutionary Algorithms and Neural Networks for Classification Problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, Vol. 35, 2005, No. 5, pp. 915–927, doi: 10.1109/TSMCB.2005.847740.
- [18] DEB, K.—ANAND, A.—JOSHI, D.: A Computationally Efficient Evolutionary Algorithm for Real-Parameter Optimization. *Evolutionary Computation*, Vol. 10, 2002, No. 4, pp. 371–395.
- [19] RISI, S.—STANLEY, K. O.: A Unified Approach to Evolving Plasticity and Neural Geometry. *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2012)*, IEEE, 2012, pp. 1–8, doi: 10.1109/IJCNN.2012.6252826.
- [20] LEE, S.—YOSINSKI, J.—GLETTE, K.—LIPSON, H.—CLUNE, J.: Evolving Gaits for Physical Robots with the HyperNEAT Generative Encoding: The Benefits of Simulation. In: *Esparcia-Alcázar, A. I. (Ed.): Applications of Evolutionary Computation (EvoApplications 2013)*. Springer, Berlin, Heidelberg, *Lecture Notes in Computer Science*, Vol. 7835, 2013, pp. 540–549.
- [21] DE CAMPOS, L. M. L.—DE OLIVEIRA, R. C. L.: A Comparative Analysis of Methodologies for Automatic Design of Artificial Neural Networks from the Beginnings Until Today. *Proceedings of the 2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence (BRICS-CCI-CBIC '13)*, IEEE Computer Society, Washington, DC, USA, 2013, pp. 453–458, doi: 10.1109/BRICS-CCI-CBIC.2013.81.
- [22] SIMON, H. A.: *The Sciences of the Artificial*. 3rd Edition, The MIT Press, Cambridge, MA, 1996.
- [23] RUSSELL, S. J.—NORVIG, P.: *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.
- [24] WHITESON, S.: Evolutionary Computation for Reinforcement Learning. In: *Wiering, M., van Otterlo, M. (Eds.): Reinforcement Learning*. Springer, Berlin, Heidelberg, *Adaptation, Learning, and Optimization*, Vol. 12, 2012, pp. 325–355, doi: 10.1007/978-3-642-27645-3_10.
- [25] PRUSINKIEWICZ, P.—RUNIONS, A.: Computational Models of Plant Development and Form. *The New Phytologist*, Vol. 193, 2012, No. 3, pp. 549–569, doi: 10.1111/j.1469-8137.2011.04009.x.
- [26] ISLAM, M.—YAO, X.—MURASE, K.: A Constructive Algorithm for Training Cooperative Neural Network Ensembles. *IEEE Transactions on Neural Networks*, Vol. 14, 2003, No. 4, pp. 820–834.
- [27] KITANO, H.: Designing Neural Networks Using Genetic Algorithms with Graph Generation System. *Complex Systems*, Vol. 4, 1990, pp. 461–476.

- [28] BOOZARJOMEHRY, R. B.—SVRCEK, W. Y.: Automatic Design of Neural Networks Structures. *Computers and Chemical Engineering*, Vol. 25, 2001, No. 7-8, pp. 1075–1088, doi: 10.1016/S0098-1354(01)00680-9.
- [29] SÁNCHEZ, D.—MELIN, P.: Optimization of Modular Granular Neural Networks Using Hierarchical Genetic Algorithms for Human Recognition Using the Ear Biometric Measure. *Engineering Applications of Artificial Intelligence*, Vol. 27, 2014, pp. 41–56, doi: 10.1016/j.engappai.2013.09.014.
- [30] KAZARYAN, D. E.—SAVINKOV, A. V.: Grammatical Evolution for Neural Network Optimization in the Control System Synthesis Problem. *Procedia Computer Science*, Vol. 103, 2017, pp. 14–19, doi: 10.1016/j.procs.2017.01.002.
- [31] OLADELE, R. O.—SADIKU, J. S.: Genetic Algorithm Performance with Different Selection Methods in Solving Multi-Objective Network Design Problem. *International Journal of Computer Applications*, Vol. 70, 2013, pp. 5–9.
- [32] ILONEN, J.—KAMARAINEN, J.-K.—LAMPINEN, J.: Differential Evolution Training Algorithm for Feed-Forward Neural Networks. *Neural Processing Letters*, Vol. 17, 2003, No. 1, pp. 93–105.
- [33] DE CAMPOS, L. M. L.—DE OLIVEIRA, R. C. L.—ROISENBERG, M.: Optimization of Neural Networks Through Grammatical Evolution and a Genetic Algorithm. *Expert Systems with Applications*, Vol. 56, 2016, pp. 368–384, doi: 10.1016/j.eswa.2016.03.012.
- [34] HORNBY, G. S.—POLLACK, J. B.: Creating High-Level Components with a Generative Representation for Body-Brain Evolution. *Artificial Life*, Vol. 8, 2002, pp. 223–246, doi: 10.1162/106454602320991837.
- [35] LEE, D. W.—SEO, S.-W.—SIM, K.-B.: Evolvable Neural Networks Based on Developmental Models for Mobile Robot Navigation. *International Journal of Fuzzy Logic and Intelligent Systems*, Vol. 7, 2007, pp. 176–181, doi: 10.5391/IJFIS.2007.7.3.176.
- [36] DE JONG, K. A.—SPEARS, W. M.: A Formal Analysis of the Role of Multi-Point Crossover in Genetic Algorithms. *Annals of Mathematics and Artificial Intelligence*, Vol. 5, 1992, No. 1, pp. 1–26, doi: 10.1007/BF01530777.
- [37] ALON, U. J.: *An Introduction to Systems Biology: Design Principles of Biological Circuits*. 1st Edition, Chapman and Hall/CRC Mathematical and Computational Biology, Chapman and Hall/CRC, 2006.
- [38] RUMELHART, D.—MCCLELLAND, J. L.: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1: Foundations. Cambridge, MA, MIT Press, 1986.
- [39] DOUGLAS, R. J.—MARTIN, K. A. C.: Recurrent Neuronal Circuits in the Neocortex. *Current Biology*, Vol. 17, 2007, No. 13, pp. 496–500, doi: 10.1016/j.cub.2007.04.024.
- [40] DAWKINS, R.: *The Blind Watchmaker*. Longman Scientific and Technical, 1986.
- [41] WRZESZCZ, M.—KOŹLAK, J.—KITOWSKI, J.: Modelling Agents Cooperation Through Internal Visions of Social Network and Episodic Memory. *Computing and Informatics*. Vol. 36, 2017, No. 1, 86–112, doi: 10.4149/cai.2017.1.86.
- [42] FERNÁNDEZ-DOMINGOS, E.—LOUREIRO, M.—ÁLVAREZ-LÓPEZ, T.—BURGUILLO, J. C.—COVELO, J.—PELETEIRO, A.—BYRSKI, A.: Emerging Cooperation in N-

- Person Iterated Prisoner's Dilemma over Dynamic Complex Networks. *Computing and Informatics*, Vol. 36, 2017, No. 3, pp. 493–516, doi: 10.4149/cai_2017.3_493.
- [43] HUANG, G.-B.: Learning Capability and Storage Capacity of Two-Hidden-Layer Feedforward Networks. *IEEE Transactions on Neural Networks*, Vol. 14, 2003, No. 2, pp. 274–281.
- [44] OJHA, V. K.—DUTTA, P.—CHAUDHURI, A.—SAHA, H.: Understating Continuous Ant Colony Optimization for Neural Network Training: A Case Study on Intelligent Sensing of Manhole Gas Components. *International Journal of Hybrid Intelligent Systems*, Vol. 12, 2015, No. 4, pp. 185–202.
- [45] ABD ELRAHMAN, S. M.—ABRAHAM, A.: Class Imbalance Problem Using a Hybrid Ensemble Approach. *International Journal of Hybrid Intelligent Systems*, Vol. 12, 2015, No. 4, pp. 219–227.
- [46] DOS SANTOS, G. A. M.—FERRÃO, V. T.—VINHAL, C. D. N.—DA CRUZ, G.: Performance Analysis for a Novel Adaptive Algorithm for Realtime Point Cloud Ground Segmentation. *International Journal of Hybrid Intelligent Systems*, Vol. 12, 2015, No. 4, pp. 229–243.
- [47] SAMUEL, O. W.—ASOGBON, G. M.—SANGAIAH, A. K.—FANG, P.—LI, G.: An Integrated Decision Support System Based on ANN and Fuzzy AHP for Heart Failure Risk Prediction. *Expert Systems with Applications*, Vol. 68, 2017, pp. 163–172, doi: 10.1016/j.eswa.2016.10.020.
- [48] SALCEDO-SANZ, S.: Modern Meta-Heuristics Based on Nonlinear Physics Processes: A Review of Models and Design Procedures. *Physics Reports*, Vol. 655, 2016, pp. 1–70, doi: 10.1016/j.physrep.2016.08.001.
- [49] YU, K.—LI, Y.—CAI, Z.: A Hybrid Generic Algorithm for Dynamic Data Mining in Investment Decision Making. *International Journal on Data Science and Technology*, Vol. 2, 2016, No. 6, pp. 62–71.
- [50] VIVEKANANDAN, T.—SRIMAN NARAYANA IYENGAR, N. CH.: Optimal Feature Selection Using a Modified Differential Evolution Algorithm and Its Effectiveness for Prediction of Heart Disease. *Computers in Biology and Medicine*, Vol. 90, 2017, pp. 125–136, doi: 10.1016/j.combiomed.2017.09.011.



Lidio Mauro Lima DE CAMPOS is Adjunct Professor at the Faculty of Computing at the Federal University of Pará. He has his B.Sc. in electrical engineering and informatics from this same university, his M.Sc. degree in computer science from the Federal University of Santa Catarina and Ph.D. degree in electrical engineering from the Federal University of Pará. His research interests are in the field of machine learning, neural networks and hybrid intelligent systems.



Roberto Célio Limão DE OLIVEIRA graduated in electrical engineering from Universidade Federal do Pará (1987), received his Master's in electrical engineering from Instituto Tecnológico de Aeronáutica (1991) and Ph.D. in electrical engineering from Universidade Federal de Santa Catarina (1999). He has experience in electrical engineering, focusing on computational intelligence, artificial neural networks and evolutionary computing.



Gustavo Augusto Lima DE CAMPOS is Associate Professor at UECE with an expertise in the field of artificial intelligence, mainly intelligent agents to solve decision problems in complex environment, based on systematic and local search methods, neural networks, logic and fuzzy systems. He obtained his Doctor degree in electrical engineering at the Systems Engineering Department at the Electrical and Computing Engineering Faculty of the State University of Campinas, São Paulo, in 2003.

GRANULAR PARTITION AND CONCEPT LATTICE DIVISION BASED ON QUOTIENT SPACE

Qiang WU, Haiyan SHI

*Department of Computer Science and Engineering
Shaoxing University, Shaoxing, China
e-mail: {cswq, csshy}@usx.edu.cn*

Liping XIE

*Mechanical and Electrical Engineering College
Shaoxing University, Shaoxing, China
e-mail: xielp@usx.edu.cn*

Abstract. In this paper, we investigate the relationship between the concept lattice and quotient space by granularity. A new framework of knowledge representation – granular quotient space – is constructed and it demonstrates that concept lattice classing is linked to quotient space. The covering of the formal context is firstly given based on this granule, then the granular concept lattice model and its construction are discussed on the sub-context which is formed by the granular classification set. We analyze knowledge reduction and give the description of granular entropy techniques, including some novel formulas. Lastly, a concept lattice constructing algorithm is proposed based on multi-granular feature selection in quotient space. Examples and experiments show that the algorithm can obtain a minimal reduct and is much more efficient than classical incremental concept formation methods.

Keywords: Classification set, granule, quotient space, concept lattice, entropy

1 INTRODUCTION TO THE FIELD

FCA (Formal Concept Analysis) is usually called concept lattice. It is a mathematical framework for discovery and design of concept hierarchies from an information

system called a formal context [10]. All concepts in a formal context form a concept lattice and can be depicted by the Hasse diagram, where each node expresses a formal concept. The concept lattice is the core structure of data in formal concept analysis, from which a set of objects and a set of attributes are uniquely reflected from each other and the relations between the generalized concept and the specialized concept are described. Formal concept analysis is an important theory for data analysis and knowledge discovery [8, 4, 6, 14, 21, 3]. Many researchers have done some research about the concept lattice and granular computing. Wu [11] introduced attribute granules in formal contexts. The mathematical structure of attribute granules was investigated. Based on the theory of concept lattice and fuzzy concept lattice, Zhang et al. [20] established a mathematical model of a concept granular computing system. And various variable threshold concept lattices and fuzzy concept lattices were then investigated. For this research, concept granules, sufficiency information granules and necessity information granules which were used to express different relations between a set of objects and a set of attributes were proposed. Approaches to construct sufficiency and necessity information granules were also shown. Wu et al. [12] proposed a granular consistent set and a granular reduct in the formal context. Discernibility matrices and Boolean functions were, respectively, employed to determine granular consistent sets and calculate granular reducts in formal contexts. Based on granular computing theory, Zhang et al. [19] proposed a new formal concept analysis method with various granularity levels of attributes. Many theorems and relationships among the concept lattices and its sublattices generated from formal context with various granularity levels of attributes were analyzed in detail. Yao et al. [15] proposed a framework for studying a particular class of set-theoretic approaches to granular computing. They thought that a granule was a subset of a universal set, a granular structure was a family of subsets of the universal set, and the relationship between granules was the standard set-inclusion relation. Qiu et al. [7] introduced extent-intent and intent-extent operators between two complete lattices and established a mathematical model for concept granular computing system. They proved that the set of all concepts in this system was a lattice with the greatest element and the least element. This framework included formal concept lattices from formal contexts, L fuzzy concept lattices from L fuzzy formal contexts and three kinds of variable threshold concept lattices. Other studies for the extension of concept lattice models for granular computing are also underway [5, 9, 17, 22].

2 RELATED WORK

The basic idea of the quotient space mainly describes a class of problems through a triple [16, 18], such as (X, f_a, T) , where X represents the universe, f_a represents the set of attributes in the universe, T is the topological structure of X and represents relationships between members in the universe. The solution of the problem can be regarded as discussing the universe X , the attributes and related structures.

Given a special size, i.e. when a special equivalence relation is confirmed, we would get a corresponding quotient set $[X]$. Then we can define the triples $([X], [f_a], [T])$ as the quotient space which relates with R , where $[f_a]$ and $[T]$ correspond respectively to quotient attribute function of quotient set X and quotient structure. If we combine different quotient sets with the corresponding quotient spaces, we will get the world of different sizes for the problem. The purpose of quotient space theory is to study the relationship of the quotient spaces, the decomposition, comprehension and combination of the quotient space, and the reasoning between and within the quotient spaces. In this theoretical system, granulation, computing and processing are the basic problems.

In the concept lattice (definition in Section 4), thanks to $X \subseteq U$, $B^* \subseteq U$, $B \subseteq A$, $X^* \subseteq A$, so X, B^*, B and X^* are part of the whole, they can be regarded as the visual granules of formal context (U, A, I) . Therefore, it shows that the concept lattice is closely related to the visual granules X, B, X^*, B^* . This is the important reason when the concept lattice is seen as the granular computing. As the results of the current research, the concept lattice combined with the quotient space has not yet been reported.

3 PAPER OBJECTIVE

Quotient space and concept lattice are generally regarded as important researches on granular computing because they all can be described by abstract granules. Our objective is to provide an approach to organizing information for constructing concept lattice, complementing granular computing-based direct search. Our research will provide a way for formulization on granular computing.

This paper is organized as follows. In Section 4, we study the concept representation mechanisms involved in the lattice model of granular computing. They directly produce the granular quotient space. Section 5 gives the description of granular entropy techniques, including some novel formulas. It can be used to compute the minimal reduct. Section 6 explains the results of empirical study to apply proposed algorithm of constructing concept lattice. Sections 7 describes our future works and draw conclusions.

4 CONCEPT LATTICE AND QUOTIENT SPACE

Now, we first introduce the definition of the formal concept analysis [2].

A formal context is a triple (U, A, I) , where $U = \{u_1, u_2, \dots, u_n\}$. U is called the universe of discourse, a nonempty and finite set of objects. $A = \{a_1, a_2, \dots, a_m\}$ is a nonempty and finite set of attributes, and $I \subseteq U \times A$ is a binary relation between U and A with $(x, a) \in I$ indicating that the object x owns the attribute a .

For $X \subseteq U$ and $B \subseteq A$, let us define a pair of operators “*”: $X^* = \{a \in A | \forall x \in X, (x, a) \in I\}$, $B^* = \{x \in U | \forall a \in B, (x, a) \in I\}$. That is, X^* is the maximal family of the attributes that all the objects in X have in common and B^* is the maximal

family of the objects shared by all the attributes in B .

Definition 1 ([2]). Let (U, A, I) be a formal context. For $X \subseteq U$ and $B \subseteq A$, the ordered pair (X, B) is called a formal concept (or simply a concept) if it satisfies $X^* = B$ and $B^* = X$. Here, X^* and B^* are termed, respectively, as the extent and the intent of the formal concept (X, B) . The set of all formal concepts of (U, A, I) is denoted by $\kappa(U, A, I)$.

The fundamental theorem of FCA states that the set of all formal concepts on a given context with the ordering $(X_1, B_1) \leq (X_2, B_2)$ if and only if $X_1 \subseteq X_2$ is a complete lattice called the concept lattice, in which the infima and suprema are given by $\bigwedge_{i \in J} (X_i, Y_i) = \left(\bigcap_{i \in J} X_i, \left(\left(\bigcup_{i \in J} Y_i \right)^* \right)^* \right) = \left(\bigcap_{i \in J} X_i, \left(\bigcap_{i \in J} X_i \right)^* \right)$, $\bigvee_{i \in J} (X_i, Y_i) = \left(\left(\left(\bigcup_{i \in J} X_i \right)^* \right)^*, \bigcap_{i \in J} Y_i \right) = \left(\left(\bigcap_{i \in J} X_i \right)^*, \bigcap_{i \in J} Y_i \right)$.

We have following definition, when the attribute of concept lattice is not binary attributes $(0, 1)$ but the multi-valued attributes.

Definition 2 ([27, 26, 28]). A quadruple (U, A, V, f) is called an information system, where:

- U is a non-empty finite set of objects;
- A is a non-empty finite set of attributes;
- $V = \bigcup_{a \in A} V_a$, where V_a is a value-universe of the attribute a ;
- $f : U \times A \rightarrow V$ is an information function, such that for all $a \in A$ and $x \in U$ it holds that $f(x, a) \in V_a$.

If f is a total function, i.e. $f(x, a)$ is defined for all $x \in U$ and $a \in A$, then the information system is called complete; otherwise it is called incomplete.

We can study the so-called concept knowledge system if the power sets of objects and attributes meet some relations in the information system.

Definition 3 (Concept knowledge system [6]). A concept knowledge system can be defined as a four-tuple: (U, A, L, H) . Here U is a finite nonempty object set. A is a finite nonempty attribute set. $P(U)$ is the power set of U . $P(A)$ is the power set of A . $X_1, X_2 \subseteq U; B_1, B_2 \subseteq A$.

- $L : P(U) \rightarrow P(A)$ is called extension-intension operator if L satisfies $(L_1) L(\emptyset) = A; L(U) = \emptyset; (L_2)L(X_1 \cup X_2) = L(X_1) \cap L(X_2)$. $L(X)$ is a common attributes set of X .
- $H : P(A) \rightarrow P(U)$ is called intension-extension operator if H satisfies $(H_1) H(\emptyset) = U; H(A) = \emptyset; (H_2)H(B_1 \cup B_2) = H(B_1) \cap H(B_2)$. $H(B)$ is a common objects set of B .

Further, it meets $(LH) : H(L(X)) \supseteq X, L(H(B)) \supseteq B$.

Definition 4 ([6]). Let (U, A, L, H) be a concept knowledge system, $X \subseteq U$ and $B \subseteq A$. The pair (X, B) is called a concept, which is known as the granular concept if $X = H(B)$, $B = L(X)$. $\mathfrak{S}(U, A, L, H) = \{(X, B) | H(B) = X, L(X) = B\}$ is the set of all granular concepts.

The concept knowledge system is an extension of general information system. It shows that any object set and any attribute set can be constituted as a concept information granule and even a concept by iteratively computing. This process describes the procedure of human cognition step by step.

Definition 5 ([6]). Let (U, A, L, H) be the concept knowledge system, $X \subseteq U$ and $B \subseteq A$, the pair (X, B) is called a concept information granule if $X \subseteq H(B)$, $B \subseteq L(X)$.

Theorem 1 ([6]). Let (U, A, L, H) be a concept knowledge system. The knowledge system generated by formal context (U, A, I) is the same as (U, A, L, H) .

We can set up a problem space for the objective of our study. The aim of representing a problem at different granularities is to enable the computer to solve the same problem at different granularity hierarchically.

Definition 6 (Quotient space [18]). Let (X, f_a, T) be a problem space. Suppose that X represents the universe with the finest granular size. There is a coarse-granular universe denoted by $[X]$ that forms a new problem space $([X], [f_a], [T])$. $([X], [f_a], [T])$ is called a quotient space of (X, f_a, T) .

Quotient space theory is a new kind of mathematical tool to deal with problems based on different grain sizes. The problem here is with a triple (X, f, T) , where X is a set of objects, known as the universe, f is a function of X for the universe. $f(u)$ said attribute values u related problems when $u \in X$. The function $f(u)$ generally depends on the specific situation. T is made up of a subset of the universe X . It forms the topological structure [31]. For example, a given problem (X, f, T) , R is equivalence relation on X . So get the division $[X] = \{X_1, X_2, \dots, X_n\}$ of R about X , coupled with the topological T , we can get the quotient space $([X], [f], [T])$. The $[f]$ is the function of $[X]$ as the universe. For $X_i \in [X]$, function value $[f](X_i)$ is determined by values that f acts each element in the X , and the determination method to different problem may be different. $[T]$ is the quotient topology. Its generation method is as follows:

- Let $p : X \rightarrow [X]$ be a function from X to $[X]$, makes for $x \in X$, if $x \in X_i$, then $p(x) = X_i$. $[T] = \{Y | Y \subseteq [X] \text{ and } p^{-1}(Y) \in T\}$. Here $p^{-1}(Y)$ said the original image set of Y about p . It can be proved that $[T]$ is the topology on $[X]$ [31], called $[T]$ the quotient topology on $[X]$. Due to the element in the $[X]$ is the equivalence class, so the quotient space $([X], [f], [T])$ is closely related to the equivalence relation. Equivalence class is the part of X , can be seen as intuitive grains of problems (X, f, T) . It is natural that $[X]$ is determined by the intuitive

grain. So the structure of the quotient space $([X], [f], [T])$ embodies the grain to approximate the problems (X, f, T) . Since the different equivalence relation for different quotient space, so one of the most important aspects of the quotient space theory is the relation between quotient spaces, such as all the quotient space corresponding equivalent relations on X , and contain relations between equivalent relation form a lattice [30, 23].

Definition 7. Let G be a visual granule of the universe. If there is space $\langle U, Form(U) \rangle$ and n -formula ($n \geq 1$), $\varphi \in Form(U)$, such that $G = |\varphi|$, then G can be described in granular space $\langle U, Form(U) \rangle$.

Theorem 2 (Formal context to quotient space). Let (U, A, L, H) be a concept knowledge system generated by formal context (U, A, I) , $X \subseteq U$ and $B \subseteq A$. Then $([U], I, [A])$ constitutes quotient space. Granule $L(X)(\subseteq A)$ and $H(B)(\subseteq U)$ can be described in the quotient space.

Proof. Since $I \subseteq U \times A$, we can obtain $B = L(X) \subseteq A$ for $\forall X \subseteq U$, that is, $X^* = B, B^* = X$. There exists $X^* = B = L(X)$. So, I is f . $([U], I, [A])$ constitutes the quotient space. For $([U], I, [A])$, $[U]$ corresponds to granular space $\langle [U], Form([U]) \rangle$, where $Form([U])$ is the set of logical formulas on $[U]$.

If using Q to represent X , then $Q(x, a) \in Form([U])$ for $a \in A, x \in [U]$, and $Q(x, a)$ is an atomic formula on $[U]$. $Q(x, a) = \{x|x \in [U] \text{ and } \langle x, a \rangle \in Q\} = \{x|x \in [U] \text{ and } \langle x, a \rangle \in I\}$. Let $B = \{a_{i_1}, a_{i_2}, \dots, a_{i_r}\}$, $H(B) = \{x|x \in [U] \text{ and } \langle x, a \rangle \in I \text{ for } \forall a \in B\} = \{x|x \in [U] \text{ and } \langle x, a_{i_1} \rangle \in I\} \cap \{x|x \in [U] \text{ and } \langle x, a_{i_2} \rangle \in I\} \cap \dots \cap \{x|x \in [U] \text{ and } \langle x, a_{i_r} \rangle \in I\} = \{x|x \in [U] \text{ and } \langle x, a_{i_1} \rangle \in Q\} \cap \{x|x \in [U] \text{ and } \langle x, a_{i_2} \rangle \in Q\} \cap \dots \cap \{x|x \in [U] \text{ and } \langle x, a_{i_r} \rangle \in Q\} = Q(x, a_{i_1}) \cap Q(x, a_{i_2}) \cap \dots \cap Q(x, a_{i_r}) = Q(x, a_{i_1}) \wedge Q(x, a_{i_2}) \wedge \dots \wedge Q(x, a_{i_r})$. If we let $\varphi(x) = Q(x, a_{i_1}) \wedge Q(x, a_{i_2}) \wedge \dots \wedge Q(x, a_{i_r})$, $\varphi(x) \in Form([U])$, $H(B) = \varphi(x)$. Therefore, granular $H(B)(\subseteq U)$ can be described in the quotient space. Similarly, we can prove $L(X)$. \square

The attributes set A in (U, A, V, f) can be divided into several disjoint subsets with equivalence relation. Each subset is an equivalence class of an attribute. The equivalence classes that constitute quotient space form a lattice.

Theorem 3. Let (U, A, V, f) be an information system, $P(A)$ be a power set of A . Then a quotient space can be constituted by $B \in P(A)$ on A . And all quotient spaces and containing relations between the equivalence relations form a lattice.

Proof. Note that $B \in P(A)$. Then there exists a partition $U/R = \{[u]_R|u \in U\}$ determined by B 's equivalence relation R on U . Denoted by $[X]_B = \{[u]_R|u \in X\}$. Obviously, it forms a quotient space. If another $C \in P(A)$, we can see that $[X]_B \vee [Y]_C = [X]_B \cup [Y]_C$, $[X]_B \wedge [Y]_C = [X]_B \cap [Y]_C$. It is easy to know that they satisfy the commutative, associative and absorption law. $[\emptyset]_A$ and $[U]_A$ are the least element and the greatest element, respectively, so $(P(A), \wedge, \vee, [\emptyset]_A, [U]_A)$ is bounded lattice. \square

Because the quotient space is to consider both the object and the attribute, we study the concept information granule of the quotient space and their characteristics in this section.

For an information system (U, A, V, f) , by equivalence relation R of $B \subseteq A$ on U , we can granulate information $U/B = [B]_R = [v_1^B, v_2^B, \dots, v_k^B]_R$, where v_i^B are the possible values of attributes, and then obtain atomic quotient space $U/[\{v_i^B\}]_R = X_{R_i} \subseteq P(U)$, a finer quotient space.

U	$Time$	$Item$	$Location$
1	Jan	cosmetic	Beijing
2	Feb	foodstuff	Beijing
3	Jan	cosmetic	Shanghai
4	Feb	cosmetic	Shanghai
5	Mar	foodstuff	Wuhan
6	Mar	foodstuff	Wuhan

Table 1. An information system

For example, in Table 1, $U/(time, item) = [time, item]_R = \{\{1, 3\}, \{2\}, \{4\}, \{5, 6\}\}$, taking $v_1^B = \{Jan, cosmetic\}$ for $B = (time, item)$, we obtain that $U/[\{v_i^B\}]_R = U/[\{Jan, cosmetic\}]_R = \{\{1, 3\}\}$.

Theorem 4. An objects set of concept in an information system (U, A, V, f) must be included in quotient space of equivalence relation R . An atomic quotient space corresponds to a concept if it has not sub-quotient space.

Proof. For information system (U, A, V, f) , defining the operator by equivalence relation R on $P(A) \rightarrow P(U)$ and $P(U) \rightarrow P(A)$ can build concept knowledge system (U, A, L, H) . (X, B) is the concept if $X \subseteq U$ and $B \subseteq A$ satisfy $L(X) = B, H(B) = X$. Obviously, $X \in U/R_B$. Therefore, object set of the concept is contained in the classification of quotient space. For X_B , if there does not exist $X_{B'} \subset X_B$, it indicates that there is no $B \subset B'$, i.e. $L(X_B) = B, H(B) = X_B$, so (X_B, B) is a formal concept. \square

Example 1. The concept lattice of Table 1 is shown in Figure 1. It is easy to see that the equivalence classes of quotient space on equivalence relation R are present in the objects sets of concept lattice.

In fact, the quotient space is a kind of division. From the view of the covering, we have the following definition.

Definition 8. Given a formal context (U, A, I) , a relation is defined as follows: $R_s = \{(x, y) \in U \times U : \forall a \in \{x\}^*, yIa\}$. There is a set $R_c(x) = \{y | (x, y) \in R_s, y \in U\}$.

Obviously, $R_c(x) = (\{x\}^*)^*$.

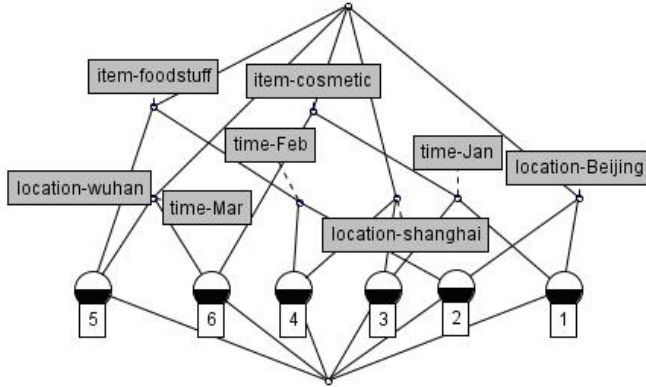


Figure 1. Concept lattice of Table 1

Definition 9. Let U be an object set. $I' = I \cap (U \times B)$, $B \subseteq A$ is a binary relation between U and B that satisfies $\forall x \in U, \forall a \in B, xI'a$. (U, B, I') is a sub-context of (U, A, I) .

Theorem 5. Suppose that (U, B, I') is a sub-context of the (U, A, I) , $B \subseteq A$. $(R_c(x), (R_c(x))^*)$, $\forall x \in U$ is a formal concept.

Proof. By the definition of $R_c(x)$, $(R_c(x), (R_c(x))^*)$ is a formal concept of (U, A, I) . And $(R_c(x)_B)^* = (R_c(x))^*$, $((R_c(x))^*_B)^* = (\{x\}^*_B)^*$, there is $(\{x\}^*_B)^* = (\{x\}^*)^* = R_c(x)$. So $(R_c(x), (R_c(x))^*)$ is a formal concept of the sub-context. \square

Theorem 6. Each formal concept on (U, A, I) can be generated by \bigvee operation of the formal concepts that are formed by R_c classified sets. If $\bigcup_{x \in U} R_c(x) = U$, then $\bigcup_{x \in U} (R_c(x), (R_c(x))^*) = \kappa(U, A, I)$.

Proof. By Theorem 5, $(R_c(x), (R_c(x))^*)$ is a concept of (U, A, I) . The intent of each concept generated by $\bigvee (R_c(x), (R_c(x))^*)$ is contained in $\{x\}^*$, that is, the intents of all concepts, whether in the the original context or sub-context, are included in $R_c(x)$'s attribute sets. The extent of concept on the sub-context is the extension on the original context, so the intent and extent obtained by \bigvee belong to the original context. All concepts on the original context can be formed by $(R_c(x), (R_c(x))^*)$. \square

5 CONCEPT GRANULAR DESCRIPTION BASED ON THE QUOTIENT SPACE

The quotient space provides a quick way of simplifying the complex system. But how to measure its effects? In this section, we will discuss granular entropy.

The entropy of a system as defined by Shannon [29], called information entropy, gives a measure of uncertainty about its actual structure. Unlike Shannon’s entropy, the proposed entropy can measure not only uncertainty in information systems, but also a granular classification. For information system (U, A, V, f) , a quotient space can be constituted by equivalence relation R of $B \subseteq A$. R can be seen as a map defined in granules based on the division function. So we can define the distribution function of the map and its information entropy.

Definition 10. Let (U, A, V, f) be an information system, $B \subseteq A$. R_B deduces quotient space $([u_i]_{R_B}, R, [A])$ with respect to U . The partition granule of (U, A, V, f) is defined by $X_i = [u_i]_{R_B} (i = 1, 2, \dots, m)$. $\|[u_i]_{R_B}\|$ is its granular value. Its granularity is denoted as $G(X_i) = \|X_i\|/|U|$.

In general, the value of $\|X_i\|$ is 1.

Theorem 7. For $(U, A, V, f), B \subseteq A, R_B \in P(A)$, quotient space $([u_i]_{R_B}, R, [A])$ has the following properties:

1. $\cup_{u \in U} [u]_{R_B} = U$;
2. $[u_i]_{R_B} \cap [u_j]_{R_B} = \emptyset$, when $[u_i]_{R_B} \neq [u_j]_{R_B}$.

Proof. It is easy to see that the results are true according to Definition 10. □

Definition 11 (Granular entropy). Let (U, A, V, f) be an information system. A granular entropy of U/R_B is defined as

$$E(R_B) = \sum_{i=1}^m G(X_i)/|U|.$$

For an information system (U, A, V, f) , $U/R_A = \{X_1, X_2, \dots, X_m\}$, Shannon defined an information entropy [29]: $H(A) = -\sum_{i=1}^m p_i \log_2 p_i, p_i = \frac{|X_i|}{|U|}$. Liang [32] put forward a new definition of information entropy: $E(A) = \sum_{i=1}^m \frac{|X_i||X_i^c|}{|U||U|} = \sum_{i=1}^m \frac{|X_i|}{|U|} (1 - \frac{|X_i|}{|U|})$, where X_i^c denotes the complement set of X_i , i.e., $X_i^c = U - X_i$. $\frac{|X_i|}{|U|}$ represents the probability of X_i within the universe U and $\frac{|X_i^c|}{|U|}$ is the probability of the complement set of X_i within the universe U . This entropy can measure not only uncertainty in information systems, but also fuzziness of a rough set and a rough classification.

$E(R_B)$ is mainly used to identify divided universe along with the attributes of monotonic increasing or decreasing in knowledge B . The attributes of knowledge B gradually increasing results in the division of B to universe finer, produced grain size smaller. $E(R_B)$ increased gradually, knowledge B identification gradually increases. The classification is more accurate. And vice versa. If the increase or decrease of attributes with B , the division B to the universe is at constant and $E(R_B)$ are the same.

Theorem 8. Let $S = (U, A, V, f)$ be an information system, $B, D \subseteq A$. The following results hold in two quotient space with respect to R_B and R_D .

1. $E(R_B) = \sum_{i=1}^m \|X_i\|/|U|^2$;
2. $E(R_{B \cup D}) = \sum_{i,j} \|X_i \cap X_j\|/|U|^2$.

Proof. It is proved by Definitions 10 and 11. □

Theorem 9. Let (U, A, V, f) be an information system, $B, D \subseteq A$. If $U/R_B = U/R_D$, then $E(R_B) = E(R_D)$.

Proof. Since $U/R_B = U/R_D$, R_B and R_D deduce same partition on universe U , that is, $[u_i]_{R_B} = [u_i]_{R_D} (i = 1, 2, \dots, m)$. They have the same quotient distribution. By the formula of the granular entropy, we can get $E(R_B) = E(R_D)$. □

This theorem states that knowledge have the same classification ability if they have same algebraic representations. However, the converse is not necessarily true.

Theorem 10. Let (U, A, V, f) be an information system, $B, D \subseteq A$. If $R_B \subseteq R_D \in P(A)$ and $E(R_B) = E(R_D)$, then $U/R_B = U/R_D$.

Proof. We assume that $U/R_B = \{X_1, X_2, \dots, X_m\}, U/(R_D \setminus R_B) = \{Y_1, Y_2, \dots, Y_m\}$. According to Definition 11 and Theorem 8 we can see that $E(R_B) = \sum_{i=1}^m \|X_i\|/|U|^2, E(R_B \cup (R_D \setminus R_B)) = \sum_{i,j} \|X_i \cap X_j\|/|U|^2$. $\|X_i\| = \sum_{j=1}^m \|X_i \cap X_j\|$ is available by $E(R_B) = E(R_D)$. Thus, there are two situations: $X_i \subset Y_j$ or that exists j_0 , so that $\|X_i \cap X_{j_0}\|, i.e. X_i \cap X_{j_0} = \emptyset$. However, $\cup_j Y_j = U$ and $\|X_i \cap X_{j_0}\| = 0$, this will not happen. Hence, $X_i \subset Y_j$ for any X_i . Thus, $U/R_B = U/R_D$. □

The result given in Theorem 10 provides a theoretical foundation for knowledge reduction based on the entropy.

Theorem 11. Let (U, A, V, f) be an information system, $B \subseteq A$. Then $b \in B$ is unnecessary if and only if $E(R_{B-\{b\}}) = E(R_B)$.

Proof. The necessity is proved by Theorem 9. The sufficiency is proved by Theorem 10. □

Theorem 11 shows that adding an unnecessary attribute to a subset will not change the size of granule.

Corollary 1. Let (U, A, V, f) be an information system, $B \subseteq A$. Then $b \in B$ is unnecessary if and only if $E(R_{B-\{b\}}) > E(R_B)$.

Corollary 2. Let (U, A, V, f) be an information system, $B \subseteq A$. Then knowledge R_B is independent if and only if $E(R_{B-\{b\}}) > E(R_B), b \in B$.

Theorem 12 (Reduct). Let (U, A, V, f) be an information system. $B \subseteq A$ is a reduct of knowledge R_B if and only if

1. $E(R_A) = E(R_B)$.
2. $E(R_{B-\{b\}}) < E(R_B)$ for $\forall b \in B$.

Proof. Suppose that B is a reduct of A , i.e., there is $U/R_A = U/R_B$ such that $E(R_A) = E(R_B)$ by Theorem 9. If $B - b \subseteq B$, exist $E(R_{B-\{b\}}) \geq E(R_B)$, then there is at least $U/R_{B-\{b\}} = U/R_B$ by Theorem 10. This contradicts with that B is a reduct. To show the converse, suppose that there is $E(R_A) = E(R_B)$, we can get $U/R_A = U/R_A$ by Theorem 10. Since $E(R_{B-\{b\}}) < E(R_B)$ for $\forall b \in B$, B is a reduct by Corollary 2. □

Example 2. An information system $S = (U, A, V, f)$, as shown in Table 2, where $U = \{1, 2, \dots, 5\}$, $A = \{a, b, c, d, e\}$. Seek out a minimum reduct of A .

U	a	b	c	d	e
1	1	0	2	1	0
2	0	0	1	2	1
3	2	0	2	2	2
4	0	0	2	2	2
5	1	1	2	1	0

Table 2. An information system

By Theorem 8, $\{a\}^* = (\{1, 5\}, \{2, 4\}, \{3\})$, $E_{R_{\{a\}^*}} = 3/25$, $\{b\}^* = (\{1, 2, 3, 4\}, \{5\})$, $E_{R_{\{b\}^*}} = 2/25$, $\{c\}^* = (\{1, 3, 4, 5\}, \{2\})$, $E_{R_{\{c\}^*}} = 2/25$, $\{d\}^* = (\{1, 5\}, \{2, 3, 4\})$, $E_{R_{\{d\}^*}} = 2/25$, $\{e\}^* = (\{1, 5\}, \{2\}, \{3, 4\})$, $E_{R_{\{e\}^*}} = 3/25$, are calculated, and then get $E_{R_A} = 5/25$.

Similarly, we can get $E_{R_{\{a,b\}}} = 4/25$, $E_{R_{\{a,b,c\}}} = 5/25$, $E_{R_{\{a,b,c,d\}}} = 5/25$, $E_{R_{\{a,c\}}} = 4/25$, $E_{R_{\{a,d\}}} = 3/25$, $E_{R_{\{a,e\}}} = 4/25$, $E_{R_{\{b,c\}}} = 3/25$, $E_{R_{\{b,d\}}} = 3/25$, $E_{R_{\{b,e\}}} = 4/25$, $E_{R_{\{c,d\}}} = 3/25$, $E_{R_{\{c,e\}}} = 3/25$, $E_{R_{\{d,e\}}} = 3/25$, $E_{R_{\{a,b,d\}}} = 4/25$, $E_{R_{\{a,b,e\}}} = 5/25$, $E_{R_{\{a,c,d\}}} = 4/25$, $E_{R_{\{a,c,e\}}} = 4/25$, $E_{R_{\{a,d,e\}}} = 4/25$, $E_{R_{\{b,c,d\}}} = 4/25$, $E_{R_{\{b,c,e\}}} = 4/25$, $E_{R_{\{b,d,e\}}} = 4/25$, $E_{R_{\{c,d,e\}}} = 3/25$, $E_{R_{\{a,b,c,e\}}} = 5/25$, $E_{R_{\{a,b,c,d\}}} = 5/25$, $E_{R_{\{a,b,d,e\}}} = 5/25$, $E_{R_{\{b,c,d,e\}}} = 4/25$.

It is easy to see that $E_{R_{\{a,b,e\}}} = E_{R_{\{a,b,c\}}} = 5/25 = E_{R_A}$, so, $\{a, b, c\}$ and $\{a, b, e\}$ are the minimum reduct.

6 CONSTRUCTING A CONCEPT LATTICE BASED ON QUOTIENT SPACE

Quotient space allows us to construct concept lattices on different granules. In practical applications, it is not necessary to construct the concept lattice of all granules and merge them to get the original concept lattice. By selecting a few suitable granules, the original concept lattice can be obtained by merging.

Example 3. The formal context in Table 3 is a minor revision of the famous example, a film “Living Beings and Water” [2]. Since we require all the formal contexts

in this paper are canonical, we delete the attribute a (water) from the original formal context. The objects are living beings mentioned in the film and are denoted by $U = \{1, 2, 3, 4, 5, 6, 7, 8\}$, where 1 is leech, 2 is bream, 3 is frog, 4 is dog, 5 is spike-weed, 6 is reed, 7 is bean, and 8 is maize. And the attributes in $A = \{b, c, d, e, f, g, h, i\}$ are the properties which the film emphasizes: b : lives in water, c : lives on land, d : needs chlorophyll to produce food, e : two seed leaves, f : one seed leaf, g : can move around, h : has limbs, and i : suckles its offspring.

U	b	c	d	e	f	g	h	i
1	×					×		
2	×					×	×	
3	×	×				×	×	
4		×				×	×	×
5	×		×		×			
6	×	×	×		×			
7		×	×	×				
8		×	×		×			

Table 3. Living beings and water (U, A, I)

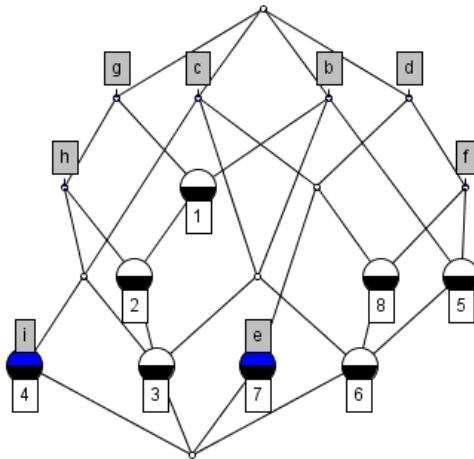


Figure 2. Concept lattice of Living Beings and Water $\kappa(U, A, I)$

The classified sets: $R_c(1) = \{1, 2, 3\}, R_c(2) = \{2, 3\}, R_c(3) = \{3\}, R_c(4) = \{4\}, R_c(5) = \{5, 6\}, R_c(6) = \{6\}, R_c(7) = \{7\}, R_c(8) = \{6, 8\}$.

The formal concepts determined by classified set $R_c(1) = \{1, 2, 3\}$ are: $(U, \emptyset), (\{3\}, \{b, c, g, h\}), (\{2, 3\}, \{b, g, h\}), (\{1, 2, 3\}, \{b, g\}), (\{3, 4\}, \{c, g, h\}), (\{2, 3, 4\}, \{g, h\}), (\{1, 2, 3, 4\}, \{g\}), (\{3, 6\}, \{b, c\}), (\{3, 4, 6, 7, 8\}, \{c\}), (\{1, 2, 3, 5, 6\}, \{b\})$.

$R_c(4) = \{4\}$: (U, \emptyset) , $(\{4\}, \{c, g, h, i\})$, $(\{3, 4\}, \{c, g, h\})$, $(\{2, 3, 4\}, \{g, h\})$, $(\{1, 2, 3, 4\}, \{g\})$, $(\{3, 4, 6, 7, 8\}, \{c\})$.

$R_c(5) = \{5, 6\}$: (U, \emptyset) , $(\{6\}, \{b, c, d, f\})$, $(\{5, 6\}, \{b, d, f\})$, $(\{6, 8\}, \{c, d, f\})$, $(\{6, 7, 8\}, \{c, d\})$, $(\{5, 6, 8\}, \{d, f\})$, $(\{5, 6, 7, 8\}, \{d\})$, $(\{1, 2, 3, 5, 6\}, \{b\})$, $(\{3, 4, 6, 7, 8\}, \{c\})$.

$R_c(7) = \{7\}$: (U, \emptyset) , $(\{7\}, \{c, d, e\})$, $(\{6, 7, 8\}, \{c, d\})$, $(\{5, 6, 7, 8\}, \{d\})$, $(\{3, 4, 6, 7, 8\}, \{c\})$.

$R_c(8) = \{6, 8\}$: (U, \emptyset) , $(\{6\}, \{b, c, d, f\})$, $(\{5, 6\}, \{b, d, f\})$, $(\{6, 8\}, \{c, d, f\})$, $(\{6, 7, 8\}, \{c, d\})$, $(\{5, 6, 8\}, \{d, f\})$, $(\{5, 6, 7, 8\}, \{d\})$, $(\{1, 2, 3, 5, 6\}, \{b\})$, $(\{3, 4, 6, 7, 8\}, \{c\})$.

Since $R_c(1) \cup R_c(4) \cup R_c(5) \cup R_c(7) \cup R_c(8) = U$, $\cup(R_c(x), L(R_c(x)))_{x \in \{1, 4, 5, 7, 8\}} = \kappa(U, A, I)$.

6.1 Algorithm

Main idea: Concept lattice is to be constructed in the simplified formal context. This concept lattice is a one-way homomorphism to the original one. The simplified concept lattice can be extended to restore according to the need.

Input: (U, A, I) where $U = \{u_1, u_2, \dots, u_{|U|}\}$, $A = \{a_1, a_2, \dots, a_{|A|}\}$

Output: Simplified (U', A', I') //To restore concept lattice of the original formal context by it.

```

01 For  $i = 1$  to  $|U|$  begin
02   For  $j = 1$  to  $|A|$  begin
03     If  $(u_i, a_j) \in I$ 
04       then  $(u_i)_{a_j}^* \leftarrow a_j$ 
05     else  $(u_i)_{a_j}^* \leftarrow \emptyset$ 
06   End
07 End
08 For  $i = 1$  to  $|A|$  begin
09   For  $j = 1$  to  $|U|$  begin
10     If  $(u_i, a_j) \in I$ 
11       then  $(a_i)_{u_j}^* \leftarrow u_j$ 
12     else  $(a_i)_{u_j}^* \leftarrow \emptyset$ 
13   End
14 End
15 For  $i = 1$  to  $|U|$  begin
16   For  $k = 1$  to  $|U|$  begin
17      $D_i \leftarrow \emptyset$ 
18     If  $(u_i)_{a_k}^* \supseteq (u_j)_{a_k}^*$ 
20       then  $D_i \leftarrow (u_i)_{a_k}^* \cup (u_j)_{a_k}^*$ 
21   End
22 End
23 For  $i = 1$  to  $|A|$  begin

```

24 For $i = 1$ to $|A|$ begin
 25 $E_i \leftarrow \emptyset$
 26 If $(a_i)_{u_i}^* \supseteq (a_j)_{u_i}^*$
 27 then $E_i \leftarrow (a_i)_{u_i}^* \cup (a_j)_{u_i}^*$
 28 End
 29 End
 30 Reduced formal context $(D, E, I) = (U', A', I')$ with quotient space division.
 31 To restore concept lattice $\kappa(U, A, I)$ of the original formal context.

Example 4. Data table is shown in Table 4. It is a classic example of formal concept analysis [2].

		1	2	3	4	5	6	7	8	9	10	11
		fur(hair)	feathers	scales can fly	lives in water	lay eggs	produces milk has a backbone	warm-blooded	cold-blooded domestic			
1	Bat	×			×		×	×	×			
2	Bear	×					×	×	×			
3	Cat	×					×	×	×			×
4	Chicken		×				×	×	×	×		×
5	Dog	×					×	×	×	×		×
6	Dolphin	×			×		×	×	×			
7	Elephant	×					×	×	×			
8	Frog				×	×	×	×			×	
9	Hawk		×	×		×	×	×				
10	Housefly	×		×		×					×	
11	Owl		×	×		×	×	×				
12	Sea lion	×			×		×	×	×			
13	Snake		×				×	×			×	
14	Spider	×					×				×	
15	Turtle		×				×	×			×	

Table 4. An illustrative example of a formal context of animals and their attributes

In order to distinguish between objects and attributes, we attach a letter subscript for each number. The uppercase is object, the lowercase is attribute.

Step 1.

$$\begin{aligned}
\{1_f\}^* &= \{1_B, 2_B, 3_C, 5_D, 6_D, 7_E, 10_H, 12_S, 14_S\}, \\
\{2_f\}^* &= \{4_C, 9_H, 11_O\}, \\
\{3_s\}^* &= \{13_S, 15_T\}, \\
\{4_f\}^* &= \{1_B, 9_H, 10_H, 11_O\}, \\
\{5_w\}^* &= \{6_D, 8_F, 12_S\}, \\
\{6_e\}^* &= \{4_C, 8_F, 9_H, 10_H, 11_O, 13_S, 14_S, 15_T\}, \\
\{7_m\}^* &= \{1_B, 2_B, 3_C, 5_D, 6_D, 7_E, 12_S\}, \\
\{8_b\}^* &= \{1_B, 2_B, 3_C, 4_C, 5_D, 6_D, 7_E, 8_F, 9_H, 11_O, 12_S, 13_S, 15_T\}, \\
\{9_w\}^* &= \{1_B, 2_B, 3_C, 4_C, 5_D, 6_D, 7_E, 9_H, 11_O, 12_S\}, \\
\{10_c\}^* &= \{4_C, 8_F, 10_H, 13_S, 14_S, 15_T\}, \\
\{11_d\}^* &= \{3_C, 4_C, 5_D\}. \\
\{6_e\}^* - \{10_c\}^* &= \{4_C, 9_H, 11_O\}, \\
\{10_c\}^* - \{3_s\}^* &= \{8_F, 10_H, 14_S\}.
\end{aligned}$$

That is,

$$\begin{aligned}
\{3_s\}^* &\subseteq \{8_b\}^*, \\
\{3_s\}^* &\subseteq \{10_c\}^* \subseteq \{6_e\}^*, \\
\{11_d\}^* &\subseteq \{9_w\}^* \subseteq \{8_b\}^*, \\
\{7_m\}^* &\subseteq \{9_w\}^* \subseteq \{8_b\}^*, \\
\{2_f\}^* &\subseteq \{9_w\}^*, \\
\{2_f\}^* &\subseteq \{6_e\}^*, \\
\{5_w\}^* &\subseteq \{8_b\}^*.
\end{aligned}$$

So, $\{1_f\}^*$, $\{4_f\}^*$, $\{6_e\}^*$, $\{8_b\}^*$ are more coarse granules.

$$\begin{aligned}
\{1_B\}^* &= \{1_f, 4_c, 7_m, 8_b, 9_w\}, \\
\{2_B\}^* &= \{1_f, 7_m, 8_b, 9_w\}, \\
\{3_C\}^* &= \{1_f, 7_m, 8_b, 9_w, 11_d\}, \\
\{4_C\}^* &= \{2_f, 6_e, 8_b, 9_w, 11_d\}, \\
\{5_D\}^* &= \{1_f, 7_m, 8_b, 9_w, 11_d\}, \\
\{6_D\}^* &= \{1_f, 5_w, 7_m, 8_b, 9_w\}, \\
\{7_E\}^* &= \{1_f, 7_m, 8_b, 9_w\}, \\
\{8_F\}^* &= \{5_w, 6_e, 8_b, 10_c\}, \\
\{9_H\}^* &= \{2_f, 4_f, 6_e, 8_b, 9_w\}, \\
\{10_H\}^* &= \{1_f, 4_f, 6_e, 10_c\}, \\
\{11_O\}^* &= \{2_f, 4_f, 6_e, 8_b, 9_w\},
\end{aligned}$$

$$\begin{aligned} \{12_S\}^* &= \{1_f, 5_w, 7_m, 8_b, 9_w\}, \\ \{13_S\}^* &= \{3_s, 6_e, 8_b, 10_c\}, \\ \{14_S\}^* &= \{1_f, 6_e, 10_c\}, \\ \{15_S\}^* &= \{3_s, 6_e, 8_b, 10_c\}. \end{aligned}$$

Similarly,

$$\begin{aligned} \{2_B\}^* &= \{7_E\}^* \subseteq \{3_C\}^* \subseteq \{1_B\}^*, \\ \{7_E\}^* &\subseteq \{12_S\}^* = \{6_D\}^*, \\ \{3_C\}^* &= \{5_D\}^*, \\ \{6_D\}^* &= \{12_S\}^*, \\ \{14_S\}^* &\subseteq \{10_H\}^*, \\ \{13_S\}^* &= \{15_T\}^*, \\ \{9_H\}^* &= \{11_O\}^*. \end{aligned}$$

$\{1_B\}^*, \{4_C\}^*, \{5_D\}^*, \{6_D\}^*, \{8_F\}^*, \{9_H\}^*, \{10_H\}^*, \{13_S\}^*$ are coarse granules.

Step 2. A simplified formal context (U', A', I') constructed by quotient space is shown in Table 5.

		1	4	6	8
		fur(hair) can fly lay eggs has a backbone			
1	Bat	×	×	×	
4	Chicken			×	×
5	Dog	×		×	
6	Dolphin	×		×	
8	Frog			×	×
9	Hawk		×	×	×
10	Housefly	×	×	×	
13	Snake			×	×

Table 5. A simplified formal context (U', A', I')

Concept lattice $\kappa'(U, A, I)$ of the simplified formal context is built as shown in Figure 3. It is easy to verify, $E_{R_{\{1_f, 4_f, 6_e, 8_b\}}} = 4/225$, $\{1_f, 4_f, 6_e, 8_b\}$ is the minimum reduct.

Step 3. According to (1), $\{3_s\}^* \subseteq \{10_c\}^* \subseteq \{6_e\}^*$, the concept that its attribute contains 6_e can be added 10_c but not objects $4_C, 9_H, 10_H$. $(\{10_H\}, \{1_f, 4_f, 6_e, 10_c\})$ is deduced from $(\{10_H\}, \{1_f, 4_f, 6_e\})$. $\{11_d\}^* \subseteq \{9_w\}^* \subseteq \{8_b\}^*$: $(\{1_B, 9_H\},$

$$\{4_f 8_b\} \Rightarrow (\{1_B, 9_H\}, \{4_f, 8_b, 9_w\}). \{7_m\}^* \subseteq \{9_w\}^* \subseteq \{8_b\}^*: (\{1_B\}, \{1_f, 4_f, 8_b\}) \\ \Rightarrow (\{1_B\}, \{1_f, 4_f, 8_b, 7_m, 9_w\}).$$

According to (2), $\{2_B\}^* = \{7_E\}^* \subseteq \{1_B\}^*$, $\{6_D\}^* = \{12_S\}^*$, $\{9_H\}^* = \{11_O\}^*$, $\{13_S\}^* = \{15_T\}^*$: $(\{1_B, 4_C, 5_D, 6_D, 8_F, 9_H, 13_S\}, \{8_b\}) \Rightarrow (\{1_B, 2_B, 4_C, 5_D, 6_D, 7_E, 8_F, 9_H, 11_O, 12_S, 13_S, 15_T\}, \{8_b\})$. $\{9_H\}^* = \{11_O\}^*$: $(\{1_B, 9_H, 10_H\}, \{4_f\}) \Rightarrow (\{1_B, 9_H, 10_H, 11_O\}, \{4_f\})$. $\{9_H\}^* = \{11_O\}^*$, $\{13_S\}^* = \{15_T\}^*$: $(\{4_C, 8_F, 9_H, 13_S\}, \{6_e, 8_b\}) \Rightarrow (\{4_C, 8_F, 9_H, 11_O, 12_S, 13_S, 15_T\}, \{6_e, 8_b\})$. $\{9_H\}^* = \{11_O\}^*$, $\{13_S\}^* = \{15_T\}^*$, $\{14_S\}^* \subseteq \{10_H\}^*$: $(\{4_C, 8_F, 9_H, 10_H, 13_S\}, \{6_e\}) \Rightarrow (\{4_C, 8_F, 9_H, 10_H, 11_O, 13_S, 14_S, 15_T\}, \{6_e\})$. $\{3_C\}^* = \{5_D\}^*$, $\{6_D\}^* = \{12_S\}^*$, $\{14_S\}^* \subseteq \{10_H\}^*$: $(\{1_B, 5_D, 6_D, 10_H\}, \{1_f\}) \Rightarrow (\{1_B, 3_C, 5_D, 6_D, 10_H, 12_S, 14_S\}, \{1_f\})$. $\{9_H\}^* = \{11_O\}^*$: $(\{9_H, 10_H\}, \{4_f, 6_e\}) \Rightarrow (\{1_B, 9_H, 11_O\}, \{4_f, 6_e\})$. $\{9_H\}^* = \{11_O\}^*$, $\{9_w\}^* \subseteq \{8_b\}^*$: $(\{1_B, 9_H\}, \{4_f, 8_b\}) \Rightarrow (\{1_B, 9_H, 11_O\}, \{4_f, 8_b, 9_w\})$. $\{2_B\}^* = \{7_E\}^* \subseteq \{1_B\}^*$, $\{14_S\}^* \subseteq \{10_H\}^*$: $(\{1_B, 10_H\}, \{1_f, 4_f, 6_e\}) \Rightarrow (\{1_B, 10_H\}, \{1_f, 4_f, 6_e\})$. $\{2_B\}^* = \{7_E\}^* \subseteq \{1_B\}^*$, $\{3_C\}^* = \{5_D\}^*$, $\{6_D\}^* = \{12_S\}^*$, $\{7_m\}^* \subseteq \{9_w\}^* \subseteq \{8_b\}^*$: $(\{1_B, 5_D, 6_D\}, \{1_f, 8_b, 9_w\}) \Rightarrow (\{1_B, 2_B, 3_C, 5_D, 6_D, 7_E, 12_S\}, \{1_f, 7_m, 8_b, 9_w\})$. $\{9_H\}^* = \{11_O\}^*$, $\{2_f\}^* \subseteq \{6_e\}^*$, $\{9_w\}^* \subseteq \{8_b\}^*$: $(\{9_H\}, \{4_f, 6_e, 8_b\}) \Rightarrow (\{9_H, 11_O\}, \{2_f, 4_f, 6_e, 9_w\})$.

In addition, according to (1), we can obtain the following results. $\{3_s\}^* \cap \{6_e\}^* = \{3_s\}^* \cap \{8_b\}^* = \{3_s\}^* \cap \{10_c\}^* = \{13_S, 15_T\} \Rightarrow (\{13_S, 15_T\}, \{3_s, 6_e, 8_b, 10_c\})$. $\{6_e\}^* \cap \{10_c\}^* = \{8_F, 10_H, 13_S, 14_S, 15_T\} \Rightarrow (\{8_F, 10_H, 13_S, 14_S, 15_T\}, \{6_e, 10_c\})$. $\{8_b\}^* \cap \{9_w\}^* = \{1_B, 2_B, 3_C, 4_C, 5_D, 6_D, 7_E, 9_H, 11_O, 12_S\} \Rightarrow (\{1_B, 2_B, 3_C, 4_C, 5_D, 6_D, 7_E, 9_H, 11_O, 12_S\}, \{8_b, 9_w\})$. $\{8_b\}^* \cap \{11_d\}^* = \{9_w\}^* \cap \{11_d\}^* = \{3_C, 4_C, 5_D\} \Rightarrow (\{3_C, 4_C, 5_D\}, \{8_b, 9_w, 11_d\})$. $\{5_w\}^* \cap \{8_b\}^* = \{6_D, 8_F, 12_S\} \Rightarrow (\{6_D, 8_F, 12_S\}, \{5_w, 8_b\})$. $\{2_f\}^* \cap \{6_e\}^* = \{2_f\}^* \cap \{8_b\}^* = \{4_C, 9_H, 11_O\} \Rightarrow (\{4_C, 9_H, 11_O\}, \{2_f, 6_e, 8_b\})$.

By (2), we get that $\{3_C\}^* \cap \{5_D\}^* = \{1_f, 7_m, 8_b, 9_w, 11_d\} \Rightarrow (\{3_C, 5_D\}, \{1_f, 7_m, 8_b, 9_w, 11_d\})$. $\{6_D\}^* \cap \{12_S\}^* = \{1_f, 5_w, 7_m, 8_b, 9_w\} \Rightarrow (\{6_D, 12_S\}, \{1_f, 5_w, 7_m, 8_b, 9_w\})$. $\{12_S\}^* \cap \{14_S\}^* = \{1_f, 6_e, 10_c\} \Rightarrow (\{12_S, 14_S\}, \{1_f, 6_e, 10_c\})$.

In the new context, object 5, 6, and 4, 8, 13 have the same attributes respectively, so we should consider the possibility that these objects form concepts. $\{5_D\}^* \cap \{6_D\}^* = \{1_f, 7_m, 8_b, 9_w\} \Rightarrow (\{1_B, 2_B, 3_C, 5_D, 6_D, 7_E, 12_S\}, \{1_f, 7_m, 8_b, 9_w\})$. The concept already exists. At the same time, $\{3_C\}^* = \{5_D\}^*$, $\{6_D\}^* = \{12_S\}^*$, object 5 and 6 cannot construct the concept. $\{4_C\}^* \cap \{8_F\}^* = \{4_C\}^* \cap \{13_S\}^* = \{6_e, 8_b\} \Rightarrow (\{4_C, 8_F, 9_H, 11_O, 13_S, 15_T\}, \{6_e, 8_b\})$. The concept already exists. $\{13_S\}^* = \{15_T\}^*$, object 13 cannot construct the concept. $\{13_S\}^* = \{15_T\}^*$, $\{8_F\}^* \cap \{13_S\}^* = \{6_e, 8_b, 10_c\} \Rightarrow (\{8_F, 13_S, 15_T\}, \{6_e, 8_b, 10_c\})$. Finally, calculate the two single object concepts, $(\{4_C\}, \{2_f, 6_e, 8_b, 9_w, 11_d\})$, $(\{8_F, \{5_w, 6_e, 8_b, 10_c\})$.

Step 4. To construct the concept lattice $\kappa(U, A, I)$ as shown in Figure 4.

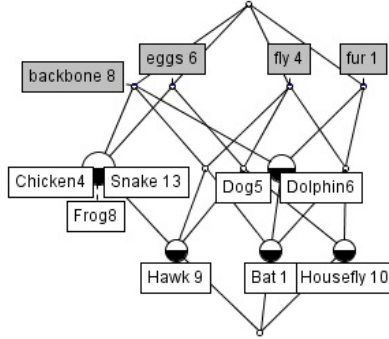


Figure 3. Concept lattice $\kappa'(U, A, I)$

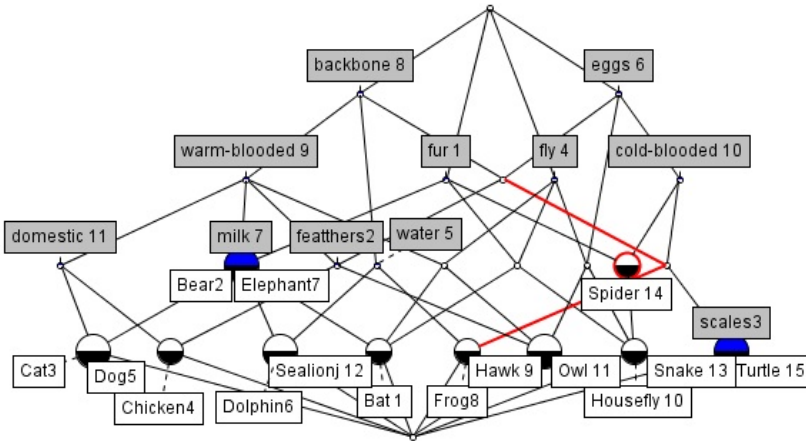


Figure 4. Concept lattice $\kappa(U, A, I)$

6.2 Experiments

Denote $N = \min\{|U|, |A|\}$, we know the time complexity of Step 1 in Algorithms is $O(N^2)$. So we can get two matters as follows.

1. The time complexity of algorithm is $O(4N^2)$.
2. Suppose that Algorithm will be terminated in the k^{th} step; then the time complexity of Algorithm is $O\left(\sum_{i=1}^{i=k} (N \times k)\right)$. We can easily get $O\left(\sum_{i=1}^{i=k} (N \times k)\right) \leq O(4N^2)$.

To verify the effectiveness of the algorithm, we compared it with Godin algorithm [24] (Basic Incremental Update Algorithms) using a server that contains

a Intel® Core™ i5-4590 3.30 GHz CPU, 4 GB memory, Windows 7 operating system and Visual C++ 6.0. The experimental data is random. The number of attributes is set to 20. The number of objects is increased from 50 to 1050. Change interval is 50. Figure 5 shows calculation results of two kinds of algorithms.

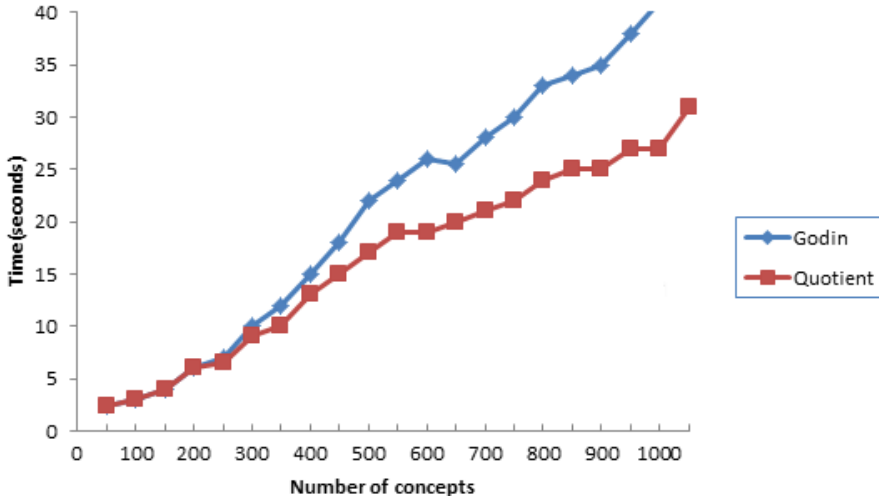


Figure 5. Comparison of the proposed algorithm with the Godin algorithm

In Figure 5, the abscissa denotes the number of formal concept. The ordinate is time to calculate the concept lattice. As it can be seen, the number is less than 200, the difference is not obvious. The proposed algorithm is better when the number becomes larger. It is proved that the proposed algorithm is effective.

It should be noted that the proposed algorithm is based on the quotient space granules. The algorithm has no obvious advantage in speed, if the information system can only be divided into a finer granule. It is, in fact, that the existing data space is abstracted into a “coarse grains” one. Thereby, the dimension of data is reduced, also simplifies the concept lattice. This simplified process can be thought of as hidden knowledge at different level, and it is suited to the needs of the problem analysis. The comparison of speed, in a sense, is not the most important.

7 CONCLUSION AND FUTURE WORK

This paper introduces concept information granule, granular entropy and quotient space into concept lattice research, and presents a unified research model for expansion and reduction of concept lattice in different granulations, and provides a detailed description of this overall process. In this model, it mainly obtains conclusions as follows:

1. A concept information granule is provided for the concept lattice research. As the knowledge in basic level, the concept information granule not only offers a uniform technology for concept learning on the whole, but it is also convenient for knowledge sharing and reuse in different levels;
2. The granular quotient space is introduced to lattice building research, which can overcome the impact on the application of FCA caused by the time complexity and space complexity problem to some extent, it helps to find useful information and avoids users getting lost in the complex information;
3. A new granular entropy between concepts is given in different granulations, which can help experts judge relations except for inheritance relation and measure the degree of reduction of the context.

Although the FCA based on granular quotient space proposed in this paper is only a starting point and a lot of subsequent study is needed, it offers a new way or guideline for the concept lattice reduction. How to combine concept lattice with quotient space more rationally and reduce human judgments is one focus of our research in the future.

REFERENCES

- [1] DAI, J.—SONG, J.—WANG, G.: A Rapid Concept Granules Extraction Algorithm Based on Cloud Model. *Journal of Computational Information Systems*, Vol. 8, 2012, No. 20, pp. 8413–8420.
- [2] GANTER, B.—WILLE, R.: *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag New York, Inc., 1999.
- [3] PAL, S. K.—MEHER, S. K.: Natural Computing: A Problem Solving Paradigm with Granular Information Processing. *Applied Soft Computing*, Vol. 13, 2013, No. 9, pp. 3944–3955.
- [4] POELMANS, J.—ELZINGA, P.—VIAENE, S.—DEDENE, G.: Formal Concept Analysis in Knowledge Discovery: A Survey. In: Croitoru, M., Ferré, S., Lukose, D. (Eds.): *Conceptual Structures: From Information to Intelligence (ICCS 2010)*. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 6208, 2010, pp. 139–153.
- [5] QIAN, Y. H.—ZHANG, H.—LI, F. J.—HU, Q. H.—LIANG, J. Y.: Set-Based Granular Computing: A Lattice Model. *International Journal of Approximate Reasoning*, Vol. 55, 2014, No. 3, pp. 834–852.
- [6] QIU, G. F.—CHEN, J.: Concept Knowledge System and Concept Information Granular Lattice. *Chinese Journal of Engineering Mathematics*, Vol. 22, 2005, No. 6, pp. 963–969.
- [7] QIU, G. F.—MA, J. M.—YANG, H. Z.—ZHANG, W. X.: A Mathematical Model for Concept Granular Computing Systems. *Science China Information Sciences*, Vol. 53, 2010, No. 7, pp. 1397–1408.

- [8] DING, S. F.—LI, J. Y.—XU, L.—QIAN, J.—ZHAO, X. W.—JIN, F. F.: Research Progress of Granular Computing (GrC). *International Journal of Digital Content Technology and Its Applications*, Vol. 5, 2011, No. 1, pp. 162–172.
- [9] WANG, L. D.—LIU, X. D.—WANG, X.: AFS-Based Formal Concept Analysis within the Logic Description of Granules. In: Yao, J. et al. (Eds.): *Rough Sets and Current Trends in Computing (RSCTC 2012)*. Springer, Berlin, Heidelberg, *Lecture Notes in Computer Science*, Vol. 7413, 2012, pp. 323–331.
- [10] WILLE, R.: Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts. In: Rival, I. (Ed.): *Ordered Sets*. Springer, Dordrecht, *NATO Advanced Study Institutes Series (Series C – Mathematical and Physical Sciences)*, Vol. 83, 2009, pp. 445–470.
- [11] WU, W. Z.: Attribute Granules in Formal Contexts. In: An, A., Stefanowski, J., Ramanna, S., Butz, C. J., Pedrycz, W., Wang, G. (Eds.): *Rough Sets, Fuzzy Sets, Data Mining and Granular Computing (RSFDGrC 2007)*. Springer, Berlin, Heidelberg, *Lecture Notes in Computer Science*, Vol. 4482, 2007, pp. 395–402.
- [12] WU, W. Z.—LEUNG, Y.—MI, J. S.: Granular Computing and Knowledge Reduction in Formal Contexts. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 21, 2009, No. 10, pp. 1461–1474.
- [13] WEIHUA, X. U.—LIU, S. H.: Granularity Representation of Knowledge in Information System Based on General Binary-Relation. *Computer Engineering and Applications*, Vol. 109, 2011, No. 2, pp. 619–630.
- [14] YAO, Y. Y.: Concept Lattices in Rough Set Theory. *IEEE Annual Meeting of the Fuzzy Information*, 2004. *Processing NAFIPS'04*, Vol. 2, 2004, pp. 796–801.
- [15] YAO, Y. Y.—ZHANG, N.—MIAO, D. Q.—XU, F. F.: Set-Theoretic Approaches to Granular Computing. *Fundamenta Informaticae*, Vol. 115, 2012, No. 2-3, pp. 247–264.
- [16] ZHANG, B.—ZHANG, L.: *Theory and Applications of Problem Solving*. Elsevier Science Inc. Press, pp. 770, 1992.
- [17] ZHANG, L.—ZHANG, B.: *Quotient Space Based Problem Solving: A Theoretical Foundation of Granular Computing*. Qinghua University Press, 2014.
- [18] ZHANG, L.—ZHANG, B.: The Quotient Space Theory of Problem Solving. In: Wang, G., Liu, Q., Yao, Y., Skowron, A. (Eds.): *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC 2003)*. Springer, Berlin, Heidelberg, *Lecture Notes in Computer Science*, Vol. 2639, 2003, pp. 11–15.
- [19] ZHANG, Q.—XING, Y.: Formal Concept Analysis Based on Granular Computing. *Journal of Computational Information Systems*, Vol. 6, 2010, No. 7, pp. 2287–2296.
- [20] ZHANG, W. X.—YANG, H. Z.—MA, J. M.—QIU, G. F.: Concept Granular Computing Based on Lattice Theoretic Setting. In: Bargiela, A., Pedrycz, W. (Eds.): *Human-Centric Information Processing Through Granular Modelling*. Springer, Berlin, Heidelberg, *Studies in Computational Intelligence*, Vol. 182, 2009, pp. 67–94.
- [21] WANG, H.—ZHANG, W. X.: Relationships Between Concept Lattice and Rough Set. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L. A., Żurada, J. M. (Eds.): *Artificial Intelligence and Soft Computing – ICAISC 2006*. Springer, Berlin, Heidelberg, *Lecture Notes in Computer Science*, Vol. 4029, 2006, pp. 538–547.

- [22] ZHANG, Q. H.—WANG, G. Y.—LIU, X. Q.: Rule Acquisition Algorithm Based on Maximal Granule. *Pattern Recognition and Artificial Intelligence*, Vol. 25, 2012, No. 3, pp. 388–396 (in Chinese).
- [23] LIN, Y.—ZHANG, C. P.: Semantic Analyses of Rough Truth for Axioms in Modal Logic. *Journal of Computer Research and Development*, Vol. 43, 2006, No. 11, pp. 1999–2004.
- [24] GODIN, R.—MISSAOUI, R.—ALAOUI, H.: Incremental Concept Formation Algorithms Based on Galois (Concept) Lattices. *Computational Intelligence*, Vol. 11, 1995, No. 2, pp. 246–267.
- [25] SCHULTZ, B. R.: *Lecture Notes on Point-Set Topology*. Unpublished, 2012.
- [26] PAWLAK, Z.: *Rough Sets: Theoretical Aspects of Reasoning about Data*. Kluwer Academic Publishers Press, 1992.
- [27] PAWLAK, Z.: *Information Systems Theoretical Foundations*. *Information Systems*, Vol. 6, 1981, No. 3, pp. 205–218.
- [28] PAWLAK, Z.—SKOWRON, A.: Rudiments of Rough Sets. *Information Sciences*, Vol. 177, 2007, No. 1, pp. 3–27.
- [29] SHANNON, C. E.—WEAVER, W.—WIENER, N.: *The Mathematical Theory of Communication*. University of Illinois Press, 1998.
- [30] ZHANG, L.—ZHANG, B.: *Problem Solving Theory and Application: Granularity Computing of Quotient Space Theory and Its Application*. Tsinghua University Press, 2007.
- [31] XIONG, J. C.: *Point Set Topology Notes*. Higher Education Press, 2011.
- [32] LIANG, J. Y.—QIAN, Y. H.: Information Granules and Entropy Theory in Information Systems. *Science China Information Sciences*, Vol. 51, 2008, No. 10, pp. 1427–1444.



Qiang Wu is presently working as Assistant Professor in the Department of Computer Science and Engineering at Shaoxing University. He has successfully graduated with M.Sc. and Ph.D. degrees from Dalian University of Technology and Shanghai University, respectively. His research interests include intelligent data processing, knowledge representation and discovery. He has more than 20 research publications in reputed international journals and conferences.



Haiyan Shi received her Ph.D. degree in control theory and control engineering from Zhejiang University of Technology, Hangzhou, China, in 2013. Her research interests include wireless sensor network, image processing and artificial intelligence.



Liping Xie received her B.Sc. degree from Northeast Normal University in 2008. Her research area is data mining and mechanical design. She has published several research publications in national and international conferences and journals.

SEMPCA-SUMMARIZER: EXPLOITING SEMANTIC PRINCIPAL COMPONENT ANALYSIS FOR AUTOMATIC SUMMARY GENERATION

Óscar ALCÓN, Elena LLORET

Department of Software and Computing Systems

University of Alicante

Apdo. de correos 99, E-03080, Alicante, Spain

e-mail: {oalcon, elloret}@dlsi.ua.es

Abstract. Text summarization is the task of condensing a document keeping the relevant information. This task integrated in wider information systems can help users to access key information without having to read everything, allowing for a higher efficiency. In this research work, we have developed and evaluated a single-document extractive summarization approach, named SemPCA-Summarizer, which reduces the dimension of a document using Principal Component Analysis technique enriched with semantic information. A concept-sentence matrix is built from the textual input document, and then, PCA is used to identify and rank the relevant concepts, which are used for selecting the most important sentences through different heuristics, thus leading to various types of summaries. The results obtained show that the generated summaries are very competitive, both from a quantitative and a qualitative viewpoint, thus indicating that our proposed approach is appropriate for briefly providing key information, and thus helping to cope with a huge amount of information available in a quicker and efficient manner.

Keywords: Natural language processing, human language technologies, intelligent information processing, automatic text summarization, principal component analysis

Mathematics Subject Classification 2010: 68-T50

1 INTRODUCTION

In the current digital society, the increase of available data and the impossibility to cope with it in an efficient manner requires more and more the development of techniques capable of reducing and analyzing such data without losing the key and relevant ideas contained. One way of reducing the dimension of a document keeping, and at the same time having a relevant information is thinkable through automatic text summarization [1], which can be applied to a wide range of domains to help digest and manage information in an easier and less-time consuming manner. In this research area, summaries are created with the purpose of identifying the gist of the original text, while discarding irrelevant information, so advanced techniques to perform data analysis and knowledge interpretation are required. As a result, only the key information is selected and extracted. The type of summarization generated under these premises is extractive, in the sense that a summary would be produced just by copying the relevant sentences from the original document, without rewriting them or making any change to the produced text. Although many different approaches have been developed since this research area emerged in the late fifties [2], it is still a field under research, where issues that may seem successfully addressed to some extent, such as the correct identification of key information for building extractive text summarization approaches, need to be further studied and improved [3]. A correct identification of the key information would greatly benefit more advanced summarization processes, where the information is then compressed, fused or re-generated, as in the case of abstractive summarization. Furthermore, companies could take advantage of the integration of text summarization systems as management tools in their daily processes. This integration would allow to optimize information harvesting and digestion. For instance, an automatic process could decide whether to distribute or not some information that may be of interest, thus avoiding employees and CEOs having to read and manually filter tons of information.

Therefore, the aim of this research work is to describe and evaluate a novel approach for single-document extractive summarization using Principal Component Analysis (PCA) technique enhanced with semantic analysis (SemPCA-Summarizer). PCA is a statistical technique to compress data by reducing the number of dimensions, without much loss of information [4]. In the proposed approach, a concept-sentence matrix is built from the input document, and then, PCA is used to identify and rank the concepts, under the hypothesis that semantic analysis may have a positive impact on the summarization process, and thus, in the generated summaries, in contrast to the traditional process of taking only into consideration the individual words contained in the text for computing frequencies. This is where the semantic analysis plays a key role within our text summarization approach. Finally, to create the summary, we analyze four different heuristics to order the sentences based on the presence and importance of each concept, so that the ones leading to the best summaries can be determined and chosen. The results obtained show that our method produces good summaries, both from a quantitative

and qualitative point of view, as well as it overperforms the results of other existing summarization approaches. Our main contributions to the state of the art are:

- Semantic analysis is performed to detect synonymous words within the text, that constitute our working elements, i.e., the concepts to be analyzed.
- A novel approach for computing and determining relevant concepts is proposed.
- Four different heuristics are analyzed to select and order the most relevant sentences, leading to the creation of different types of summaries within the same method, as well as determining the approach that generates the best summaries.
- The performance of the proposed approach against a set of state-of-the-art summarization systems is compared, obtaining very good results for SemPCA-summarizer.

The remainder of the paper is structured as follows. In Section 2, the information about previous research related to extractive text summarization with data mining techniques is described. In Section 3, we explain PCA technique and how we have used it for text summarization, i.e., our Semantic PCA-based extractive summarization approach, SemPCA-Summarizer. Section 4 explains the automatic and manual evaluation methodology carried out to test our suggested approach. The results obtained together with their discussion are provided in Section 5 and finally, the conclusions as well as several future research works are outlined in Section 6.

2 RELATED WORK

A wide range of techniques, including statistical [5], graph-based [6, 34], machine learning [7], or linguistically motivated theories [8, 33] have been used by the research community to address extractive text summarization [3]. For this type of summaries, such techniques are mainly used for determining the key information in documents, and scoring sentences based on their relevance using different techniques and algorithms, such as spreading activation [9], genetic algorithms [35], word and phrase embeddings [10], integer linear programming [11], or even neural networks, which have gained a lot of interest in the recent years [12, 13]. Other summarization approaches integrate several techniques (statistical, linguistic, etc.) in order to analyze the benefits of combining them. This is the case of COMPENDIUM summarizer [36], which employs textual entailment together with statistical and linguistic-based features for scoring sentences, and determines which ones are more relevant to take part in the summary. Also, the approach proposed in [37] analyzed different combinations of statistical and linguistic settings, such as anaphora resolution together with Word Sense Disambiguation (WSD) methods for extracting the most important sentences based on the number of concepts they contained, instead of terms.

When focusing on similar or related statistical techniques as the ones used in the current research work (i.e., the use of PCA for summarization), the first ap-

proach using PCA for text summarization was presented in [14]. In this approach, PCA was employed for quantifying both word frequency and co-occurrence in the document to extract thematic words. The significant sentences of the text were selected using those thematic words under the hypothesis that those sentences would represent the main topic of a document. An improved approach derived from this research work was later developed in [15], in which PCA was used to identify relevant terms and then, Singular Value Decomposition (SVD) was employed for extracting the significant sentences associated to such terms (the higher number of terms a sentence had, the more important it was considered) and build the final summary. The experiments performed over a set of Korean newspaper articles showed that the method that used PCA and SVD jointly achieved the best performance for F-measure (0.436), compared to the method in which only PCA was employed (0.416).

A different approach was developed in [16], where single- and multi-document summaries were generated using a modified version of the vector space model, called Semantic Vector Space Model (SVSM) to model the set of documents including what the authors defined as action words. According to them, action words are verbs that are used to strengthen the way experiences are presented regardless expressing positive or negative experiences (e.g. break, destroy, arrive, etc.). In this approach, PCA was used to extract topic features. In the context of single-document summarization, it was evaluated in a controlled environment, using three individual articles belonging to several topics. It obtained an average accuracy of 61.85% when the extracted sentences of the proposed summarization method were compared to the sentences that a human would have extracted. Despite the high accuracy obtained, the authors did not test their approach with a bigger corpus, so this approach could not be validated with a wider set of documents.

Moreover, at some period of time, there seemed to be an increasing trend towards the use of data mining techniques for text summarization, including SVD, Latent Semantic Analysis (LSA), or PCA. Indeed, all these techniques (SVD, LSA, PCA) are tightly related, differing in the way they process and decompose the term-document matrix. In [17], a comparison between different approaches using SVD for generating summaries was carried out. From this comparison, [18] got a F-measure of 0.60 with their feature-based approach created by single LSA-based sentence scores. Their system included a set of textual features belonging to prosodic and lexical features. LSA was also used for computing the score of the sentences. Their experiments were carried out using both human transcriptions and the output of an automatic speech recognizer.

Although PCA has been already employed for text summarization [14], to the best of our knowledge, the influence of semantic analysis for computing relevant concepts with PCA has not been exploited so far. Traditionally, the PCA matrix is built from word co-occurrence, and consequently, the potentials of the knowledge obtained through a semantic analysis process are not integrated.

In the following section, our PCA-based approach for single-document extractive text summarization is described in detail.

3 SEMPCA-SUMMARIZER: A SEMANTIC PCA-BASED EXTRACTIVE SUMMARIZATION SYSTEM

Text summarization approaches follow a generic flow distinguishing three phases [19]:

1. *interpretation*;
2. *transformation*; and
3. *summary generation*.

In the interpretation stage, the text has to be understood. From a computational perspective, this is usually done using language processing tools, capable of analyzing the text at different language-levels (e.g., lexical, syntactic, or semantic). Next, in the transformation stage, the text is represented using the information obtained from the previously mentioned tools. The text could be represented in different ways using, for instance, a vector-space model, graphs, etc. Finally, once this representation is obtained, the last stage is devoted to apply some algorithm to determine the important information and generate the final summary based on it. We followed the same convention for defining the stages of our text summarization approach. Figure 1 shows an overview of the process for the proposed *SemPCA-Summarizer* approach.

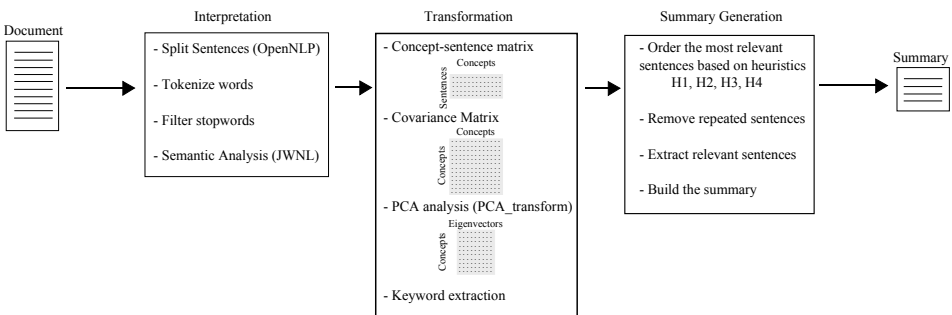


Figure 1. SemPCA-Summarizer approach for extractive single-document summarization

Firstly, the document needs to be interpreted to create the concept-sentence matrix, which will be analyzed using PCA. This is carried out by performing a linguistic analysis of the text, explained in Section 3.1. Next, the concept-sentence matrix reflects the co-occurrence of the concepts (instead of words) in each sentence. That is the way how we obtain a higher level of abstraction by considering concepts instead of just words, since a concept groups a set of synonymous words (for instance, *hurricane*, *tornado*, and *cyclone*). The process of creating such a matrix, applying PCA, and subsequently, extracting the concepts with the highest weights, determined by PCA, corresponds to the transformation stage, explained in Section 3.2. Lastly, in the summary generation phase, using these con-

cepts as a sign of relevance, the most important sentences will be selected and ordered using different heuristics, so the final summary can be generated (Section 3.3).

3.1 Interpretation: Linguistic Analysis and Concept Identification

A basic linguistic text preprocessing is necessary to proceed with the creation of the concept-sentence matrix. Once the input text is split in sentences, using the OpenNLP¹ library for Java, each of them is tokenized to subsequently filter stopwords (i.e., those words lacking semantic meaning that are not useful for calculating the frequency of occurrence, such as “the”). Afterwards, semantic analysis is applied to each word in order to identify concepts. For this, the knowledge base WordNet [20] was chosen to perform the semantic analysis using its Java library JWNL². WordNet is a lexico-semantic English resource that groups words into sets of synonyms called synsets, providing short and general definitions. It also provides information about the semantic relationships between the synsets. In our proposed approach, it is used to infer existing sets of synonyms in the documents, thus working with concepts instead of terms.

For identifying concepts, the process searches for the first synset of each word in the document. The first synset WordNet returns corresponds to the most frequent sense of that word, and therefore its most probable meaning. This approach has been proven to obtain very competitive results compared to more sophisticated word sense disambiguation techniques [21, 22, 23, 24]. If two words have the same first synset, we will consider them as synonyms. Therefore, they will be grouped under the same concept, and their occurrences will be added together. For example, *detonation* and *explosion* are different words but their WordNet’s first synsets are the same (07323181), so we keep them as a single concept in the concept-sentence matrix.

At the end of this stage, the text is prepared to compute the concept-sentence matrix and apply PCA technique, as it is next explained.

3.2 Transformation: PCA for Key Information Detection

In this stage, the concept-sentence matrix is created, from which the PCA technique will be applied.

PCA is a statistical technique focused on the extraction of information to compress and interpret the data [4]. For large volumes of data, the aim of this algorithm is to find a set of patterns or trends to reduce the dimensionality in the input data set.

PCA creates projections of the input samples in a subspace of a smaller dimension by finding a linear combination of the original data. The linear combina-

¹ <https://opennlp.apache.org/>

² <http://sourceforge.net/projects/jwordnet>

tion is constructed with respect to the importance in terms (in decreasing order) of the total variability of the sample population. The covariance matrix is computed to obtain the principal components (eigenvectors, e) and their corresponding weights (eigenvalues, λ). In this respect, the covariance matrix can be decomposed into a set of eigenvalue-eigenvector pairs $(\lambda_1, e_1), (\lambda_2, e_2), \dots, (\lambda_p, e_p)$, where $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p$.

PCA returns a matrix in which the eigenvectors are the columns and the rows are the variables of the covariance matrix. The eigenvectors are composed by the contribution of each variable, which determines the importance of the variable in the eigenvector. Moreover, the eigenvectors are derived in decreasing order of importance determined by the eigenvalues.

In this manner, an eigenvector with high eigenvalue carries a great amount of information. Therefore, the first eigenvectors collect the major part of the information extracted from the covariance matrix.

In our approach, PCA is applied using the `PCA_transform`³ Java library to process the covariance matrix from the concept-sentence matrix. In our concept-sentence matrix, the concepts (nouns, verbs, adverbs, and adjectives) are considered as variables (columns), whereas the sentences are the observations of the matrix (rows). This is also a novelty with respect to [14], where only nouns were used as variables.

Once PCA is applied to the covariance matrix, for each eigenvector, ordered by importance, the concept(s) with the highest value is/are extracted, as they are considered being more relevant than others. These concepts will be used for selecting the most important sentences, as shown next.

3.3 Summary Generation

From the previous stages, the matrix with the eigenvectors from PCA is obtained; however, an important stage in any extractive summarization process is to finally determine and select the specific sentences that will constitute the summary to be used by users or other Natural Language Processing (NLP) processes. Therefore, in this stage, the strategy for choosing the key sentences to form the final summary, based on the values obtained from PCA is defined.

Once we know the relevant keyword, it is frequent that the same keyword appears in more than one sentence of the documents, so we need to come up with a method to determine which sentences should be part of the final summary that contain such keyword. Therefore, four heuristics are defined for selecting and ordering the most relevant sentences from the document. The heuristics were chosen following the definition and purposes of different types of summaries (e.g., generic, informative, topic-focused, etc.) with the aim to allow the approach to be more flexible concerning the generation of various kind of summaries that could be appropriate depending on the users' information needs. Taking into account the concept

³ https://github.com/mkobos/pca_transform

with the highest value for each eigenvector from the PCA matrix, the approach selects and extracts:

H1: one sentence (searched in order of appearance in the original text) in which such concept appears. During this process, if a sentence had been already selected by a previous concept to take part in the summary, we would select and extract the next sentence in which the concept appears to avoid including redundant information.

H2: only the first sentence in which such concept appears. If a sentence had been already selected by a previous concept to take part in the summary, we would skip that sentence, and go ahead to the next concept. This heuristic is similar to H1, but since H2 focuses only on the first sentence, it does not allow to select and extract other sentences associated to each specific concept.

H3: all the sentences in which such concept appears.

H4: all the sentences in which at least two relevant concepts appear, giving priority to the concept with higher weight according to the PCA matrix. This heuristic is similar to the one proposed in [14] which selects the sentences based on the number of thematic words determined by PCA the sentence contain, trying to maximize the number of thematic words a sentence has. However, H4 differs from it in the fact that we prioritize the importance of the concepts in the sentence, rather than the number of concepts that a sentence may contain.

If we found different concepts with the same highest value for the same eigenvector, we would extract the corresponding sentences for all these concepts. In the same manner, if a concept was represented by several terms that are synonyms, we would extract the corresponding sentences for each of these synonyms. For instance, let us suppose that the concept id *07323181* was detected as an important concept by the approach. This concept corresponds to the terms *detonation* and *explosion*. In case both terms appear in the original document, the sentences in which those terms appear would be potentially selected to be part of the resulting summary.

Moreover, these strategies provide us with the relevance of the sentences in decreasing order, that are chosen for building the summary until the desired length is reached. In all cases, redundancy in the final summary is avoided by not allowing the inclusion of repeated sentences, if these have already been selected.

4 EVALUATION ENVIRONMENT

The conducted evaluation verifies the performance of the proposed automatic text summarization approach, i.e., determines the accuracy of the generated summaries and assesses their quality with respect to the information contained, and their readability.

In this section, the corpus and the evaluation methods designed to test our summaries are described. Both the quantitative and qualitative evaluation was performed. Quantitative evaluation conducts an automatic evaluation comparing the generated automatic summaries with respect to manual summaries, which are taken into account as model summaries. Qualitative evaluation completes the automatic evaluation, by means of a manual evaluation, which gathers and analyzes the users' opinion about the different generated summaries.

4.1 Corpus

Our approach was evaluated with news documents using the DUC 2002 corpus⁴. In particular, this corpus consists of 533 different English news articles, and it contains up to three human summaries for each document. These human summaries, that have a length of approximately 100 words, will be considered as model summaries for our quantitative evaluation.

Despite the fact that more recent datasets are available for testing novel summarization approaches, the reason for selecting this dataset is twofold. Firstly, it is a corpus suitable for developing and testing generic single-document summarization systems, since further DUC editions proposed specific summarization tasks (e.g. headline generation, update summarization), and in most of the cases the provided corpora focused on multi-document summarization. Secondly, this dataset is one of the most widespread corpus for single-document summarization, thus allowing to compare our proposed method with state-of-the-art approaches under the same conditions.

We therefore used the DUC 2002 news documents as input for SemPCA-Summarizer (one document at a time) and generated the corresponding summaries with each of the heuristics previously described. To provide a fair comparison with respect to the model summaries available, automatic summaries were produced taking into account a 100 word length limit.

4.2 Quantitative Evaluation

The quantitative evaluation was performed using ROUGE [25]. This tool allows the automatic evaluation of text summaries by comparing their content to a model one, created by a human. The idea under ROUGE is to measure the number of overlapping n-gram units between both types of summaries (e.g., unigrams, bigrams, or word sequences). Based on these, ROUGE implements different metrics, such as ROUGE-1, ROUGE-2, ROUGE-L, or ROUGE-SU4. Following [26], we will take into account: unigram similarity (ROUGE-1); bigram similarity (ROUGE-2); longest common subsequence (ROUGE-L) and bigram similarity skipping unigrams (ROUGE-SU4).

⁴ http://www-nlpir.nist.gov/projects/duc/data/2002_data.html

In addition, for each of the aforementioned metrics, ROUGE returns the commonly used NLP measures: Precision, Recall and F-measure.

$$\textit{Precision} = \frac{\# \textit{CorrectPhrasesExtracted}}{\# \textit{TotalPhrasesExtracted}}, \quad (1)$$

$$\textit{Recall} = \frac{\# \textit{CorrectPhrasesExtracted}}{\# \textit{CorrectPhrasesTest}}, \quad (2)$$

$$\textit{F-measure} = \frac{2 * \textit{Precision} * \textit{Recall}}{\textit{Precision} + \textit{Recall}} \quad (3)$$

where $\# \textit{CorrectPhrasesExtracted}$ is the number of correct sentences that the evaluated approach extracts, $\# \textit{TotalPhrasesExtracted}$ the total number of sentences that the evaluated approach extracts and $\# \textit{CorrectPhrasesTest}$ the total number of sentences included in the human summaries. Human summaries were manually generated by experts, so they are considered as gold-standard summaries.

4.3 Qualitative Evaluation

To verify if the generated summaries show an acceptable degree of coherence, they will be qualitatively evaluated through a manual evaluation. With this type of evaluation, our aim is to know the users' opinion about our summaries regarding different quality criteria and their readability.

In order to define an appropriate manual evaluation environment, a revision of previous works also conducting this type of evaluation was carried out first. In [27], seven questions were formulated regarding the following aspects: coherence, non-redundancy (2 questions), clarity, cohesion, grammaticality and readability. Each of them was evaluated giving a scoring from 1 to 5 (1 = very poor, and 5 = very good). This way of proceeding is known as the Likert rating scale [28].

Similar to the previous one, other examples can be found ([27, 29]) which employ a Likert scale with seven questions to evaluate the summaries too, but slightly varying the aspects that are evaluated in a summary: five questions about linguistic features (*grammaticality*, *non-redundancy*, *referential clarity*, *focus*, *structure and coherence*), one question based on amount of information in the summary that contributes to meeting the information need expressed in the topic (*content responsiveness*), and another one about the overall response of the summary (i.e., *overall responsiveness*).

More recently, the criteria to qualitatively evaluate summaries by the National Institute of Standards and Technology [30] focused on *content*, *readability/fluency* and *general responsiveness*. In this case, summaries were truncated to 100 words, and each summary was assessed by four people. Again, each of these criteria was evaluated according the Likert scale. In [31], the Likert-scale assessment was proposed using the following five criteria of quality: *grammaticality*, *non-redundancy*, *clarity*, *accuracy (focus)*, and *coherence*.

For qualitative evaluation, the Likert-scale evaluation seems to be one of the most accepted methodologies, so we also opted for this type of scale for our evaluation. Regarding the types of questions for assessing the summaries, seven questions were defined, some of which were adapted from already existing ones [32]:

1. Reading only the first sentence of the summary, does it allow to have a clear idea of what the original document is about?
2. Do you think the summary reflects the relevant information from the original document?
3. Does the summary contains the information in a non-redundant way?
4. Do you think that the sentences in the summary are correctly ordered?
5. Do you think that the content of the summary can be well understood?
6. Would you say that the summary is easy to read?
7. In general, do you consider it a good summary?

These questions were grouped with respect to:

- Accuracy of the information (questions 1 and 2)
- Redundancy (question 3)
- Readability/understanding (questions 4, 5, and 6)
- Overall assessment (question 7)

The responses associated to each question were measured in the range of 1 to 5, under the following scale:

1. Strongly disagree
2. Disagree
3. Neither agree nor disagree
4. Agree
5. Strongly agree

5 RESULTS AND DISCUSSION

In this section, the performed experiments and the results are shown and discussed.

5.1 Quantitative Results

As mentioned in Section 4.2, ROUGE was used for the quantitative evaluation. ROUGE was computed for the summaries obtained from the heuristics proposed in Section 3.3. Since we used 533 input texts from DUC 2002 Corpus, we obtained 533 summaries for each heuristic (we have four heuristics), thus having a total of

2 132 summaries for evaluation, and 1 085 DUC 2002 model summaries⁵ as reference. Figure 2 provides example summaries generated from document id *WSJ880912-0064* of the DUC 2002 corpus, described in Section 4.1.

To check the appropriateness of performing semantic analysis, our approach was run using semantic analysis (i.e., concept-sentence matrix) and without it (i.e. using word-sentence matrix). The results can be seen in Table 1 and Table 2.

	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
H1	0.46454	0.46688	0.46508	0.21011	0.21127	0.21041	0.42233	0.42459	0.42290	0.22804	0.22928	0.22834
H2	0.46540	0.46048	0.46148	0.20814	0.20491	0.20571	0.42183	0.41742	0.41831	0.22699	0.22379	0.22453
H3	0.43619	0.44018	0.43799	0.18561	0.18732	0.18638	0.39602	0.39966	0.39767	0.20724	0.20923	0.20814
H4	0.44037	0.43446	0.43535	0.18605	0.18296	0.18359	0.39955	0.39441	0.39521	0.20886	0.20555	0.20614

Table 1. Results for the SemPCA-Summarizer summarization system

	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
H1	0.46265	0.46691	0.46452	0.20885	0.21052	0.20952	0.41917	0.42299	0.42083	0.22661	0.22850	0.22738
H2	0.46479	0.46594	0.46490	0.20915	0.20904	0.20881	0.42121	0.42217	0.42125	0.22711	0.22713	0.22682
H3	0.43310	0.43840	0.43563	0.18267	0.18503	0.18380	0.39267	0.39750	0.39498	0.20515	0.20781	0.20642
H4	0.43898	0.43882	0.43758	0.18630	0.18541	0.18506	0.39706	0.39711	0.39599	0.20840	0.20889	0.20787

Table 2. Basic PCA-based extractive summarization approach without semantic analysis

The best overall F-measure values are emphasized in boldface. These correspond to H1 with semantic analysis (Table 1). A t-test was conducted to account for statistical significance. We compared the use of semantic knowledge with respect to the basic method without using this type of knowledge for each F-measure result of H1,H2, H3, and H4, but no statistical significance was found. Nevertheless, analyzing and comparing the same heuristic with and without semantic analysis, we find that in two of them (H1 and H3), the semantic analysis contributes to slight improvement of the results, obtaining an average ROUGE improvement of 0.36 % and 0.86 %, respectively. The reason of this small improvement could be probably caused by either the absence of synonyms in texts due to the finer granularity of the resource used (i.e. WordNet). However, for H2 and H4, there was not any improvement at all when employing semantic knowledge, and what is more, the results with semantic analysis even suffer a dropped of 1 % and 0.57 %, respectively. This may be due to the fact that these two heuristics prioritize the amount of concepts in contrast to H1 and H3, that gave priority to the importance of the concepts in sentences. Therefore, this issue could negatively affect the results, since it would be difficult to find synonyms in the same sentences, and as a result, the sentences extracted would be less relevant.

Considering the nature of the four suggested heuristics, and when having a look at the summaries generated, we believe that each of the heuristic could be particularly suitable for generating a specific type of summary. In this manner, H1

⁵ The number of model summaries is lower because the range of human summaries existing for each document varies from 1 to 3.

Human summary: Hurricane Gilbert swept toward Jamaica yesterday with 100 mph winds, and officials issued warnings to residents on the southern coasts of the Dominican Republic, Haiti and Cuba. Hal Gerrish, with the National Hurricane Center in Coral Gables, Fla., said, "It's still gaining strength. It's certainly one of the larger systems we've seen in the Caribbean for a long time." The governments of Haiti, Cuba and Jamaica issued warnings. In Puerto Rico, the storm injured three people, tore off several roofs, caused coastal flooding and brought down trees and power lines in the west and southwest regions.

Best DUC 2002: Hurricane Gilbert swept toward Jamaica yesterday with 100-mile-an-hour winds, and officials issued warnings to residents on the southern coasts of the Dominican Republic, Haiti and Cuba. The storm ripped the roofs off houses and caused coastal flooding in Puerto Rico. Forecasters said the hurricane was gaining strength as it passed over the ocean and would dump heavy rain on the Dominican Republic and Haiti as it moved south of Hispaniola, the Caribbean island they share, and headed west. At 3 p.m. EDT, the center of the hurricane was about 100 miles south of the Dominican Republic and 425 miles east of Kingston, Jamaica.

SemPCA- H1: The storm ripped the roofs off houses and caused coastal flooding in Puerto Rico. Forecasters said the hurricane was gaining strength as it passed over the ocean and would dump heavy rain on the Dominican Republic and Haiti as it moved south of Hispaniola, the Caribbean island they share, and headed west. Hurricane Gilbert swept toward Jamaica yesterday with 100-mile-an-hour winds, and officials issued warnings to residents on the southern coasts of the Dominican Republic, Haiti and Cuba. The hurricane was moving west at about 15 mph and was expected to continue this motion for the next 24 hours. In the Dominican Republic, all domestic flights and flights to and from Puerto Rico and Miami were canceled.

SemPCA- H2: The storm ripped the roofs off houses and caused coastal flooding in Puerto Rico. Forecasters said the hurricane was gaining strength as it passed over the ocean and would dump heavy rain on the Dominican Republic and Haiti as it moved south of Hispaniola, the Caribbean island they share, and headed west. Hurricane Gilbert swept toward Jamaica yesterday with 100-mile-an-hour winds, and officials issued warnings to residents on the southern coasts of the Dominican Republic, Haiti and Cuba. In the Dominican Republic, all domestic flights and flights to and from Puerto Rico and Miami were canceled. Forecasters said the hurricane's track would take it about 50 miles south of southwestern Haiti.

SemPCA- H3: The storm ripped the roofs off houses and caused coastal flooding in Puerto Rico. In the Dominican Republic, all domestic flights and flights to and from Puerto Rico and Miami were canceled. The hurricane center said small craft in the Virgin Islands and Puerto Rico should remain in port until conditions improve. In Puerto Rico, besides tearing off several roofs, the storm caused coastal flooding and brought down power lines and trees along roads and highways in the west and southwestern regions. Three people were injured in Guayanilla, Puerto Rico, when a tree fell on their vehicle as they traveled along Route 97, police reported.

SemPCA- H4: The storm ripped the roofs off houses and caused coastal flooding in Puerto Rico. In the Dominican Republic, all domestic flights and flights to and from Puerto Rico and Miami were canceled. The hurricane center said small craft in the Virgin Islands and Puerto Rico should remain in port until conditions improve. In Puerto Rico, besides tearing off several roofs, the storm caused coastal flooding and brought down power lines and trees along roads and highways in the west and southwestern regions. Four policemen stationed on Mona Island, between Puerto Rico and the Dominican Republic, were stranded as a result of the weather.

Figure 2. Example of automatic and human summaries generated from document *WSJ880912-0064* of the DUC 2002 corpus

would be appropriate for generic informative summaries, H2 for generic indicative summaries, H3 for topic-oriented informative summaries, and H4 for topic-oriented indicative summaries. In this case, we performed a one-way ANOVA statistical test with a Tukey HSD Post Hoc analysis for each of the heuristics in order to account for statistical differences between them in each of the methods analyzed (SemPCA-summarizer and Basic PCA-based summarizer). We found that, in both approaches, the results were statistically significant (at least with $(p\text{-value} < 0.05)$ in all the pairwise comparison, except for H1–H2 and H3–H4. This is expected since H1–H2 focus on the production of generic summaries, while H3–H4 are aimed to produce topic-oriented summaries, and the differences between them is the amount of detail about a concept that is provided in the summary, being both types appropriate. Moreover, since the human summaries we used for comparing our automatic summaries were built from a generic and informative point of view, this could explain why for this corpus, H1 leads to the best ROUGE results for all the metrics studied.

Furthermore, our best approach (PCA enhanced with semantic analysis and using H1 heuristic) was compared to other state-of-the-art summarization approaches under the same conditions (same corpus and same evaluation metrics). The comparison can be seen in Table 3, in which ROUGE-1 recall value of our best approach (H1) is compared with:

- the lead baseline of DUC 2002, which takes approximately the first 100 words in the documents;
- the best DUC 2002 approach [33], which was based on a supervised sentence extraction using a Hidden Markov Model and a Logistic Regression Model;
- wMVC summarizer [34]. This approach, called the Weighted Minimum Vertex Cover summarization approach, considered the text summarization task as an optimization problem. A graph-based algorithm was used, where vertices of the graph represented the sentences and graph edges represented the connections between sentences;
- MUSE [35], an approach based on the linear optimization of several sentence ranking measures using a genetic algorithm;
- COMPENDIUM [36], which employed textual entailment, statistical and linguistic-based features for scoring sentences, and determines which ones had to take part in the summary;
- TS + MFS-approach [37];
- TS + UKB-approach [37];

The last two approaches, [37], used anaphora resolution and a Word Sense Disambiguation (WSD) method for enriching the document with semantic knowledge, and extracting the most important sentences based on the number of concepts they contained, instead of terms. The difference between them was the WSD method employed: MFS for the most frequent sense, and UKB for a PageRank-based WSD method [38].

The reason for using ROUGE recall metric for the comparison, instead of precision or F-measure is that we did not reimplement the summarization methods. Instead, we relied on published results for all the compared approaches, so the only common metric across all the approaches was the recall metric for ROUGE-1. From the comparison shown in Table 3, it can be seen that our PCA-based approach (H1) obtains the best results in terms of Rouge-1 recall metric over the DUC 2002 corpus.

	ROUGE -1 (Recall value)
SemPCA-Summarizer (H1)	0.46688
Best DUC 2002 approach	0.42776
Lead baseline DUC 2002	0.41132
wMVC summarizer [34]	0.38800
MUSE [35]	0.45490
COMPENDIUM [36]	0.46008
TS + MFS approach [37]	0.42339
TS + UKB approach [37]	0.42556

Table 3. Comparison with other approaches that used the DUC 2002 corpus

The quantitative results obtained show the appropriateness of the proposed approach for summarization, which is competitive with respect to the state of the art, having as the additional advantage that it is flexible enough to create different types of summaries (generic vs. topic-oriented; informative vs. indicative). This would definitely benefit the adaptation of the generated summary depending on the information needs and user profile. For instance, indicative summaries or headlines may be useful for CEOs to have a daily update of the most important facts without spending too much time reading more detailed information.

5.2 Qualitative Results

Given the importance of assessing the readability of a summary besides its content, a manual evaluation was conducted, as explained in Section 4.3. However, due to the difficulty and time-consuming task involving manual evaluation, we did not conduct this type of evaluation to all the generated summaries with our approach. This would have implied the manual evaluation of more than 2000 summaries, meaning a good amount of human resources and effort that was unfortunately not available. Therefore, we finally opted for selecting a representative sample of summaries of each type for the manual evaluation.

To find out the representative number of summaries, a statistical formula called *Representative sample (M)* [39] was employed:

$$M = \frac{N * K^2 * P * Q}{E^2 * (N - 1) + K^2 * P * Q} \quad (4)$$

where N is the current population, K is confidence level, P is the assumed probability of success, Q is the probability of failure, i.e., $1-P$, and E is the error rate. The values for each parameter were set according to the suggestions in [40]:

$$K = 0.95; \quad E = 0.05; \quad P = 0.5; \quad Q = 0.5. \quad (5)$$

Since we also wanted to compare our approach with the best DUC 2002 approach, previously mentioned in Section 5.1, and the model summaries, the population was calculated taking into account 3 198 summaries (533 best DUC 2002 approach summaries, 533 human summaries and 2 132 summaries generated with our approach). The other approaches compared (e.g., MUSE, COMPENDIUM, etc.) were not manually evaluated since their generated summaries were not available.

Using Equation 4 we obtained $M = 87.79 \approx 88$ summaries. On account of this, it was decided to take a total sample of 90 abstracts to be manually evaluated. The total sample of 90 summaries was made up of 60 summaries of PCA text summarization types (15 summaries for each type), 15 summaries of the best DUC 2002 approach, and 15 model summaries created by humans in DUC 2002. All the summaries were truncated to approximately 100 words. The 90 abstracts were evaluated by 18 people (comprising PhD students and senior researchers with advanced knowledge in English), so each one had to evaluate 5 summaries. Summarization evaluation is a subjective task, so people evaluating a summary may have different opinions about its quality. In order to minimize the bias that may be obtained by allowing only one user to evaluate a summary, we enrich the qualitative evaluation, asking an additional external user to evaluate also the 90 summaries. In this manner, a summary would be evaluated by two people, and therefore the assessment would not only depend on one user. The final manual evaluation results were computed by averaging the scores assigned by the two assessors for each summary. In this manner, if the discrepancy between the two assessors was high, for instance, if one person assigned a good score, i.e. 5, and the other one a low score, i.e. 2, we would consider that the summary would not be either very bad or very good, and in this case the average would be a fair value for its final score.

In Table 4, the results of the manual evaluation are shown. As it can be seen, the results prove that, in a general way, the summaries are acceptable, obtaining values closer to the positive range of the Likert scale.

	Q1	Q2	Q3	Q4	Q5	Q6	Q7
DUC 2002 human	4.3	4.1	4.5	4.5	4.2	4.2	4.2
Best DUC 2002 approach	4.1	3.3	4.6	4.6	3.5	4.1	3.4
SemPCA-Summarizer with H1	3.7	3.6	4.4	3.6	3.6	4.0	3.6
SemPCA-Summarizer with H2	3.4	3.3	4.7	3.6	3.3	4.0	3.1
SemPCA-Summarizer with H3	3.1	3.3	3.9	3.8	3.6	3.5	3.2
SemPCA-Summarizer with H4	3.0	3.4	3.8	3.6	3.4	3.5	3.4

Table 4. Qualitative evaluation results obtained through a manual inspection of the generated summaries

As we expected, the human summaries exhibit the best results. This could be because abstractive summaries always seem to be better considered by humans than extractive ones. However, for several questions (e.g., Q3, Q6), the results do not greatly differ from the rest of the approaches (Best DUC 2002 approach, SemPCA-summarizer with H1, and SemPCA-summarizer with H2).

When focusing on the extractive summarization approaches, and firstly comparing only our heuristics among them, H1 is the one which obtains the best results, reaching the highest values in five out of the seven questions outlined. On the other hand, H3 and H4 show the lowest results, although the average obtained between all the questions is above a 3-rating score.

When inspecting the results for each type of question, the results of Q1 and Q2 show that the accuracy of the information in the summaries is coherent with the heuristics tested, being H1 the best heuristic. Here, it is important to mention that the first sentence of the Best DUC 2002 approach shows a clear idea of the information of the original text. This is logical, since this approach ensured that first sentence of the original document was always included in the automatic summary. However, when having a look at Q2, it seems that the rest of the summary contains less important information compared with the generated summaries using the other heuristics. Concerning the redundancy of the summaries (Q3), the results of all the evaluated approaches indicate that they are very good with respect to this issue.

Switching now to the readability/understanding questions, the results of Q4 reflect that the ordering of the sentences in the summary is better in the Best DUC 2002 approach than in any of our suggested heuristics. Again, this is expected, since the sentences in the Best DUC 2002 approach are shown in the same order as in the original document. In contrast, in the case of our heuristics, the sentences extracted are ordered with respect to the importance of the concepts and not necessarily by the order the sentences appear in the original texts. To improve this issue in our heuristics, we could have taken into account the position of the sentence in the text, and order the extracted sentences in the same order as in the original document. However, it seems that in the case of H1 this fact minimally affects the legibility of the summaries, since the results of Q5 and Q6 show good values. Compared to our heuristics, the summaries produced by the Best DUC 2002 approach does not exhibit a very high understanding score (Q5), which is lower than the score obtained for H1 and H3.

Finally, according to overall assessment results (Q7), some of our heuristics (H1 and H4) obtain equal or higher results than the Best DUC 2002 approach. This question (Q7) gives an idea about to what extent the generated summary would be appropriate. In this respect, we computed the number of summaries scored under each rating for this question for the two manual evaluations conducted. Figure 3 graphically depicts the results obtained comparing both manual evaluations.

As it can be seen, although the individual scoring may vary from different assessors, partly due to the subjectivity involved in the evaluation process and the fuzzy differences between the scores assigned (an assessor may have doubts between lower or upper boundaries, e.g., 1–2 or 4–5), we can see that better summaries are clearly

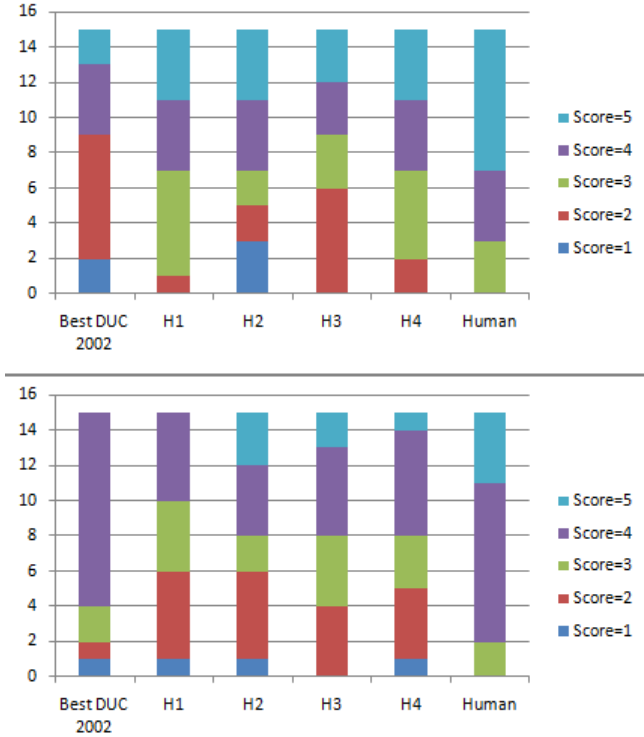


Figure 3. Results of manual evaluation for Q7. The first graphic corresponds to the original manual evaluation, and the second one to the enriched manual evaluation, where an additional external expert reevaluated all the summaries again.

distinguished from poor ones, as in the case of human summaries. In contrast, there are more discrepancies when it comes to automatic summaries, where there may be a greater variability in as far as users' opinions is concerned.

6 CONCLUSIONS AND FUTURE WORK

In this paper, a single-document extractive text summarization approach using Principal Component Analysis (PCA) technique enhanced with semantic information was proposed and analyzed. Our approach has been called SemPCA-Summarizer. Specifically, PCA was used as a detector and ranker of the relevant concepts within the text, to further extract the sentences associated to them according to different heuristics, so the final summary could be generated.

The results obtained have confirmed that SemPCA-Summarizer produces good summaries from the quantitative and qualitative point of view, and it is competitive with respect to the state of the art, so it can be very useful for detecting and

providing key information. Having analyzed different heuristics for selecting sentences based on the relevant concepts identified, the best performing heuristic is H1, which selects one sentence (searching in order of appearance in the original text) per relevant concept. In particular, it obtained the best results for all the ROUGE metrics tested, when semantic knowledge was included in the base system. Concerning the qualitative evaluation, H1 also achieved very good results compared with the remaining heuristics. It is also worth noting that the performed experiments and the obtained results allow us to gain some insights concerning the potential that our approach may have for generating other types of summaries, e.g., generic vs. topic-oriented, or indicative vs. informative summaries.

Although there is still some room for improvement, this finding could be very encouraging since it may constitute the starting point for further research.

In the short- and medium-term, we plan to analyze whether the approach could be improved if fragments of sentences instead of complete sentences were taken into account as the linguistic units to process. This could help to raise the accuracy of the relevant information. Moreover, it would be very interesting to test the approach for other types of documents, integrating more advanced word sense disambiguation methods that would take into account the context of the whole document in which a word appears. For this, word embeddings could be used. In the long-term, it would be very useful for the research community to test the appropriateness of our approach for multilingual text summarization, so that the approach could deal with texts of different languages.

Acknowledgements

This research work has been partially funded by the Generalitat Valenciana and the Spanish Government through the projects PROMETEOII/2014/001, TIN2015-65100-R, and TIN2015-65136-C2-2-R. Additionally, the authors would like to thank all the users who contributed to the summaries evaluation.

REFERENCES

- [1] GAMBHIR, M.—GUPTA, V.: Recent Automatic Text Summarization Techniques: A Survey. *Artificial Intelligence Review*, Vol. 47, 2017, No. 1, pp. 1–66, doi: 10.1007/s10462-016-9475-9.
- [2] LUHN, H. P.: The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development*, Vol. 2, 1958, No. 2, pp. 159–165.
- [3] NENKOVA, A.—K. MCKEOWN: Automatic Summarization. *Foundations and Trends in Information Retrieval*, Now Publishers Inc., Vol. 5, 2011, No. 2-3, pp. 103–233, doi: 10.1561/15000000015.
- [4] JOLLIFFE, I. T.: *Principal Component Analysis*. Springer Verlag, New York, 2002.

- [5] MCCARGAR, V.: Statistical Approaches to Automatic Text Summarization. *Bulletin of the American Society for Information Science and Technology*, Vol. 30, 2004, No. 4, pp. 21–25, doi: 10.1002/bult.319.
- [6] CANHASI, E.—KONONENKO, I.: Multi-Document Summarization via Archetypal Analysis of the Content-Graph Joint Model. *Knowledge and Information Systems*, Vol. 41, 2014, No. 3, pp. 821–842, doi: 10.1007/s10115-013-0689-8.
- [7] MEI, J.-P.—CHEN, L.: SumCR: A New Subtopic-Based Extractive Approach for Text Summarization. *Knowledge and Information Systems*, Vol. 31, 2012, No. 3, pp. 527–545, doi: 10.1007/s10115-011-0437-x.
- [8] UZÊDA, V. R.—PARDO, T. A. S.—NUNES, G. V.: A Comprehensive Comparative Evaluation of RST-Based Summarization Methods. *ACM Transactions on Speech and Language Processing*, Vol. 6, 2010, No. 4, Art. No. 4, doi: 10.1145/1767756.1767757.
- [9] NASTASE, V.—FILIPPOVA, K.—PONZETTO, S. P.: Generating Update Summaries with Spreading Activation. *Proceedings of Text Analysis Conferences (TAC)*, NIST, 2008.
- [10] KÅGEBÄCK, M.—MOGREN, O.—TAHMASEBI, N.—DUBHASHI, D.: Extractive Summarization Using Continuous Vector Space Models. *Proceedings of the 2nd Workshop on Continuous Vector Space Models and Their Compositionality (CVSC)*, Association for Computational Linguistics, Gothenburg, Sweden, 2014, pp. 31–39, doi: 10.3115/v1/W14-1504.
- [11] BOUDIN, F.—MOUGARD, H.—FAVRE, B.: Concept-Based Summarization Using Integer Linear Programming: From Concept Pruning to Multiple Optimal Solutions. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 1914–1918, doi: 10.18653/v1/D15-1220.
- [12] CAO, Z.—WEI, F.—DONG, L.—LI, S.—ZHOU, M.: Ranking with Recursive Neural Networks and its Application to Multi-Document Summarization. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, January 25–30, 2015, Austin, Texas, USA, 2015, pp. 2153–2159.
- [13] RUSH, A. M.—CHOPRA, S.—WESTON, J.: A Neural Attention Model for Sentence Summarization. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 379–389, doi: 10.18653/v1/D15-1044.
- [14] LEE, C. B.—KIM, M. S.—PARK, H. R.: Automatic Summarization Based on Principal Component Analysis. In: Pires, F. M., Abreu, S. (Eds.): *Progress in Artificial Intelligence (EPIA 2003)*. Springer, Berlin, Heidelberg, *Lecture Notes in Computer Science*, Vol. 2902, 2003, pp. 409–413.
- [15] LEE, C.—PARK, H.—OCK, C.: Significant Sentence Extraction by Euclidean Distance Based on Singular Value Decomposition. In: Dale, R., Wong, K. F., Su, J., Kwong, O. Y. (Eds.): *Natural Language Processing – IJCNLP 2005*. Springer, Berlin, Heidelberg, *Lecture Notes in Computer Science*, Vol. 3651, 2005, pp. 636–645.

- [16] VIKAS, O.—MESHRAM, A. K.—MEENA, G.—GUPTA, A.: Multiple Document Summarization Using Principal Component Analysis Incorporating Semantic Vector Space Model. *Computational Linguistics and Chinese Language Processing*, Vol. 13, 2008, No. 2, pp. 141–156.
- [17] BADRY, R. M.—ELDIN, A. S.—ELZANFALLY, D. S.: Text Summarization within the Latent Semantic Analysis Framework: Comparative Study. *International Journal of Computer Applications*, Vol. 81, 2013, No. 11, pp. 40–45.
- [18] MURRAY, G.—RENALS, S.—CARLETTA, J.: Extractive Summarization of Meeting Recordings. *Proceedings of the 9th European Conference on Speech Communication and Technology (Interspeech-2005 – Eurospeech)*, Lisbon, Portugal, September 4–8, Association for Computational Linguistics, 2005.
- [19] SPARCK JONES, K.: *Automatic Summarising: Factors and Directions*. *Advances in Automatic Text Summarization*, 1999, MIT Press.
- [20] MILLER, G. A.: WordNet: A Lexical Database for English. *Communications of the ACM*, Vol. 38, 1995, No. 11, pp. 39–41, doi: 10.1145/219717.219748.
- [21] PREISS, J.—DEHDARI, J.—KING, J.—MEHAY, D.: Refining the Most Frequent Sense Baseline. *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions (DEW'09)*, 2009, pp. 10–18, doi: 10.3115/1621969.1621973.
- [22] LLORET, E.—MOREDA, P.—MORENO, I.—CANALES, L.: *Meaning Disambiguator: v2.1*. Technical report, University of Alicante, 2014, <http://first-asd.eu/?q=D4.1>.
- [23] BHINGARDIVE, S.—SINGH, D.—RUDRA MURTHY, V.—REDKAR, H.—BHATTACHARYYA, P.: Unsupervised Most Frequent Sense Detection Using Word Embeddings. *Proceedings of the 2015 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 1238–1243, <http://www.aclweb.org/anthology/N15-1132>, doi: 10.3115/v1/N15-1132.
- [24] BHINGARDIVE, S.—SHUKLA, R.—SARASWATI, J.—KASHYAP, L.—SINGH, D.—BHATTACHARYYA, P.: Synset Ranking of Hindi WordNet. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Portorož, Slovenia, May 23–28, 2016.
- [25] LIN, C.-Y.: ROUGE: A Package for Automatic Evaluation of Summaries. *Proceedings of the ACL-04 Workshop on Text Summarization Branches Out*, Association for Computational Linguistics, 2004, pp. 74–81.
- [26] STEINBERGER, J.—JEZEK, K.: Sentence Compression for the LSA-Based Summarizer. *Proceedings of the 7th International Conference on Information Systems Implementation and Modelling*, 2006, pp. 141–148.
- [27] YEN, J.—OVER, P.: An Introduction to DUC-2004 Intrinsic Evaluation of Generic News Text. *Proceedings of the Document Understanding Conference (DUC 2004)*, 2004.
- [28] LIKERT, R.: A Method of Constructing an Attitude Scale. In: Maranell, G. M. (Ed.): *Scaling: A Sourcebook for Behavioral Scientists*. Chapter 19. Aldine Publishing Company, Chicago, 1974, pp. 233–243.

- [29] CONROY, J.—MCKEOWN, K.—SPARCK-JONES, K.—VANDERWENDE, L.: Overview of DUC 2006. Proceedings of the Document Understanding Conference (DUC 2006), 2006.
- [30] National Institute of Standards and Technology NIST: TAC 2011 Guided Summarization Task Guidelines. 2011.
- [31] LLORET, E.—PALOMAR, M.: Towards Automatic Tweet Generation: A Comparative Study from the Text Summarization Perspective in the Journalism Genre. *Expert Systems with Applications*, Vol. 40, 2013, No. 16, pp. 6624–6630, doi: 10.1016/j.eswa.2013.06.021.
- [32] CARLSSON, B.—JÖNSSON, A.: Using the Pyramid Method to Create Gold Standards for Evaluation of Extraction Based Text Summarization Techniques. Proceedings of the Swedish Language Technology Conference (SLTC 2010), 2010.
- [33] CONROY, J. M.—SCHLESINGER, J. D.—O’LEARY, D. P.—OKUROWSKI, M. E.: Using HMM and Logistic Regression to Generate Extract Summaries for DUC. Proceedings of the DUC 01 Conference, 2001, pp. 13–14.
- [34] GUPTA, A.—KAUR, M.—SINGH, A.—GOEL, A.—MIRKIN, S.: Text Summarization Through Entailment-Based Minimum Vertex Cover. Proceedings of the Third Joint Conference on Lexical and Computational Semantics (SEM 2014), 2014, pp. 75–80, doi: 10.3115/v1/S14-1010.
- [35] LITVAK, M.—LAST, M.—FRIEDMAN, M.: A New Approach to Improving Multilingual Summarization Using a Genetic Algorithm. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL ’10), Association for Computational Linguistics, 2010, pp. 927–936.
- [36] LLORET, E.—PALOMAR, M.: COMPENDIUM: A Text Summarisation Tool for Generating Summaries of Multiple Purposes, Domains, and Genres. *Natural Language Engineering*, Vol. 19, 2013, No. 2, pp. 147–186, doi: 10.1017/S1351324912000198.
- [37] VODOLAZOVA, T.—LLORET, E.—MUÑOZ, R.—PALOMAR, M.: The Role of Statistical and Semantic Features in Single-Document Extractive Summarization. *Artificial Intelligence Research*, Vol. 2, 2013, No. 3, pp. 35–44, doi: 10.5430/air.v2n3p35.
- [38] AGIRRE, E.—SOROA, A.: Personalizing PageRank for Word Sense Disambiguation. Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL ’09), Association for Computational Linguistics, 2009, pp. 33–41, doi: 10.3115/1609067.1609070.
- [39] PITA FERNÁNDEZ, S.: Sample Size Calculation (Determinación del Tamaño Muestral). *Cad Atención Primaria*, Vol. 3, 1996, pp. 114–138.
- [40] GUTIÉRREZ VÁZQUEZ, Y.—FERNÁNDEZ ORQUÍN, A.—MONTOYO GUIJARRO, A.—VÁZQUEZ PÉREZ, S.: Integration of Semantic Resources Based on WordNet. *Procesamiento del Lenguaje Natural*, Vol. 47, 2011, pp. 161–168.



Óscar ALCÓN currently works at CYPE Ingenieros S.A. as Product Development Responsible Manager. His tasks include the specification and supervision of software development for the design of common telecommunications infrastructures in buildings for Spain and Portugal. It includes activities such as market research, determination of deadlines, contacts with customers and collaborators, definition of lines of work, product presentations, commercial activities and relations with marketing and sales departments. In addition, he also conducts research into natural language processing and text summarization as a collaborator with the GPLSI research group at the University of Alicante, and more specifically for the project “DIIM2.0: Desarrollo de técnicas inteligente e interactivas de minería y generación de información sobre la web 2.0”.



Elena LLORET is a full-time Ph.D. Assistant Lecturer at the University of Alicante in Spain. There she obtained her Ph.D. focused on text summarisation in 2011. Her main interests are natural language processing and more specifically text summarisation, and natural language generation. She is the author of over 60 scientific publications in international peer-reviewed conferences and refereed journals. She has served in the program committee on international conferences, such as ACL, EACL, RANLP, or COLING. She is a member of the Spanish Society for Natural Language Processing (SEPLN) and she has participated in a number of national and EU-funded projects. She has also been collaborating with international groups at the University of Wolverhampton (UK), the University of Sheffield (UK), the University of Edinburgh (UK), and the Lorraine Research Laboratory in Computer Science and Its Applications in France. Since 2009 she has been involved in teaching activities at the University of Alicante. Specifically, for the degrees in computer science engineering and multimedia engineering and for the master’s programme in information technologies and English and Spanish language for specific purposes, involving 200 teaching hours per year.

GUMCARS: GENERAL USER MODEL FOR CONTEXT-AWARE RECOMMENDER SYSTEMS

Sergio INZUNZA, Reyes JUÁREZ-RAMÍREZ
Samantha JIMÉNEZ, Guillermo LICEA

Universidad Autónoma de Baja California
Calzada Universidad 14418, Tijuana BC, 22390 México
e-mail: {sinzunza, reyesjua, sjimenez, glicea}@uabc.edu.mx

Abstract. Context-Aware Recommender Systems (CARS) are extensions of traditional recommender systems that use information about the context of the user to improve the recommendation accuracy. Whatever the specific algorithm exploited by the CARS, it can provide high-quality recommendations only after having modeled the user and context aspects. Despite the importance of the data models in CARS, nowadays there is a lack of models and tools to support the modeling and management of the data when developing a new CARS, leaving designers, developers and researchers the work of creating their own models, which can be a hard and time-consuming labor, and often resulting in overspecialized or incomplete models. In this paper, we describe GUMCARS – a General User Model for Context-Aware Recommender Systems, where the main goal is to help designers and researchers when creating a CARS by providing an extensive set of User, Context and Item aspects that covers the information needed by different recommendation domains. To validate GUMCARS, two experiments are performed; first, the completeness and generality of the model are evaluated showing encouraging results as the proposal was able to support most of the information loaded from real-world datasets. Then the structural correctness of the model is assessed, the obtained results strongly suggest that the model is correctly constructed according to Object-Oriented design paradigm.

Keywords: User-model, context-aware, recommender-systems, cars, GUMCARS

Mathematics Subject Classification 2010: 68-N01, 68-N30, 68-T30

1 INTRODUCTION

A recent research trend in Recommender Systems (RS) is the inclusion of context information in the recommendation algorithms, as contextual information has been proved to help increasing the prediction accuracy of recommender systems [24, 86]. This type of RS are known as Context-Aware Recommender System (CARS). Contextual information plays an important role in CARS, as user behavior is affected by the user current context [16], e.g., time, location, mood, and weather. CARS are based on the idea that similar users, in similar contexts, like similar items, and that user's preferences change according to his/her contextual situation [39]. Therefore, in CARS, as in most personalization systems, a user model is an essential component used to store the information about the user, his/her context, and interactions with the systems, which can later be used to adapt and personalize the system in order to improve the experience of the user in future interactions [49, 37].

Despite the advances in context information management [73], the design and development of context-aware systems, such as CARS, remain significantly more challenging than traditional systems, especially without supporting tools that facilitate this process, as to add context-aware capabilities to a software system brings design and development overhead inherited from the complexity of managing (acquiring, aggregating, storing) the user and context information [80].

Nowadays there is a lack of tools that support and facilitate the development of CARS [44], especially infrastructures for user and context information management [6, 49], which leaves developers and researchers the work of designing and implementing their own context-aware models to manage the information needed by recommendation algorithms, based on their knowledge and with no model to use as a reference, often resulting in overspecialized inefficient or incomplete models [80].

Even when some proposals of user model (e.g. [40]) and context model (like [89]) exist in the literature, they are not designed specifically for recommender systems, which means that such models do not consider key information for recommender systems, like items' data or ratings history. Also, some proposals are too abstract, designed at ontological level (e.g. [94]); others present the information categories, but not the specific attributes of user or context (e.g. [89]), and getting them to implementation implies a lot of work.

Summarizing, a context-aware user model that can be used in a range of CARS domains, which could improve CARS designer and developer efficiency, providing a data structure to manage the information of the context, user and his/her interaction with the system has not yet been proposed in the literature.

In this paper, we propose a General User Model for Context-Aware Recommender Systems (GUMCARS), that addresses the above-described problem providing a generic model for CARS information that can be used by CARS designers and developers to support the information for their CARS implementation, or as reference model that can be easily adapted to specific project needs.

GUMCARS proposes a taxonomic categorization of the information used by CARS, and provides an extensive set of User, Context and Item aspects that covers

the domain recommendation most commonly found in CARS literature, as well as the relation between these information elements resulted from the interaction of the user with the system. The result is a user model for context-aware recommender system that aims to achieve balance between *Completeness* so it can be used into a CARS systems with minimum modifications, and *Generality* so it can be extended to suit specific project needs.

The main contributions of this paper are:

1. From a *context-aware recommender systems* perspective, this paper presents a taxonomy that organizes the information needed by CARS systems to perform recommendations.
2. From a *user modeling* perspective, GUMCARS presents a general user model for context-aware recommender systems that can be adapted to suit specific needs, or used as a basis for future user model developments.

The remainder of this paper is structured as follows. Section 2 presents the related work. Section 3 presents the proposed taxonomy of CARS information. Section 4 describes a generic user model for CARS, Section 5 presents an experiment performed to assess GUMCARS completeness, and then Section 6 describes another experiment where the structural correctness of the proposal is evaluated. Finally, Section 7 presents the conclusions and future work.

2 RELATED WORK

In order to generate relevant recommendations, CARS needs to model the information of the user, the interaction of the user with the system, and the context where the interaction takes place [53].

A user model is defined as the knowledge about the user, explicitly or implicitly encoded, which is used by the system to improve user interaction [78].

The recent advance in technology that offers “anytime, anywhere, anyone” computing, has enabled software system to acquire more information about the user and his/her surroundings, which introduced the challenge of context-aware user modeling. A user model can be considered context-aware if it can express aspects of the user’s contextual situation and helps software systems to adapt their functionality to the context of use [83].

A commonly used definition of context comes from Dey and Abowd [28], they define context as any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves. From an *infrastructural* perspective, context provides computing devices with information about their environment as provided by other system components. In order to provide such information, context needs to be classified in different ‘types’ or ‘dimensions’ of context, e.g. physical, computational, etc.

Dourish [30] introduces a taxonomy of context, according to which context can be classified into *representational* or *interactional* view. The representational view assumes that the contextual attributes are identifiable and known a priori, and hence, can be captured and used within the context-aware applications. In contrast, the interaction view assumes that the user behavior is induced by an underlying context, but that context itself is not necessarily observable. As the interactional view of context is an approach borrowed from psychology [3] that considers context non-observable, it cannot be used to explicitly store the context information into a computer system. Therefore, this work is based on the representational view of context, as this approach allows to identify context attributes and to model it.

The areas of user modeling and context representation are well-established research topics by themselves. Next we review some of the most related publications on modeling user or context information in software systems.

Heckman [40] introduced the General User Model Ontology (GUMO) for the uniform interpretation of distributed user models. GUMO is intended to represent the information of the user for adaptive systems, the proposal presents a long list of user aspects to be considered in such systems organized into 12 *Basic User Dimensions*. As GUMO is created at an ontological level, it does not present further organization of the user characteristics or any architecture or implementation design that could help CARS developers in their effort of designing a context-aware user model.

Kaklanis et al. [52] proposed another user model in their efforts of the Virtual User Modeling and Standardization 'VUMS'. Their proposal aims to model user preferences for graphical interfaces, as well as some cognitive and physical human abilities. As the proposal is aimed at modeling people with disabilities and elderly people, the user aspects are limited to such interests. Therefore, this model would be of little help when designing a context-aware user model for CARS.

Jawaheer et al. [49] classify the different types of user feedback as a source of information for user modeling in recommender systems. However, they do not describe what user or context information must be included in such a model. They conclude the work with a series of future research challenges, including the need for a unified user model for recommender systems.

With respect to context modeling, Zimmermann et al. [98] defined five fundamental categories for context information: Activity, Time, Relations, Individuality, and Location. They describe such categories as the design space of context models for context-aware application to build upon. Later on, Verbert [89], through an extensive survey on context-aware systems, increases the number of categories for context information up to 8, namely, Computing, Location, Time, Physical Conditions, Activity, Resource, Social Relations, and User. Unlike Zimmermann, Verbert presented a series of subcategories for each context category, e.g., Computing is categorized into *software*, *hardware*, and *network*. However, a low-level detail of what information about context designers and developers should consider when creating their own context-aware user model has not been described by either proposal.

As the need for tools that help in the design and development of CARS is known in the literature, some proposals have emerged proposing tools for that matter.

Mettouris and Papadopoulos [61] present a tool designed to facilitate the development of reusable user models for CARS. The researchers publish the proposal as a web application, where developers can describe the context aspect that will be included in their model design. Even when their works [61] help developers as a tool to enlist context aspects and the possible values of each aspect, the work of identifying the aspects that will be included in the model is left to designers and developers. This tool does not allow to specify the data type of each aspect or relations between aspects, neither the proposal uses a formal or semi-formal notation (such as UML or XML) to help to take their list of aspects closer to a model design. Mettouris proposals [61] differ from GUMCARS proposal as follows: their tool is intended as a place to register, test and share a list of context aspects that later can be used to create a model, and our proposal is closer to the design and implementation phases as it presents the conceptual design and the UML model of a context-aware user model.

3 CARS INFORMATION TAXONOMY

In a relatively short time of CARS existence, a lot of different proposals have been created [74], either to test new algorithms or to measure how including certain information can help the recommendation process to generate better results. In most cases, each proposal arbitrary selects which information to take into account.

In order to create a general user model for CARS, and before the identification of what attributes will be included in our general model, we needed to define a classification of the concepts involved, that can be used by the model as a basis to structure the information it contains.

In this section, we describe a taxonomy proposal of the information that CARS use to generate context-based predictions. This taxonomy represents one of the contributions of our work, as it organizes the high-level information of the user, context and items that CARS need to work with, and, to the best of our knowledge, is the first taxonomy for CARS information.

The taxonomy comprises four categories of information:

1. User information,
2. Context information,
3. Activity information, and
4. Item information.

Next, each category and its corresponding sub-categories are described.

3.1 User Information

This top level category represents all the information of the user that describes him/her as a person, as such information is heterogeneous, and based on [40], we further organize the user category into eight subcategories: *Physiology*, *Contact information*, *Personality*, *Emotion*, *Role*, *Interest and Preference*, *Demographic*, *Mental*.

Compared to Heckman’s [40] list of 12 dimensions of user information (contact information, demographics, ability and proficiency, personality, characteristics, emotional state, physiological state, mental state, motion, role, nutrition, and facial expression); we do not consider a category for user’s characteristics as we consider (supported by Heckman’s [40] itself), that there is no clear separation between personality traits and characteristics; we propose to support both information in the *Personality* subcategory. Heckman uses the ability and proficiency dimension to support information about abilities and disabilities of the user, such information is supported in the *Physiological* subcategory of our taxonomy. Heckman uses the motion user dimension to represent whether the user is *walking*, *sitting*, *goingUpStairs*, etc. We consider such information to be part of the activity of the user, and is considered in the *Activity* information category (described in Section 3.3). The last dimensions we do not include in our taxonomy are *Nutrition* and *Facial expressions*, as we considered them very specific, this can increase the computational complexity of the model and most important, will make the model more cumbersome and harder to understand.

We also include a category called *Interest and Preference*, as CARS literature expresses that end users will consider useful a recommendation only if it fits in his/her interests and preferences [74].

Next, the proposed organization of the user information category is depicted in Figure 1, then each subcategory is briefly described.

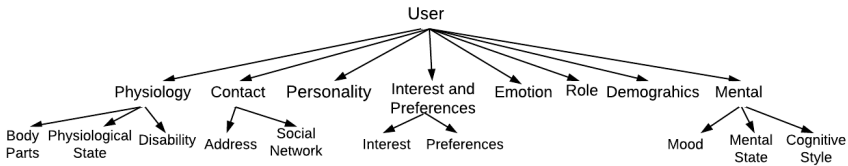


Figure 1. Proposed taxonomy for CARS user information

Physiology. Physiological aspects represent the information about user’s body and its functionality. This information is further subdivided into:

Body Parts: This represents information about parts of the human body. The body parts subcategory is subdivided into *Head*, *Hand*, *Foot*. And could the list of body parts of humans as described in [20].

Physiological State: Generic class to store information about physiological states, whose attributes are *name*, *level* and *isNormal*, these attributes can be used to represent states like respiration, blood pressure, heart rate, etc.

Disability: Superclass for specific disabilities classes. As designing a class for all the existing disabilities is outside the scope of this work, the model currently included only *ColorBlindness* and *Musculoskeletal* disability classes as a probe of concept, specific disabilities can be added as needed by specific projects.

Contact Information. For CARS, it is important to know whom is the user interacting with in the system, in order to provide tailored recommendations [74]. Contact subcategory refers to user aspects that identify a person (such as *name*, *last name*, and *email*), and the user's address. This information is available in most commercial systems since the user has to create an account and provide his data. Contact information is further subdivided into:

Address: used to represent the data related to the user's address, which could include information such as *house number*, *street name*, and *country*.

Social Network: represents the contact information of the user in social network sites that can be used by recommender systems to infer user preferences from their social activities as done by [7] and [81].

Personality. Describes permanent or very slow changing patterns that are associated with a person [41]. CARS can use the personality information to decide on what items will fit better to the user, for example, [14] uses the user personality as an important factor in their travel CARS. Certainly, the personality subcategory can be subdivided with the list of personality traits, but as there are several personality models in psychology like Big Five [50] and Cattell's traits of personality [56], we opt to include the basic elements in the taxonomy to support all the personality models, and not to stick to any particular model (as described in Section 4.2.1).

Emotion. Emotions are subjective human experiences [37]. Even when emotion information may look similar to a personality, there are different terms of duration of its contained information. Emotions tend to be closely associated with a specific event (context), and have a short duration of minutes up to an hour [40], while personality reflects long-term user characteristics.

Role. This user subcategory represents information about the role the user is playing at a certain moment. According to [43], the user can take several roles and frequently change between them, for example, a user can visit a town as tourist or businessperson and the CARS should be able to recommend different places to visit depending on such a role.

Demographic. This subcategory represent user's demographical information such as *age* or *family status*. This information can be used by CARS to improve the recommendations, for example [27] uses demographical information, such as

gender and *socio-economic* information in a context-aware music recommendation system. Demographical information along with interest and preference can be used by CARS to improve the interaction of the user with the system, using such information to adapt the user interface accordingly as presented in [56].

Mental. Used to describe the user's state of mind, this subcategory is subdivided into *mood*, *mental state* and *cognitive style* subcategories that can be used by CARS to provide more tailored recommendations to users. For example [7] uses the user *mental stress*, while [22] uses the user mood in their CARS.

Interest and Preference. This user information category allows to explicitly store user preferences for, or interest in some type of items. In a CARS presented in [95], the interest of the user is used to recommend places and events to visit accordingly, a user with music interest will be recommended to attend concerts, while a user with shopping interest will be recommended to visit near shopping malls. Similarly the CARS of [95] also implicitly collects the preferences of the users, which help the system to decide what specific type of place to recommend, in the music-lover user example, the system could use the preferences of the user to decide whether recommend country, rock or classical events. This user information category is further subdivided into:

Interest: Represents the interest of the user in certain topics or items.

Preference: Represents the preference of the user for certain items in relation with a certain context.

3.2 Contextual Information

This category represents information about the environment that surrounds the user, as well as other information about the items or the recommendation system itself that can be used to characterize the situation of the elements considered in the CARS. We organize the context information into the following six subcategories: *Computing*, *Location*, *Time*, *Physical condition*, *Resource* and *Social relation*.

Compared to the organization of contextual information proposed in [89], that describes a list of 5 types of context like *Computing*, *Location*, *Time*, *Physical Condition*, *Activity*, *Resource*, *User*, and *Social Relations*; we do not consider a subcategory for the activity information, as we consider such information in a category at a higher level. We do not consider the user information to be part of the context category, as we have a more detailed categorization of the user information, and it is supported in a top-level category of the proposed taxonomy. The similarities and differences between the matching types of context from [89] and the proposed taxonomy are discussed along with the description of each subcategory.

Our taxonomy supports other proposals of context information organization, like the one presented in [11] that argues that is important to include interactions between the environment (supported in *Computing* and *Physical Conditions* subcategories), the user (supported in *User* category), their tasks *Activity*), and other users (*Social Relation*) as part of the contextual information.

Next, the proposed taxonomy for context information category is depicted in Figure 2, and then each subcategory is briefly described.

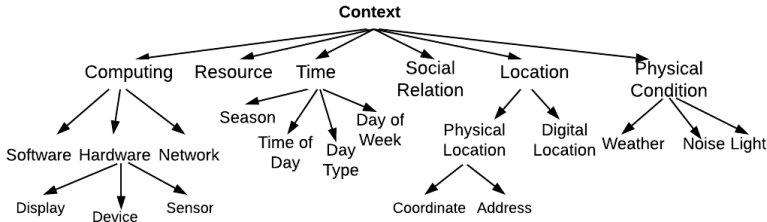


Figure 2. Proposed taxonomy for contextual information user by CARS

Computing. This subcategory represents the information that describes computational elements. Following [89], this subcategory is further divided into:

Software: This sub-subcategory describes the characteristics of software systems that may be available to the user under a certain context, or software systems that the user is interacting with at certain moment. Let us consider a CARS that is recommending persons to a friend on different social networks, for such a system, the information about what applications the user have installed on their smart-phone may be very useful.

Hardware: Comprises information about the hardware as a physical item, such as *Vendor* and *Model*. We follow the Composite Capabilities/Preference Profile (CC/PP) [93] to further organize the hardware sub-subcategory into *Sensors*, *Display*, and *Device* that represent all the computing devices (like tablet, laptop and smart-phone) and include information of the minimum components of the computer device like *RAM*, *Processor* and *Storage*.

Network: Information related to properties of the networks that are being used by computing devices. As for uses of *Network* information, CARS can take into account if the user’s mobile device is currently on WiFi connectivity or via the cellular network in order to recommend High Definition or Standard Definition videos.

Resource. The resource information models relevant characteristics that describe elements and the services that items or places provide, for example, if a restaurant has wheelchair ramps, or if a hotel has transportation to and from the airport. The resource sub-subcategory of context can also be used to describe digital resources that are relevant to users, for example, a learning material can be used by a CARS to recommend learning topics to students.

Time. Time subcategory represents information about the time of the day, time of the week or time of the year (season), this information allows CARS to record the moment that some actions took place. The use of *time* information in CARS

is very common, as its values are easy to gather and have a deep impact on the user decisions [38].

Social Relation. This subcategory of contextual information refers to social associations or connection between the user of the system and other persons that may or may not be part of the CARS. The user social relations can help CARS to tailor items specific to a current user companion, or to infer user preferences based on the knowledge the system has about other users related to him/her [79].

Location. In the proposed taxonomy, the location subcategory refers to information that relates a user or an item with a physical or a digital location. Location information is used by CARS to recommend the user items like restaurants or movie theaters near him/her. Location is so often used in CARS, that there is a whole research topic called location-aware recommender systems [32].

Physical Location: Comprises information used to identify a specific point in the world, using the coordinates (latitude and longitude) or an address.

Digital Location: Refers to information of where a digital resource (like a web page, video or song) can be located on the Internet. A common example of a digital locator is a URL (Uniform Resource Locator) that specifies the location of a computer in a network.

Physical Condition. Describes the environmental conditions where user or items are situated at a certain point in time. Physical condition describes physical conditions external to user or items like the level of crowdedness and the level of traffic. This subcategory is further subdivided in:

Weather: Represents weather information that CARS can use to better tailor suggestions, for example, the weather is important when recommending places to visit as recommending an outside place with a rainy weather may not be well perceived by users [26].

Noise: Refers to information about the level of noise at a certain place, at a given time.

Light: Used to represent the light level as well as the light source of the physical environment.

3.3 Activity Information

This information category represents the activity performed by the users of the system and relates such an activity to the user and the context.

The activity of the user can help CARS perform more accurate recommendations, for example, Spotify, a music service and recommender system, identifies when the user is running or biking and plays inspirational music to keep them going.

Unlike other proposals like [40, 89] that consider the activity information as part of the contextual information, we create a category for such information at the same level of *User* or *Context* information.

3.4 Item Information

This category represents the information about the items of the CARS catalog like *Video*, *Book* and *Audio*. The importance of items information for CARS is paramount as items along with the transaction log are the base information for the CARS algorithms [74, 4].

4 THE GENERAL USER MODEL FOR CONTEXT-AWARE RECOMMENDER SYSTEMS

This section introduces GUMCARS, a General User Model for Context-Aware Recommender Systems that along with the information taxonomy, represents the main contributions of this paper.

The goal of GUMCARS is to structure the user, context, and items aspects that can be used by CARS systems either as the input for recommendation algorithms or by the system itself to improve interaction with the user. GUMCARS can be used by researchers as a basis for future model developments, as it contains a holistic view of the information used by the research community to generate context-based recommendations, by designers of CARS as the model can serve as a guide when selecting the information aspects to create their own model, and by CARS developers as the model is expressed not only from a conceptual perspective (the taxonomy), but also is presented from a software architectural point of view, and is ready to support the information needed by the most common recommendation domains.

Based on Dourish [30] representational view of context, GUMCARS is formed by a finite set of user, context, and item attributes. The methodology used to gather the specific aspects that form GUMCARS is described in the following subsection. Then, the proposed architecture for GUMCARS is presented.

4.1 Gathering of CARS Information Aspects

As GUMCARS intends to structure a finite set of user, context and item aspects that can be used by CARS as input information for the recommendation algorithm. In the previous section the proposed taxonomy for the information organization was described, as the taxonomy is an organization of concepts and does not contain the specific attributes that will support the CARS information, the next step towards the creation of GUMCARS structure was the identification of such specific information aspects, for which a systematic literature review of CARS was performed and presented in [46].

For the Systematic Literature Review (SLR), the Kitchenham [54] methodology was followed, the SLR reviews the user modeling, context-awareness, and CARS literature to respond the following questions:

1. Which user aspects have been used in CARS literature as input information for the recommendation algorithms?

2. Which context aspects have been used in CARS?
3. What items are being recommended by CARS?
4. What aspects about such items are being used by the recommendation algorithms?

The execution of the SLR consisted in defining the search queries, executing such queries in the selected literature sources, gathering all the matching publications and the grooming of the results based on a series of predefined filters that goes from reading the title to reading the entire publications. The results obtained from SLR was an extensive list of aspects (like address, mood, companion, season, battery level, show size, etc.) and aspects values (like partner, family, and friends for the companion aspect) that publications have used in their CARS proposals. For a more detailed description of the followed steps or the obtained results please refer to [46].

4.2 The GUMCARS Model Architecture

This section describes the architectural view of the model, compared with the conceptual view (taxonomy), the architectural view goes deeper in detailing what specific aspects are considered for each information category. The architecture of GUMCARS is designed following the object-oriented (OO) paradigm [12], and the semi-formal language Unified Modeling Language (UML) [13].

GUMCARS architecture uses the same four top-level categories proposed in the taxonomy, which are called (packages) in the UML notation, the sub-categories and sub-subcategories are *classes*, and the aspects are *class attributes*.

The rest of this section describes the proposed architecture for each category.

4.2.1 User Aspects Considered in GUMCARS

The UML class diagram of user information is presented in Figure 3, and the counting of classes and attributes considered for this package is presented in Table 1. In addition, representative examples of what user aspects were found through the SLR, and how they were organized in the proposed model are given. For a detailed documentation of each class and attribute please refer to http://bit.ly/GUMCARS_user, where the GUMCARS model is used as the core component of the user modeling framework for CARS.

For the model architecture, the *body part* sub-subcategory presented in the taxonomy is specialized into four types of body parts: *Foot*, *Arm*, *Head* and *Hand*. Aspects of this classes are being used in CARS, for example *Shoe Size* which is an attribute of the *Foot* class, is considered by [44], while *Ring Size* and *Bracelet Size* attributes mapped to *Hand* and the *Arm* classes, respectively, can be used by a CARS recommending fashion accessories like the proposed in [57].

Contact information like *Address* and *email address* are used by [84]; the physical *address* and *phone number* are also attributes of the *Contact* class as used by [71].

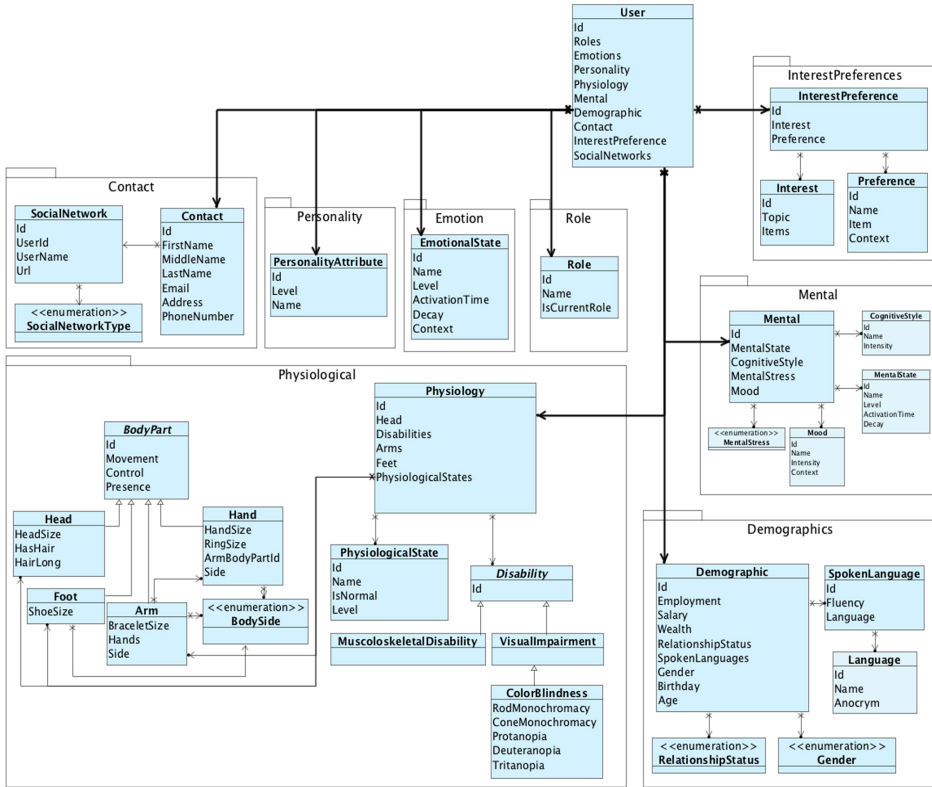


Figure 3. User aspects considered in GUMCARS

The personality information is used for example by [51] as a way to improve the music preference predictions in their CARS. GUMCARS support personality information containing base attributes such as *Level* and *Name* that can support different personality models or can be specialized to support a specific personality model.

The demographics information was commonly used in the reviewed literature, for example [92, 17] use *age*, *gender*, and *employment* among others. In this package, *Spoken Language* class was added to support cases like [7] where the *language* that the user speak is considered by their CARS.

4.2.2 Context Aspects Considered in GUMCARS

In this section, Figure 4 presents UML class diagram for the context information, and Table 2 presents the number of classes and attributes considered in GUMCARS; then, representative examples of what context aspect were found through the SLR,

Subcategory	Classes	Attributes
Physiology	12	33
Contact	3	23
Personality	1	4
Interest and Preferences	3	10
Emotion	1	6
Role	1	3
Demographics	5	28
Mental	5	20
<i>Total</i>	31	127

Table 1. Number of classes and attributes of GUMCARS for User information

and how they were organized in the proposed model are given. A detailed documentation of each class can be found at http://bit.ly/GUMCARS_context.

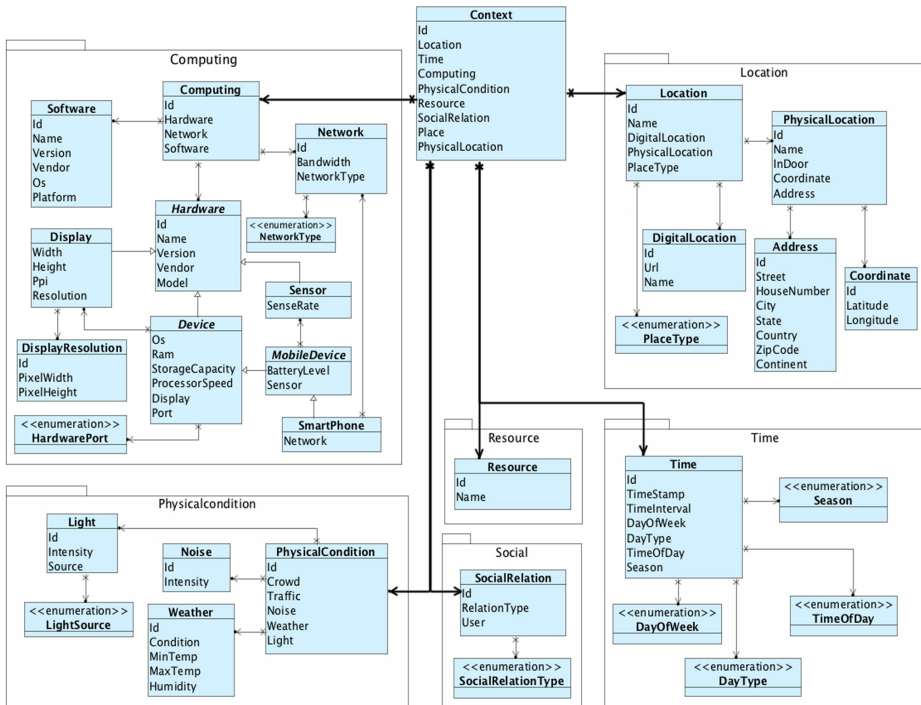


Figure 4. Context aspects considered in GUMCARS

Subcategory	Classes	Attributes
Computing	12	67
Resource	1	2
Time	5	33
Social Relation	2	11
Location	6	40
Physical Condition	5	19
<i>Total</i>	31	172

Table 2. Number of classes and attributes of GUMCARS for context information

The information of computing devices like smart phones is used by [26]. Also [77] considers that storing the battery level of mobile computing devices is important for CARS. To support this information, the *device* class of the *Computing* package, is further specialized into *mobile device* class, also *sensor* class is created as an specialization of *hardware* class.

Time is one of the most used contextual information in CARS, as found in the SLR. For example, [26] uses the *time of the day* to tailor news, [92] uses *weekday* and *weekend* categorization, and [71] uses the type of day (e.g. holiday or weekend) as valuable information in their CARS. GUMCARS further organizes the *Time* package adding *Time of Day*, *Day of Week*, *Season* and *Day Type* classes.

The aspects referring to other packages like the *Social Relation*, *Location* and *Physical conditions* were also found in the SLR, for example [16] considers the user companion when recommending movies, [8] uses the location of the user (expressed in latitude and longitude) when recommending places to visit, and [71] uses the weather condition as valuable information in their CARS.

4.2.3 Activity Information Considered in GUMCARS

Recommender systems use the information of what item the user consumed and in what context such activity took place, as base information for the prediction generation [74]. A good example of how all this information is used is presented in [71], their CARS consider the activity of the user (running, driving, standing, etc.), along with some contextual information (like location, time of day, day of week) and information about what song the user is currently listening, to decide what to recommend next.

With the *Activity* package, GUMCARS supports the past and current activity of the user, the item consumed in such a transaction, the contextual information, and the rating given by the user. GUMCARS contains a total of 5 classes and 18 attributes to represent the activity information as depicted by the UML diagram in Figure 5. For a detailed description of each class and attribute please refer to http://bit.ly/GUMCARS_act.

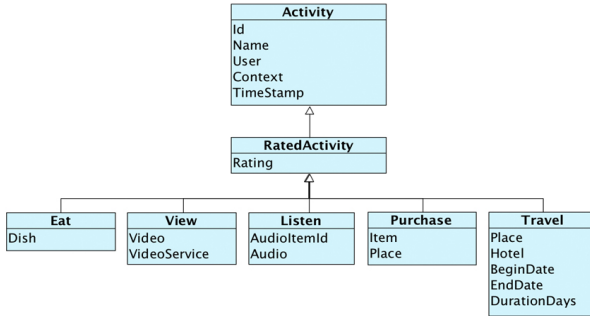


Figure 5. Act aspects considered in GUMCARS

4.2.4 Items Information Considered in GUMCARS

In addition to User, Context and Activity information, GUMCARS also includes the base aspects related to the *Items* that a CARS could recommend. These aspects are included in an *Item* package, which contains classes to represent the elements the reviewed publications are recommending as shown in Figure 6. The item package contains a total of 8 classes and 41 attributes to represent the information about the items. For a UML diagram of the classes and a detailed description of each class and attribute, please refer to http://bit.ly/GUMCARS_items.

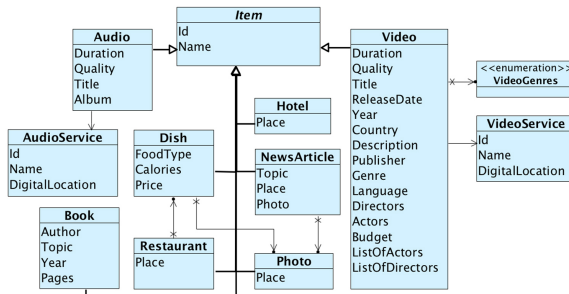


Figure 6. Item aspects considered in GUMCARS

Next sections present two evaluations performed to the GUMCARS model, the first one is dedicated to a mathematical evaluation using complexity metrics intended to evaluate the *Correctness* of the model. The second describes a practical evaluation that evaluates the *Completeness* and *Generality* of GUMCARS by measuring its ability to hold data from real-world datasets.

5 EVALUATING GUMCARS COMPLETENESS AND GENERALITY

According to [58], the quality of a conceptual model can be categorized in

Syntactic: the model contains only statements that are valid in the language;

Semantic: how well the model represents the important elements of the domain;
and

Pragmatic: how the intended audience interprets the models.

In this experiment, we perform a *Semantic* quality evaluation of GUMCARS. The *Semantic* quality of the model (M) with respect to the target domain (D), can be defined by two quality attributes: *Validity* and *Completeness* [58].

Validity ensures that the elements included in the model are relevant to the modeled domain. As the GUMCARS model is based on the SLR [46] performed explicitly in the target (CARS) domain, all the included elements in GUMCARS were stated as important for the domain by their respective publications.

Therefore, in this iteration, we focus on evaluating the *Completeness* of GUMCARS, which refers to how many of the relevant elements of the domain are included in the model. In [64], *Completeness* is also defined as *whether the model supports all the information required by target systems*. This experiment also allowed us to validate the *Generality* of the model that according to Moody [65] describes how wide the application scope of the model is. In this work, we define *Generality* as the ability of the model to support data from different domains.

5.1 Materials

We used a total of 8 datasets gathered from the literature. We selected only datasets that were not considered during the creation of the model, that were created for recommendation purposes, and, that contain data about the 3 main elements (*Users*, *Items* and *Context*). The datasets correspond to different item domains, with different quantity and type of information. Table 3 shows the details of each dataset.

In addition, to perform this evaluation, an implementation of GUMCARS is used, such implementation is part of a wider project where the proposed model is used as part of a CARS modeling framework (<http://um4rs.com>). Such implementation was created by exporting the UML representation of the model to a C# code library. We consider the code to be just another representation of the model, as there is no change between the architectural view of the model and the implemented code. In this experiment, as well as in the one described in Section 6, we opted to use the implemented model as a way to automatize the evaluation and avoid biases.

Dataset	Description	Type	Attributes				Instances
			User	Context	Item	Total	
Japan	Restaurant dataset by [70]. Describes the restaurant, time, environment, and user's social information.	Food	3	14	17	34	800
Trip Advisor	Hotel booking dataset by [96]. Contains information about the hotel, the context, and the user.	Hotel/Travel	3	2	4	9	14 176
Expedia	Travel booking dataset from expedia.com. Contains 150 aspects about the travel destination, aspects of the user and the context.	Hotel/Travel	4	15	154	173	62 100
STS	A travel dataset by [31]. Is rich in user and context information including Big Five personality traits.	Hotel/Travel	8	15	4	27	2 535
DePaul Movie	Movie dataset by [97]. Contains information about the time and companion of the user.	Movie	1	4	1	6	50 043
LDOS CoMoDa	A movie rating dataset by [55]. This dataset contains emotional information about the user while rating/watching the movie, as well as movie and context information.	Movie	9	8	14	31	2 296
Concert Tweets	A music dataset by [1]. This dataset contains information about concert events, the user, and the context of the user when attending the events.	Music	1	7	4	12	249 470
InCar Music	A music dataset by [9]. This dataset contains information about the song, the mood and driving style of the user while listening to the song, as well as other contextual information.	Music	2	8	8	18	4 012

Table 3. Information of datasets loaded into GUMCARS

5.2 Method

First, we mapped all the possible aspects found in each dataset to their corresponding attribute in GUMCARS classes. This enables us to identify what dataset aspects are supported, and what aspects are not supported by GUMCARS. An aspect is considered to be supported if GUMCARS contains an equal or similar attribute in their classes. For example, *Social* aspect of LDOS CoMoDa dataset refers to *the companion of the user to watch the movie*, and was mapped to *Context.SocialRelation.RelationType* enumerator in the model, that represents the same information.

Then, a simple script was developed for each dataset, where the dataset file was read and its contained features are mapped into GUMCARS implemented representation, following the mapping created in the first step. During this step, no changes were made to GUMCARS code.

The *Completeness* of GUMCARS was calculated as the percentage of the total numbers of aspects contained in the dataset (D), that were successfully mapped into the model (M), i.e. $Completeness = M/D$. Each dataset aspect scores a 1 into the D variable if a corresponding attribute exists in GUMCARS, and all of its values were fully supported by the model. In cases where a corresponding attribute exists in GUMCARS, but some of the values present in the dataset were not supported by the model aspect, such attributes scored a 0.5 into D and considered as *Partially Supported*.

5.3 Results

A total of 31 user, 73 context, 206 items aspects, and 323 332 instances were contained in the 8 datasets, and were mapped (separately) into GUMCARS. Along the datasets, some aspects are repeated, being the most common *userId*, *itemId*, *rating* and *weather*. As a result, in this experiment, 22 unique *User* aspects were used, 64 *Context* aspects, and 194 *Item* aspects of 6 different types of items, namely: *food*, *hotels*, *travels*, *movies*, *concerts* and *songs*, were used.

For *Japan* dataset, 6 of the 32 attributes were not supported, and 1 attribute (*RelationType*) was considered partially supported, as 2 (*Boss* and *Subordinate*) of the possible 6 values could not be mapped to GUMCARS's *RelationType* enumerator. This result in a *Completeness* value of 0.81, which means that the 81% of the attributes were supported. We got *Completeness* of 89% for *TripAdvisor*, 96% for *Expedia*, and 70% for *STS*. For *DePaul Movie* we obtained a *Completeness* of 100%, and 90% for *LDOS CoMoDa*. In the *ConcertTweets* dataset, we were able to map all the contained attributes to GUMCARS, thus a *Completeness* value of 100%. Finally, for *InCar Music* dataset we obtained 83% of *Completeness* as 3 of its 18 attributes were not supported.

In general, GUMCARS supported the 93.55% of the user aspects, 75.34% of contextual aspects, and 96.12% of item aspects contained in the eight datasets used

in this experiment. The obtained *Completeness* for each dataset as well as the non supported attributes are presented in Table 4.

Dataset	Total Attributes	Not Supported	Partially	Completeness
Japan	34	6 Budget Num. of Male (Partners) Num. of Female (Partners) Lowest Age (Partners) Highest Age (Partners) Partner Status	1 Relation type	0.81
Trip-Advisor	9	1 User time zone	0	0.89
Expedia	173	7 Orig_destination_distance Is_package (is a travel package) Srch_adults (num of adults) Srch_children (num of children) Srch_rm (num of rooms) Cnt (number of similar events) Posa_continent (point of sale)	0	0.96
STS	27	8 distance knowledgeOfSurroundings budget travel goal means of transport POI's category1 POI's category2 POI's category3	0	0.70
DePaul Movie	6	0	0	1.00
LDOS CoMoDa	31	3 Physical (health condition) Decision (to watch the movie) Interaction (n-th interaction)	0	0.90
Concert-Tweets	12	0	0	1.00
InCar Music	18	3 DrivingStyle RoadType Sleepiness	0	0.83

Table 4. Results of mapping and loading dataset into GUMCARS model

5.4 Discussion

The average value of *Completeness* for the 310 aspects tested was a 0.88 of 1, with a standard deviation of 0.09, which strongly suggests that GUMCARS is capable of supporting most of the aspects considered by the datasets.

The lowest *Completeness* value for a dataset corresponds to *STS*, which scored a 0.70, as 8 out of the 27 aspects considered in the dataset were not supported by

GUMCARS. This dataset contains some very specific traveling aspects like *Travel Goal* and *Traveled Distance* that would have been difficult to foresee.

We consider that GUMCARS performed well in supporting the datasets, especially in user and item information with 94% and 96% of *Completeness*, respectively. The obtained 75% of *Completeness* for the context information, even when this is a good value, is lower than the obtained value for others datasets. We attribute this to the fact that recommendation system field just started to consider contextual information [2], and there is not an established standard of what context information works best for CARS. Also, the amount of information of the environment (context) that surrounds users is enormous, and trying to create a contextual model that encompasses everything is practically impossible. Nevertheless, we strongly believe that GUMCARS has the needed base to allow model designers and researchers to tailor or extend the model to their specific contextual needs.

In general, this experiment allowed us to evaluate to some degree the *Completeness* of the model by mapping into it different datasets with different characteristic obtaining encouraging results. Also, the fact that the dataset contains information from different domains, allows us to prove GUMCARS *Generality*, as it was able to support most of the aspects of the tested domains. As for the non supported elements, they were included after finishing the evaluations, and the increased version of GUMCARS will be published as open-source.

6 EVALUATING GUMCARS STRUCTURAL CORRECTNESS

Assessing the quality of a complex model is generally a difficult task [44], since GUMCARS is, to the best of our knowledge, the first user model designed specially to build Context-Aware Recommender Systems, a direct comparison with other proposals is difficult or even misleading.

As the goal of GUMCARS is to be used in CARS implementations to support their data, we considered important to assess some quality attributes of the *architectural* and *implemented* representation of the model. Therefore, in this experiment, we focus on evaluating the *Correctness* of GUMCARS, which is a very important quality attribute of conceptual models [58].

The *Correctness* of a model is defined as whether the model conforms to the rules of data modeling techniques [64]. Since GUMCARS was created following the Object-Oriented (OO) design technique, we used software quality metrics to assess the level of *Correctness* of the proposed model, which (according to [36]) allow software designers to evaluate the quality characteristics of OO models, objectively.

6.1 Materials Used

For this evaluation, the *Depth of Inheritance Tree*, *Number of Children* and *Coupling Between Objects* metrics from the (CK) metric suite [18] were selected, as these met-

rics that can be applied either to class diagram or to code [36]. We also considered important to assess the Maintainability of GUMCARS, as it expresses the easiness with which the model could be adapted to suit specific needs [25], therefore we also include the *Maintainability Index* metric in this evaluation. Next, the used metrics are briefly described.

- *Depth of Inheritance Tree (DIT)*: The depth of a class within the inheritance is defined as the maximum length from the class node to the root of the class hierarchy tree and is measured by the number of ancestor classes [18]. A low value of DIT implies less complexity in the model organization, a value between $1 \leq DIT \leq 6$ is desired [69].
- *Number Of Children (NOC)* Number of immediate sub-classes subordinated to a class in the class hierarchy [18]. The value of NOC represents the number of classes that inherit from a specific class. An optimal value for NOC is between 0 and 11, values from 12 to 28 are regular and a value greater than 29 is considered bad NOC [35].
- *Coupling Between Objects (CBO)* Coupling Between Objects, also known as Class Coupling measures the relationships between entities [19]. CBO is a measure of how many classes does a single class use. The CBO value threshold proposed by Microsoft [67] is a 0–9 range for optimal (green) CBO value, a 10–80 range for acceptable (yellow) value, and CBO greater than 80 is considered critical (red) value.
- *Maintainability Index (MI)* ISO/IEC 9126 [47] defines maintainability as “*the capability of the software product to be modified. Modification may include corrections, improvements or adaptation of the software to changes in the environment*”. To measure the *maintainability* of the GUMCARS we used the *Maintainability Index* (MI) metric, which measures maintainability by taking the size, complexity, and self-descriptiveness of the classes into account, resulting in a MI range between 0 and 100, where 100 represents the best possible value of maintainability of the system [68].

6.2 Method

The selected metrics could be applied either to UML class diagram performing the calculation by hand or to classes in code using tools to perform the calculations.

To avoid any bias in the metric applications, and to get the most objective results, we opted to apply the metrics to the code representation of GUMCARS (described in Section 5), using automated tools to perform the calculations.

The GUMCARS’s values for *DIT* and *NOC* metric were gathered using NDepend. For the *CBO* and *MI* metrics, Visual Studio IDE was used. Both tools gave a numeric value of each class for the evaluating metric. The results of each metric are loaded into SPSS for the data analysis.

6.3 Results

According to [18], a lower DIT value is preferred, with a threshold between 1 and 6 for an optimal value. This experiment revealed a maximum DIT value of 4, a mean value 1.41 with a Standard Deviation (SD) of 0.69, as can be seen in Table 5, placing GUMCARS inside the optimal threshold for this metric. The deepest leaf found in GUMCARS inheritance tree is *Hardware > Device > MobileDevice > SmartPhone* in the context package.

The NOC mean value of GUMCARS is 0.41 with an SD of 1.40 and a maximum number of 8 children for a single class, that correspond to *Item* class, as all the types of items considered in GUMCARS are dependent of *Item* class.

In this experiment, we got a CBO value of 1.42, with SD of 1.79 and a maximum value of 10. The maximum CBO value corresponds to *User* class, as it is dependent of 10 other classes, even when the 10 is outside the optimal value (as shown in Figure 7), it just occurred once, and the rest of the 85 classes fall between 1 and 7.

The last metric applied in this experiment is MI, the mean value obtained is 94.8, with SD of 2.89 and the lowest value of 90 with only 1 occurrence, and corresponds to *Item* class.

Next, Figure 7 shows the frequency distribution for the DIT, NOC and CBO metrics, and Figure 8 shows the distribution for MI values. Table 5 summarizes the results obtained from the application the quality metrics to GUMCARS. Then, Table 6 shows some representative examples of the raw data obtained from the metrics, showing some best and worst cases.

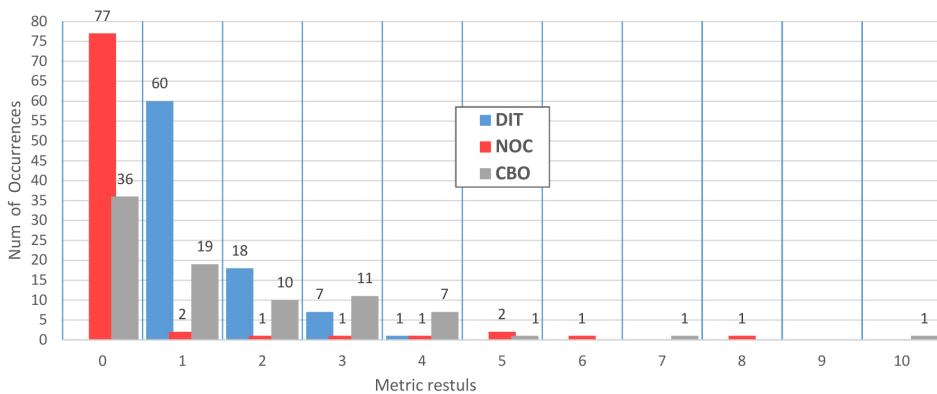


Figure 7. Frequency distribution of DIT, NOC and CBO

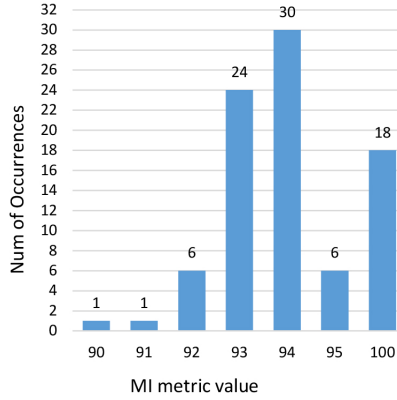


Figure 8. Frequency distribution of MI values

Metric	Min	Max	Mean	SD	Threshold	Better when	Tool
DIT	1	4	1.41	0.69	$1 \leq DIT \leq 6$	Lower	ND
NOC	0	8	0.41	1.40	$0 \leq NOC \leq 29$	Lower	ND
CBO	1	10	1.42	1.79	$0 \leq CBO \leq 80$	Lower	VS
MI	90	100	94.83	2.89	$0 \leq MI \leq 100$	Greater	VS

Table 5. Results of applying quality metrics to GUMCARS

6.4 Discussion

The low value obtained for DIT (1.41) reflects that the model organization is not complex, which will help GUMCARS to be understood and used by its target users (CARS designers, researchers, and developers). This DIT value supports *Correctness* with respect to the organization of the model classes.

For the NOC metric, we obtained a 0.41 with a maximum of 8, this places GUMCARS in the optimal values part of the threshold. The result of NOC strongly

Class	Location in Model	DIT	NOC	CBO	MI
User	(top level)	1	0	10	90
Context	(top level)	1	0	7	92
Item	(top level)	2	8	0	91
Activity	(top level)	1	6	2	93
RelationType	Context.Social	1	0	0	100
SmarthPhone	Context.Computing	4	0	4	95
MentalStress	User.Mental	1	0	0	100

Table 6. Representative examples of raw data obtained from the application of the metrics

suggests that the proposed model uses a correct level of abstraction, as there is not a single class with too many dependent children.

Coupling Between Objects (CBO) metric yielded a mean 1.42 value which reflects that GUMCARS has an optimal level of coupling between classes. From the 86 total classes of the model, only one lays outside the optimal value, namely the *User* class, that got a CBO of 10, placing it in the *good* part of the threshold. Certainly, the *User* CBO could be improved sub-dividing the class, but we need to consider the increase of complexity by adding another level on the inheritance tree, thus increasing the overall DIT value. Overall, the result of this metric supports the general *Correctness* of the proposal, showing that the model represents the intended domain using a correct abstraction level.

This experiment yielded a mean value of 94.83 for the MI metric, meaning that the proposal's correct organization of classes and attributes resulted in an optimal level of MI. Figure 8 shows that most of the classes have an MI of 93 and 94. The lowest value of MI found in the whole model was 90 that corresponds to *Item* class, as all the items that the model considers are dependent on this class. This means that any change made to the *Item* class will be propagated through other classes, and that designers need to consider this before making any change to *Item* class. Nevertheless, an MI value of 90 falls very close to an optimal value, and we consider it a good value considering that *Item* class is some of the cornerstones of the GUMCARS.

Overall, in this experiment we evaluated the *computational* representation of the proposal, obtaining very good results for all the quality metrics, which strongly suggest that GUMCARS contains a high level of structural *Correctness*, and can be used by CARS designers and developers to support the data needed for their systems, and in cases where the model does not consider some specific aspect(s), it will support such a modification with minimum possibilities of affecting the overall structure of the model.

7 CONCLUSIONS AND FUTURE WORK

In this paper, a Generic User Model for Context-Aware Recommender Systems was described in terms of: the taxonomy to organize the information, the model structure, an extensive set of user, context, and item aspects needed by CARS to generate recommendations. The goal of GUMCARS is to help CARS designers, developers and researchers presenting them a base data model that can use directly or as a reference when developing their own CARS.

The proposed model was evaluated using some real world datasets from CARS domain, with the intention to assess the model completeness and generality. The obtained results from this experiment strongly suggest that GUMCARS has a great degree of completeness as 88% of the dataset aspects were supported. This experiment also showed that the model has a great level of generality, being able to support the data from 6 different domains of CARS. This implies that the

model has the needed attributes and classes to support the information that CARS needs.

Another evaluation performed to GUMCARS allowed us to assess the *correctness* respect to the structural quality of the model. The application of 4 quality metrics for object-oriented models design showed that GUMCARS contains an optimal level of *Depth of Inheritance*, *Number of Children* and *Maintainability*, and an acceptable level of *Class Coupling*. Overall, the results from the second experiment, strongly suggest that GUMCARS has a high level of structural *correctness*, which implies that the attributes are organized in the precise classes and that proper relationships between classes are present in the model.

The main limitation of this work is that the proposed model is based on CARS literature, therefore the model contains only the elements that CARS researchers considered in their publications. Based on the results obtained from the evaluations here presented, we are sure that the model will work as expected; nevertheless, we acknowledge that a dynamic validation of the model is needed to support the correct functionality of the model.

Future work on the GUMCARS proposal is aimed to test the model in a real implementation of CARS systems. Currently, there is ongoing work to create an entire modeling framework for CARS using GUMCARS as a core to organize the information. Another interesting research path is to put GUMCARS outside CARS domains, to see what modifications the model needs to support the information needed by an adaptive system in a Human-Computer Interaction area, or by an Intelligent System in a Medical field.

GUMCARS proposal contributes to user modeling area, presenting the first user and context model designed specifically for context-aware recommender systems. The model has an impact in the real world as it is ready to be implemented in CARS, and enables to avoid the work of structuring and organizing the systems information, saving developers and designer such work. We firmly believe that GUMCARS also contributes to the research community as it can be used as a reference for researchers to create more detailed or more specific user models. For example, a research seeking to create a computational model to represent human disabilities can take advantage of the *Physiological* package of GUMCARS.

REFERENCES

- [1] ADAMOPOULOS, P.—TUZHILIN, A.: ConcertTweets: A Multi-Dimensional Data Set for Recommender Systems Research. 8th ACM Conference on Recommender Systems (RecSys 2014), Foster City, Silicon Valley, USA, 2014.
- [2] ADOMAVICIUS, G.—TUZHILIN, A.: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering, Vol. 17, 2005, No. 6, pp. 734–749, doi: 10.1109/TKDE.2005.99.

- [3] ADOMAVICIUS, G.—TUZHILIN, A.: Context-Aware Recommender Systems. Proceedings of the 2008 ACM Conference on Recommender Systems (RecSys '08), 2008, p. 335–336, doi: 10.1145/1454008.1454068.
- [4] AGGARWAL, C. C.: An Introduction to Recommender Systems. In: Aggarwal, C. C.: Recommender Systems: The Textbook. Springer, 2016, pp. 1–29.
- [5] ALBERT, W.—DIXON, E.: Is This What You Expected? The Use of Expectation Measures in Usability Testing. Proceedings of Usability Professionals Association Conference, Scottsdale, AZ, 2003.
- [6] ALEGRE, U.—AUGUSTO, J. C.—CLARK, T.: Engineering Context-Aware Systems and Applications: A Survey. Journal of Systems and Software, Vol. 117, 2016, pp. 55–83, doi: 10.1016/j.jss.2016.02.010.
- [7] ALHAMID, M. F.—RAWASHDEH, M.—AL OSMAN, H.—HOSSAIN, M. S.—EL SADDIK, A.: Towards Context-Sensitive Collaborative Media Recommender System. Multimedia Tools and Applications, Vol. 74, 2015, No. 24, pp. 11399–11428.
- [8] BAGCI, H.—KARAGOZ, P.: Context-Aware Location Recommendation by Using a Random Walk-Based Approach. Knowledge and Information Systems, Vol. 47, 2016, No. 2, pp. 241–260, doi: 10.1007/s10115-015-0857-0.
- [9] BALTRUNAS, L.—KAMINSKAS, M.—LUDWIG, B.—MOLING, O.—RICCI, F.—AYDIN, A.—LÜKE, K.-H.—SCHWAIGER, R.: InCarMusic: Context-Aware Music Recommendations in a Car. In: Huemer, C., Setzer, T. (Eds.): E-Commerce and Web Technologies (EC-Web 2011). Springer, Berlin, Heidelberg, Lecture Notes in Business Information Processing, Vol. 85, 2011, pp. 89–100.
- [10] BANGOR, A.—KORTUM, P.—MILLER, J.: Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. Journal of Usability Studies, Vol. 4, 2009, No. 3, pp. 114–123.
- [11] BEALE, R.—LONSDALE, P.: Mobile Context Aware Systems: The Intelligence to Support Tasks and Effectively Utilise Resources. In: Brewster, S., Dunlop, M. (Eds.): Mobile Human-Computer Interaction – MobileHCI 2004. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 3160, 2004, pp. 240–251.
- [12] BOOCH, G.: Object-Oriented Analysis and Design with Applications. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1994.
- [13] BOOCH, G.—RUMBAUGH, J.—JACOBSON, I.: The Unified Modeling Language for Object-Oriented Development. Unix Review, Vol. 14, 1996, No. 13, <http://www.ccs.neu.edu/research/demeter/course/f96/readings/uml9.pdf>.
- [14] BRAUNHOFER, M.—RICCI, F.: Contextual Information Elicitation in Travel Recommender Systems. In: Inversini, A., Schegg, R. (Eds.): Information and Communication Technologies in Tourism 2016. Springer, Cham, 2016, pp. 579–592.
- [15] BROOKE, J.: SUS – A Quick and Dirty Usability Scale. Usability Evaluation in Industry, Vol. 189, 1996, No. 194, pp. 4–7.
- [16] CHEN, B.—YU, P.—CAO, C.—XU, F.—LU, J.: ConRec: A Software Framework for Context-Aware Recommendation Based on Dynamic and Personalized Context. 2015 IEEE 39th Annual Computer Software and Applications Conference (COMPSAC), 2015, Vol. 2, pp. 816–821.

- [17] CHEN, G.—CHEN, L.: Augmenting Service Recommender Systems by Incorporating Contextual Opinions from User Reviews. *User Modeling and User-Adapted Interaction*, Vol. 25, 2015, No. 3, pp. 295–329, doi: 10.1007/s11257-015-9157-3.
- [18] CHIDAMBER, S. R.—KEMERER, C. F.: A Metrics Suite for Object Oriented Design. *IEEE Transactions on Software Engineering*, Vol. 20, 1994, No. 6, pp. 476–493, doi: 10.1109/32.295895.
- [19] CHOWDHURY, I.—ZULKERNINE, M.: Using Complexity, Coupling, and Cohesion Metrics as Early Indicators of Vulnerabilities. *Journal of Systems Architecture*, Vol. 57, 2011, No. 3, pp. 294–313, doi: 10.1016/j.sysarc.2010.06.003.
- [20] CHURCHILL, E.—LAUBACH, L. L.—MCCONVILLE, J. T.—TEBBETTS, I.: *Anthropometric Source Book. Volume 1: Anthropometry for Designers*. NASA, Yellow Springs, Ohio, 1978.
- [21] CLARKE, S.: Measuring API Usability. *Dr. Dobb's Journal*, Vol. 29, 2004, No. 5, pp. S1–S5.
- [22] CODINA, V.—RICCI, F.—CECCARONI, L.: Distributional Semantic Pre-Filtering in Context-Aware Recommender Systems. *User Modeling and User-Adapted Interaction*, Vol. 26, 2015, No. 1, pp. 1–32, doi: 10.1007/s11257-015-9158-2.
- [23] COLEMAN, D.—ASH, D.—LOWTHER, B.—OMAN, P.: Using Metrics to Evaluate Software System Maintainability. *Computer*, Vol. 27, 1994, No. 8, pp. 44–49, doi: 10.1109/2.303623.
- [24] COLOMBO-MENDOZA, L. O.—VALENCIA-GARCÍA, R.—RODRÍGUEZ-GONZÁLEZ, A.—ALOR-HERNÁNDEZ, G.—SAMPER-ZAPATER, J. J.: *RecomMetz: A Context-Aware Knowledge-Based Mobile Recommender System for Movie Showtimes*. *Expert Systems with Applications*, Vol. 42, 2015, No. 3, pp. 1202–1222, doi: 10.1016/j.eswa.2014.09.016.
- [25] COUNSELL, S.—LIU, X.—ELDH, S.—TONELLI, R.—MARCHESI, M.—CONCAS, G.—MURGIA, A.: Re-Visiting the ‘Maintainability Index’ Metric from an Object-Oriented Perspective. 2015 41st Euromicro Conference on Software Engineering and Advanced Applications, 2015, pp. 84–87.
- [26] DE PESSEMIER, T.—VANHECKE, K.—MARTENS, L.: A Personalized and Context-Aware News Offer for Mobile Devices. In: Monfort, V., Krempels, K. H., Majchrzak, T. A., Turk, Ž. (Eds.): *Web Information Systems and Technologies (WEBIST 2015)*. Springer, Cham, *Lecture Notes in Business Information Processing*, Vol. 246, 2016, pp. 147–168.
- [27] DENG, J. J.—LEUNG, C. H. C.—MILANI, A.—CHEN, L.: Emotional States Associated with Music: Classification, Prediction of Changes and Consideration in Recommendation. *ACM Transactions on Interactive Intelligent Systems*, Vol. 5, 2015, No. 1, Art.No. 4.
- [28] DEY, A. K.—ABOWD, G. D.: Towards a Better Understanding of Context and Context-Awareness. *Computing Systems*, Vol. 40, 1999, pp. 304–307.
- [29] DJOUDI, B.—BOUANAKA, C.—ZEGHIB, N.: A Formal Framework for Context-Aware Systems Specification and Verification. *Journal of Systems and Software*, Vol. 122, 2015, pp. 445–462.

- [30] DOURISH, P.: What We Talk about When We Talk about Context. *Personal and Ubiquitous Computing*, Vol. 8, 2004, No. 1, pp. 19–30, doi: 10.1007/s00779-003-0253-8.
- [31] ELAHI, M.—BRAUNHOFER, M.—RICCI, F.—TKALCIC, M.: Personality-Based Active Learning for Collaborative Filtering Recommender Systems. In: Baldoni, M., Baroglio, C., Boella, G., Micalizio, R. (Eds.): *International Conference of the Italian Association for Artificial Intelligence (AI*IA 2013)*. Springer International Publishing, Cham, *Lecture Notes in Computer Science*, Vol. 8249, 2013, pp. 360–371.
- [32] FAN, X.—HU, Y.—LI, J.—WANG, C.: Context-Aware Ubiquitous Web Services Recommendation Based on User Location Update. *2015 International Conference on Cloud Computing and Big Data (CCBD)*, 2015, pp. 111–118, doi: 10.1109/CCBD.2015.20.
- [33] FANG, Q.—XU, C.—HOSSAIN, M. S.—MUHAMMAD, G.: STCAPLRS: A Spatial-Temporal Context-Aware Personalized. *ACM Transactions on Intelligent Systems and Technology*, Vol. 7, 2016, No. 4, Art.No. 59.
- [34] FENTON, N.—BIEMAN, J.: *Software Metrics: A Rigorous and Practical Approach*. Third Edition. CRC Press, 2014.
- [35] FILÓ, T. G. S.—BIGONHA, M. A. S.—FERREIRA, K. A. M.: A Catalogue of Thresholds for Object-Oriented Software Metrics. *The First International Conference on Advances and Trends in Software Engineering (SOFTENG 2015)*, 2015, pp. 48–55.
- [36] GENERO, M.—PIATTINI, M.—CALERO, C.: A Survey of Metrics for UML Class Diagrams. *Journal of Object Technology*, Vol. 4, 2005, No. 9, pp. 59–92, doi: 10.5381/jot.2005.4.9.a1.
- [37] GERMANAKOS, P.—BELK, M.: *Human-Centred Web Adaptation and Personalization from Theory to Practice*. Springer, *Human-Computer Interaction Series*, 2016.
- [38] HAMED, A. A.—ROOSE, R.—BRANICKI, M.—RUBIN, A.: T-Recs: Time-Aware Twitter-Based Drug Recommender System. *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2012, pp. 1027–1031, doi: 10.1109/ASONAM.2012.178.
- [39] HAWALAH, A.—FASLI, M.: Utilizing Contextual Ontological User Profiles for Personalized Recommendations. *Expert Systems with Applications*, Vol. 41, 2014, No. 10, pp. 4777–4797, doi: 10.1016/j.eswa.2014.01.039.
- [40] HECKMANN, D.: *Ubiquitous User Modeling*. Ph.D. thesis, Saarland University, Germany, 2005.
- [41] HECKMANN, D.—SCHWARZKOPF, E.—MORI, J.—DENGLER, D.—KRÖNER, A.: The User Model and Context Ontology GUMO Revisited for Future Web 2.0 Extensions. *Proceedings of the International Workshop on Contexts and Ontologies: Representation and Reasoning (C & O:RR)*, 2007, pp. 37–46.
- [42] HEITLAGER, I.—KUIPERS, T.—VISSER, J.: A Practical Model for Measuring Maintainability. *6th International Conference on the Quality of Information and Communications Technology (QUATIC 2007)*, 2007, pp. 30–39, doi: 10.1109/QUATIC.2007.8.
- [43] HSIEH, C. K.—YANG, L.—WEI, H.—NAAMAN, M.—ESTRIN, D.: Immersive Recommendation: News and Event Recommendations Using Personal Digital Traces.

- Proceedings of the 25th International Conference on World Wide Web (WWW'16), 2016, pp. 51–62, doi: 10.1145/2872427.2883006.
- [44] HUSSEIN, T.—LINDER, T.—GAULKE, W.—ZIEGLER, J.: Hybreed: A Software Framework for Developing Context-Aware Hybrid Recommender Systems. *User Modeling and User-Adapted Interaction*, Vol. 24, 2014, No. 1-2, pp. 121–174, doi: 10.1007/s11257-012-9134-z.
- [45] INZUNZA, S.—JUÁREZ-RAMÍREZ, R.—JIMÉNEZ, S.: User Modeling Framework for Context-Aware Recommender Systems. In: Rocha, Á., Correia, A. M., Adeli, H., Reis, L. P., Costanzo, S. (Eds.): *Recent Advances in Information Systems and Technologies (WorldCIST 2017)*, Vol. 1. Springer, Cham, *Advances in Intelligent Systems and Computing*, Vol. 569, 2017, pp. 899–908.
- [46] INZUNZA, S.—JUÁREZ-RAMÍREZ, R.—RAMÍREZ-NORIEGA, A.: User and Context Information in Context-Aware Recommender Systems: A Systematic Literature Review. In: Rocha, Á., Correia, M. A., Adeli, H., Reis, P. L., Mendonça Teixeira, M. (Eds.): *New Advances in Information Systems and Technologies*. Springer, Cham, *Advances in Intelligent Systems and Computing*, Vol. 444, 2016, pp. 649–658.
- [47] ISO/IEC TR 9126-2:2003 – Software Engineering – Product Quality – Part 2: External Metrics. International Organization for Standardization, Geneva, Switzerland, 2003.
- [48] ISO 9241-11:1998 – Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs). The International Organization for Standardization, 1998.
- [49] JAWAHEER, G.—WELLER, P.—KOSTKOVA, P.: Modeling User Preferences in Recommender Systems: A Classification Framework for Explicit and Implicit User Feedback. *ACM Transactions on Interactive Intelligent Systems*, Vol. 4, 2014, No. 2, Art. No. 8, doi: 10.1145/2512208.
- [50] JOHN, O. P.—DONAHUE, E.—KENTLE, R.: The Big Five: Factor Taxonomy: Dimensions of Personality in the Natural Language and in Questionnaire. In: Pervin, L. A. (Ed.): *Handbook of Personality: Theory and Research*. Guilford Press, New York, 1990, pp. 66–100.
- [51] KAMINSKAS, M.—RICCI, F.: Contextual Music Information Retrieval and Recommendation: State of the Art and Challenges. *Computer Science Review*, Vol. 6, 2012, No. 2-3, pp. 89–119, doi: 10.1016/j.cosrev.2012.04.002.
- [52] KAKLANIS, N.—BISWAS, P.—MOHAMAD, Y.—GONZALEZ, M. F.—PEISNER, M.—LANGDON, P.—TZOVARAS, D.—JUNG, C.: Towards Standardisation of User Models for Simulation and Adaptation Purposes. *Universal Access in the Information Society*, Vol. 15, 2016, No. 1, pp. 21–48.
- [53] KIM, H. N.—HA, I.—LEE, K. S.—JO, G. S.—EL-SADDIK, A.: Collaborative User Modeling for Enhanced Content Filtering in Recommender Systems. *Decision Support Systems*, Vol. 51, 2011, No. 4, pp. 772–781.
- [54] KITCHENHAM, B.—BRERETON, O. P.—BUDGEN, D.—TURNER, M.—BAILEY, J.—LINKMAN, S.: Systematic Literature Reviews in Software Engineering – A Systematic Literature Review. *Information and Software Technology*, Vol. 51, 2009, No. 1, pp. 7–15, doi: 10.1016/j.infsof.2008.09.009.

- [55] KOŠIR, A.—ODIĆ, A.—KUNAVER, M.—TKALČIČ, M.—TASIČ, J. F.: Database for Contextual Personalization. *Elektrotehniški Vestnik*, Vol. 78, 2011, No. 5, pp. 270–274.
- [56] KUFLIK, T.—KAY, J.—KUMMERFELD, B.: Challenges and Solutions of Ubiquitous User Modeling. In: Krüger, A., Kuflik, T. (Eds.): *Ubiquitous Display Environments*. Springer, Berlin, Heidelberg, Cognitive Technologies, 2012, pp. 7–30.
- [57] LANDIA, N.: Building Recommender Systems for Fashion. *Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17)*, 2017, p. 343, <http://dl.acm.org/citation.cfm?doid=3109859.3109929>.
- [58] LINDLAND, O. I.—SINDRE, G.—SOLVBERG, A.: Understanding Quality in Conceptual Modeling. *IEEE Software*, Vol. 11, 1994, No. 2, pp. 42–49, doi: 10.1109/52.268955.
- [59] LUO, J.—FENG, H.: A Web-Based Framework for Lightweight Context-Aware Mobile Applications. *International Journal of Database Theory and Application*, Vol. 9, 2016, No. 4, pp. 119–134.
- [60] McCABE, T. J.: A Complexity Measure. *IEEE Transactions on Software Engineering*, Vol. SE-2, 1976, No. 4, pp. 308–320, doi: 10.1109/TSE.1976.233837.
- [61] METTOURIS, C.—PAPADOPOULOS, G. A.: Designing Context Models for CARS Incorporating Partially Observable Context. In: Christiansen, H., Stojanovic, I., Papadopoulos, G. (Eds.): *Modeling and Using Context (CONTEXT 2015)*. Springer, Cham, Lecture Notes in Computer Science, Vol. 9405, 2015, pp. 118–131, doi: 10.1007/978-3-319-25591-0_9.
- [62] METTOURIS, C.—PAPADOPOULOS, G. A.: Using Appropriate Context Models for CARS Context Modelling. In: Kunifuji, S., Papadopoulos, A. G., Skulimowski, M. J. A., Janusz, K. (Eds.): *Knowledge, Information and Creativity Support Systems (KICSS '2014)*. Springer International Publishing, Cham, *Advances in Intelligent Systems and Computing*, Vol. 416, 2016, pp. 65–79.
- [63] Microsoft: Code Metrics Values. 2015, <https://msdn.microsoft.com/en-us/library/bb385914.aspx>.
- [64] MOODY, D. L.: Measuring the Quality of Data Models: An Empirical Evaluation of the Use of Quality Metrics in Practice. *Proceedings of the European Conference on Information Systems (ECIS 2003)*, 2003, pp. 1337–1352.
- [65] MOODY, D. L.: Theoretical and Practical Issues in Evaluating the Quality of Conceptual Models: Current State and Future Directions. *Data and Knowledge Engineering*, Vol. 55, 2005, No. 3, pp. 243–276, doi: 10.1016/j.datak.2004.12.005.
- [66] MORRIS, W. N.: More on the Mood-Emotion Distinction. *Psychology*, Vol. 3, 1992, No. 7, Art.No. 6.
- [67] NABOULSI, Z.: Code Metrics – Class Coupling. 2011, <https://blogs.msdn.microsoft.com/zainnab/2011/05/25/code-metrics-class-coupling/>.
- [68] NABOULSI, Z.: Code Metrics – Maintainability Index. 2011, <https://blogs.msdn.microsoft.com/zainnab/2011/05/26/code-metrics-maintainability-index/>.
- [69] NABOULSI, Z.: Code Metrics – Depth of Inheritance (DIT). 2011, <https://blogs.msdn.microsoft.com/zainnab/2011/05/19/code-metrics-depth-of-inheritance-dit/>.

- [70] OKU, K.—NAKAJIMA, S.—MIYAZAKI, J.—UEMURA, S.: Context-Aware SVM for Context-Dependent Information Recommendation. Proceedings of the 7th International Conference on Mobile Data Management (MDM '06), IEEE, 2006, doi: 10.1109/MDM.2006.56.
- [71] OTEBOLAKU, A. M.—ANDRADE, M. T.: Context-Aware Media Recommendations for Smart Devices. *Journal of Ambient Intelligence and Humanized Computing*, Vol. 6, 2015, No. 1, pp. 13–36.
- [72] PEREPLETCHIKOV, M.—RYAN, C.—FRAMPTON, K.: Cohesion Metrics for Predicting Maintainability of Service-Oriented Software. Proceedings of the Seventh International Conference on Quality Software (QSIC 2007), 2007, pp. 328–335, doi: 10.1109/QSIC.2007.4385516.
- [73] PERERA, C.—ZASLAVSKY, A.—CHRISTEN, P.—GEORGAKOPOULOS, D.: Context Aware Computing for the Internet of Things: A Survey. *IEEE Communications Surveys and Tutorials*, Vol. 16, 2014, No. 1, pp. 414–454.
- [74] RICCI, F.—ROKACH, L.—SHAPIRA, B.—KANTOR, P. B.: *Recommender Systems Handbook*. Springer US, Boston, MA, 2011.
- [75] SAMOLADAS, I.—STAMELOS, I.—ANGELIS, L.—OIKONOMOU, A.: Open Source Software Development Should Strive for Even Greater Code Maintainability. *Communications of the ACM*, Vol. 47, 2004, No. 10, pp. 83–87, doi: 10.1145/1022594.1022598.
- [76] SAURO, J.: *Measuring Usability with the System Usability Scale (SUS)*. 2011.
- [77] SCHEDL, M.: Ameliorating Music Recommendation: Integrating Music Content, Music Context, and User Context for Improved Music Retrieval and Recommendation. Proceedings of International Conference on Advances in Mobile Computing and Multimedia (MoMM '13), Vol. 3, 2013, pp. 3–9, doi: 10.1145/2536853.2536856.
- [78] SCHRECK, J.: *Security and Privacy in User Modeling*. Springer Netherlands, Human-Computer Interaction Series, Vol. 2, 2003, doi: 10.1007/978-94-017-0377-2.
- [79] SETH, A.—ZHANG, J.: A Social Network Based Approach to Personalized Recommendation of Participatory Media Content. Proceedings of the 2nd International Conference on Weblogs and Social Media (ICWSM '08), 2008.
- [80] SHANI, G.—GUNAWARDANA, A.: Evaluating Recommendation Systems. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P. (Eds.): *Recommender Systems Handbook*. Springer, Boston, MA, 2011, pp. 257–297.
- [81] SHAPIRA, B.—ROKACH, L.—FREILIKHMAN, S.: Facebook Single and Cross Domain Data for Recommendation Systems. *User Modeling and User-Adapted Interaction*, Vol. 23, 2013, No. 2-3, pp. 211–247, doi: 10.1007/s11257-012-9128-x.
- [82] SHATNAWI, R.—LI, W.—SWAIN, J.—NEWMAN, T.: Finding Software Metrics Threshold Values Using ROC Curves. *Journal of Software Maintenance and Evolution: Research and Practice*, Vol. 22, 2010, No. 1, pp. 1–16, doi: 10.1002/smr.404.
- [83] SIOLAS, G.—CARIDAKIS, G.—MYLONAS, P.—KOLLIAS, S.—STAFYLOPATIS, A.: Context-Aware User Modeling and Semantic Interoperability in Smart Home Environments. 2013 8th International Workshop on Semantic and Social Media Adaptation and Personalization, 2013, pp. 27–32, doi: 10.1109/SMAP.2013.19.
- [84] SKILLEN, K.-L.—CHEN, L.—NUGENT, C. D.—DONNELLY, M. P.—BURNS, W.—SOLHEIM, I.: Ontological User Profile Modeling for Context-Aware Application Per-

- sonalization. In: Bravo, J., López-de-Ipiña, D., Moya, F. (Eds.): *Ubiquitous Computing and Ambient Intelligence (UCAmI 2012)*. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7656, 2012, pp. 261–268.
- [85] SMACCHIA, P.: *NDepend Code Metrics Definitions*. 2016.
- [86] STEFANIDIS, K.—NTOUTSI, E.—PETROPOULOS, M.—NØRVÅG, K.—KRIEGEL, H.-P.: A Framework for Modeling, Computing and Presenting Time-Aware Recommendations. In: Hameurlain, A., Küng, J., Wagner, R., Liddle, S. W., Schewe, K. D., Zhou, X. (Eds.): *Transactions on Large-Scale Data- and Knowledge-Centered Systems X: Special Issue on Database- and Expert-Systems Applications*. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 8220, 2013, pp. 146–172.
- [87] STOREY, V. C.—TRUJILLO, J. C.—LIDDLE, S. W.: Research on Conceptual Modeling: Themes, Topics, and Introduction to the Special Issue. *Data and Knowledge Engineering*, Vol. 98, 2015, pp. 1–7, doi: 10.1016/j.datak.2015.07.002.
- [88] TROELSEN, A.—JAPIKSE, P.: *ADO.NET Part III: Entity Framework*. In: Troelsen, A., Japikse, P.: *C# 6.0 and the .NET 4.6 Framework*. Apress, Berkeley, CA, 2015, pp. 929–999.
- [89] VERBERT, K.—MANOUSELIS, N.—OCHOA, X. et al.: Context-Aware Recommender Systems for Learning: A Survey and Future Challenges. *IEEE Transactions on Learning Technologies*, Vol. 5, 2012, No. 4, pp. 318–335, doi: 10.1109/TLT.2012.11.
- [90] WALTER, T.—PARREIRAS, F. S.—STAAB, S.: An Ontology-Based Framework for Domain-Specific Modeling. *Software and Systems Modeling*, Vol. 13, 2014, No. 1, pp. 83–108.
- [91] WU, D.—CHEN, L.—ZHOU, Y.—XU, B.: A Metrics-Based Comparative Study on Object-Oriented Programming Languages. *Proceedings of 27th International Conference on Software Engineering and Knowledge Engineering*, 2015, pp. 272–277, doi: 10.18293/SEKE2015-064.
- [92] WU, S.—LIU, Q.—WANG, L.—TAN, T.: Contextual Operation for Recommender Systems. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 28, 2016, No. 8, pp. 2000–2012.
- [93] W.W.W. Consortium: *Composite Capabilities/Preference Profiles*. <http://www.w3.org/Mobile/CCPP>, 2004.
- [94] YAMAN, Ç.—ÇIÇEKLI, N.: A Content Recommendation Framework Using Ontological User Profile. In: Gelenbe, E., Lent, R. (Eds.): *Computer and Information Sciences III*. Springer, London, 2013, pp. 381–389.
- [95] YIN, H.—CUI, B.: *Spatial Context-Aware Recommendation. Spatio-Temporal Recommendation in Social Media*. Springer Singapore, SpringerBriefs in Computer Science, 2016, pp. 41–63.
- [96] ZHENG, Y.—MOBASHER, B.—BURKE, R.: Context Recommendation Using Multi-Label Classification. *Proceedings of the 13th IEEE/WIC/ACM International Conference on Web Intelligence (WI 2014)*, Vol. 2, 2014, pp. 288–295, doi: 10.1109/WI-IAT.2014.110.
- [97] ZHENG, Y.—MOBASHER, B.—BURKE, R.: CARSKit: A Java-Based Context-Aware Recommendation Engine. *Proceedings of the 2015 IEEE International*

Conference on Data Mining Workshop (ICDMW '15), 2015, pp. 1668–1671, doi: 10.1109/ICDMW.2015.222.

- [98] ZIMMERMANN, A.—LORENZ, A.—OPPERMANN, R.: An Operational Definition of Context. In: Kokinov, B., Richardson, D. C., Roth-Berghofer, T. R., Vieu, L. (Eds.): Modeling and Using Context (CONTEXT 2007). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 4635, 2007, pp. 558–571.



Sergio INZUNZA received his Ph.D. degree in engineering from the Autonomous University of Baja California i México in 2018 focused on building a user and context model for context-aware recommender system. He received his Master's degree in computer science from the same university in 2014. He is passionate about software engineering and creating tool for developers.



Reyes JUÁREZ-RAMÍREZ has a Bachelor's degree in computer engineering, a Master's degree in computer science and Ph.D. in computer science (2008). Since 2002 he is Professor-Researcher at Universidad Autónoma de Baja California, Campus Tijuana, México. His research areas are software engineering, human-computer interaction, knowledge engineering, and engineering education and training. He is the Technical Coordinator of the Mexican Network of Software Engineering since 2014 and IEEE Senior Member since 2014.



Samantha JIMÉNEZ holds her Ph.D. from the Universidad Autónoma de Baja California. She received her Bachelor's degree in 2011 in computer systems and Master's in engineering in 2013 from University of Colima (Colima, México). Currently she is Lecturer at Universidad Autónoma de Baja California Tijuana, México. Her research interests are in the areas of human computer-interaction, dialogue systems, affective computing, educational systems.



Guillermo LICEA is Full Time Professor at Universidad Autónoma de Baja California. He has Ph.D. in computer science from CICESE Research Center. He is Member of the research group “Software Technology and Interactive Systems” and his research interests include software engineering and the improvement of engineering education using new education strategies and software tools.

A STOCHASTIC ADJUSTMENT STRATEGY FOR COORDINATION PROCESS IN DISTRIBUTED NETWORKS

Pingting HAO, Liang HU
Jingyan JIANG, Xilong CHE

College of Computer Science and Technology

Jilin University

Changchun, 130000, P. R. China

e-mail: {haopt15, jiangjy14}@mails.jlu.edu.cn,

{hul, chexilong}@jlu.edu.cn

Abstract. Cloud computing has become a popular basis that integrated into amount of large platforms to support applications (e.g., multimedia, vehicle traffic, and IoT). It is critical to focus on coordinating the part of these applications that execute in the cloud to provide reliable, scalable and available services. Nevertheless, the problem of optimally coordinating the applications is rarely addressed. In this paper, we develop a stochastic model to analyze the fundamental characteristics that occur in ZooKeeper during the coordination process. The model primarily addresses two aspects: demands of followers and the load of a leader. Then, we derive the optimal strategy for provision with deployment of coordinated servers to achieve load balancing based on various factors (e.g. server capacity and network load), so that the overall network performance is optimized. We evaluate our algorithm under realistic settings and reveal the trend of factors such as CPU, memory utilization and network bandwidth with the increasing number of requests. We propose the algorithm that considers how many servers should be deployed and when. Our results demonstrate that the strategy guarantees the performance by making suitable deployment adjustment.

Keywords: ZooKeeper, deployment, load balancing, queuing theory

Mathematics Subject Classification 2010: 49-K30, 60-G07

1 INTRODUCTION

The popularity envisioned for the fifth generation (5G) mobile network has transformed cloud services into the basic infrastructure for large-scale distribution systems. To date, not only traditional applications, but also numerous other architectures (e.g. SDN [1] and CDN [2]) have been integrated into the cloud. This rapid growth has posed significant challenges to the cloud, e.g. how to provide distributed synchronized services while maintaining configuration information to support more scalable, available and reliable services for users.

To address these challenges, ZooKeeper [3] which includes built-in, large-scale, coordinated mechanisms has been widely adopted in the cloud. The shared hierarchical namespace and data model are used for coordinating the transactions. Generally, ZooKeeper runs on several closely located servers, namely, a ZooKeeper cluster. Different roles are assigned to the servers in order to communicate with each other, to provide the functions of fault-tolerance and to deal with the information about the transactions according to the Paxos algorithm. Thus, ZooKeeper could provide a set of guarantees (e.g. sequential consistency, reliability and atomicity). ZooKeeper is typically deployed and operated by third-party architectures (e.g. HBase [4] and Storm [5]), and it is also widely used in the applications [6, 7, 8]. Such mechanisms have contributed to ZooKeeper's increasing popularity, particularly because of its distributed capabilities. Thus, achieving an optimal performance and better utilization of ZooKeeper is important also for these applications. In some cases from the companies (e.g. Baidu, Alibaba), the poor performance would not be solved only by expanding the size of cluster according to the SLA, and the results are caused by improper operations, such as the way of storage, the assignment of roles and the deployment of ZooKeeper. Thus, we pick the problem for deployment of ZooKeeper, which is one of the improper mistakes, to analyze it and find the solution.

Okorafor and Patrick [9] proposed the model based on the Hadoop/MapReduce and ZooKeeper, and utilized the ZooKeeper to coordinate the transactions instead of traditional MPI (message passing interface). Although it analyzed the number of servers for ZooKeeper, the primary goal was designed for the availability to achieve better performance (e.g. the repair rates for software and hardware, the initial failure rate, and the number of available servers). Specific to the characteristic and analysis of the ZooKeeper itself, it has no more deep research on this field. Pham et al. [13] evaluated ZooKeeper's performance under different situations (e.g. resilient name service, dynamic system configuration, and recovery databases). Complementary to [13], we combine the servers' states with ZooKeeper's performance. Furthermore, we provide an in-depth study of ZooKeeper, specific to its deployment.

In this paper, we propose the deployment for ZooKeeper according to the coordinating process. In view of the analytical results, the plan comprehensively considers the trade-off between the leader and followers with the load balancing, so that to guarantee the performance of the system. We can summarize our contributions as follows:

1. We give a brief description of the ZooKeeper workflow and develop a stochastic model to describe the coordination process using two modes: a simple mode and a general mode. Meanwhile, our stochastic model considers two conflicting roles to determine the server deployment by introducing load balancing as the metric, which provides a general baseline for understanding ZooKeeper deployment.
2. We observe the dynamic server states and their performance with dealing with requests, and derive a strategy for provisioning the optimal number of servers and determining when to adjust the deployment by considering the server states and the request distribution in the system.
3. Through a numerical analysis, we observe the performance variations: the bandwidth becomes more sensitive as the number of requests increases, and the CPU load also varies significantly. At a threshold, the throughput influence holds steady for both the leader and followers. When the service utility achieves optimality from a global viewpoint, we change the deployment to guarantee the quality of service.

The rest of this paper is organized as follows. In Section 2 we present related work. In Section 3, we describe the workflow of ZooKeeper which combines with our challenge, and provide the definition of our model. In Section 4, we concentrate on building the model for the coordination process and solving the problems mentioned in Section 3, and the results of experiments are discussed in Section 5. Finally, we conclude the paper in Section 6.

2 RELATED WORK

Recently, in the emerging coordinated services field, ZooKeeper's management consistency exhibits promising potential that can significantly improve system efficiency. Cai et al. [6] proposed a decentralized vehicle routing service system and used ZooKeeper to establish the coordination system between subtask processors. Goel and Majumdar [7] presented mutual exclusion property in ZooKeeper to guarantee payment processing. Skeirik et al. [8] focused on security services and group key management in the cloud and used ZooKeeper's logic model to bolster anti-spam and anti-virus activities. In [10], the importance of the organization and management for the service is addressed. However, the ZooKeeper is mainly used in the cloud distributed systems to support the valuable search related services.

Researchers have expended considerable effort to improve the ZooKeeper's performance. Kalantari and Schiper [11] developed a prototype to reduce the synchronization time between ZooKeeper servers. Junqueira et al. [12] designed a crash-recovery atomic broadcast algorithm to guarantee state coordination. In 5G architecture, both homogenous and heterogenous units have been integrated into the infrastructure. While the applications are likely to possess full support for developing intelligent platforms, it is challenging to utilize ZooKeeper to guarantee the services. However, performance analysis of ZooKeeper are rarely seen in the literature.

In this section, we discuss deployment, which is closely related to our work. Deployment has been a key architectural component for many years, and the literature includes many studies on deployment. The deployment objective is to identify the most appropriate number of servers, replicas and locations to achieve various optimal solutions, such as minimizing the delay [14, 15, 21, 22], bandwidth [16], energy consumed [17, 18, 19, 24], and so on. Moreover, LP relaxation techniques are widely used as a practical method to approximate the optimal solution [18]. Heuristic algorithms [16, 20] are often employed in these studies.

Our work concentrates on adjusting server deployment to guarantee the quality of service at both low cost and energy consumption. Due to the properties (e.g. intensive cluster, small sizes of items) of ZooKeeper, our approach is novel for ZooKeeper, and is more critical than the problems of location and size as the scale of components in the architecture increases. We introduce load balancing into our model. As a distributed service system, load balancing is a non-negligible factor in achieving system optimal performance [23]. Kim [24] improved a novel load balancing scheme that balances the energy consumption of the sensor nodes and the maximum network lifetime by applying sub-network management in wireless sensor networks. Bui et al. [25] presented approaches to improve networking performance by rebalancing the load on the physical links of a supercomputer. Thus, our model combines ZooKeeper with the system seamlessly, and providing insights for addressing ZooKeeper deployment.

3 SCENARIOS AND PROBLEM STATEMENT

3.1 ZooKeeper and Various Scenarios

The architecture for a generic example, the coordination process, is shown in Figure 1. Each server in the ZooKeeper cluster is configured with the service. The system involves three roles: *leader*, *follower* and *observer*. Due to the semi-distributed manner of ZooKeeper, the coordination service needs a role for the collection, computing and issuing decisions, that is the responsibility of *leader*. The number of leader is limited to one. The other two roles *follower* and *observer* mainly connect with the user, and receive the requests from one part of the area. The *follower* is responsible for satisfying the needs of users within the range, as the *observer*. The number of them is not limited.

The whole servers of *follower* and *observer* could cover the range of needs, and the number of them is not the same with *leader* that is limited to one. The part of the requests passed on to the *leader*, the other part would be dealt with by the *follower* and *observer*. The *observer* functions similarly like the followers. The difference between observers and followers is that the *observers* provide only reading services, but do not vote. Therefore, in this paper, because of their popularity and universality, we primarily discuss the first two roles.

Figure 1 shows the coordination process through the six steps. The client makes a request (step 1) to a follower. Then, the follower takes different actions depending

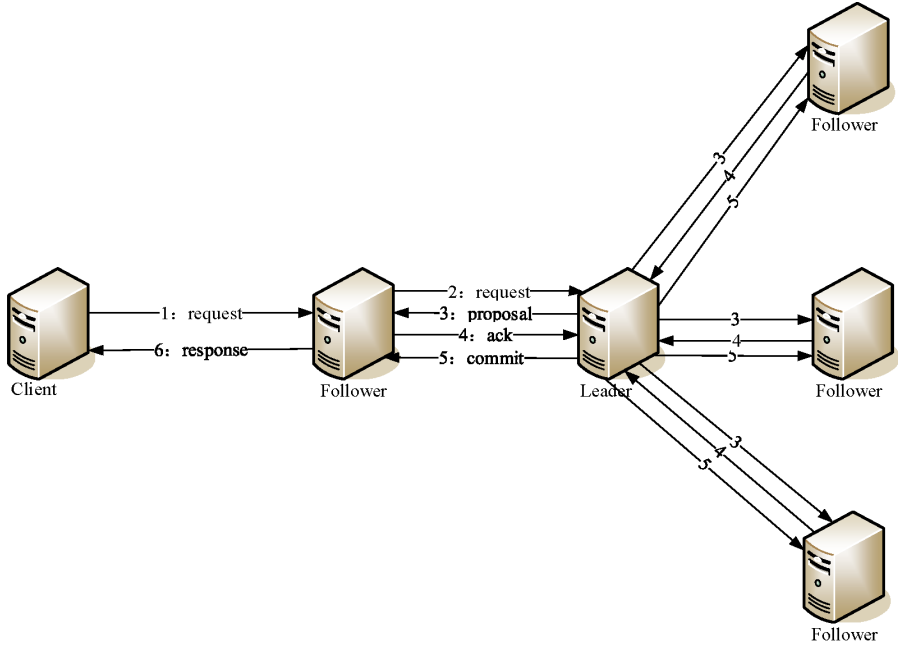


Figure 1. The coordination process (six steps) between a leader and its followers using the TCP/IP protocol

on the request type. The requests can be categorized as *read* or *write*. When the request type is *read*, the follower serves the request itself by performing step 6. When the type of the request is *write*, it is forwarded to the leader (step 2). The packet categories resulting from the write request consist of PING, REQUEST, ACK and REVALIDATE packets. The results will be returned by the followers from the leader (step 5). When the leader receives the request, it proposes a vote among its active followers (step 3). After the vote is returned by each follower (step 4), the leader calculates the result and notifies all the followers (step 5). The packet types implemented by each follower including PROPOSAL, COMMIT, UPDATE and REVALIDATE, and the request procedure obeys the coordination process. The protocol Zab, derived from the Paxos algorithms, is the basis of the coordination process that makes the system reliable, consistent and available. For example, if the leader crashes, the remaining followers elect a new leader based on the Zab protocol. In addition, when the leader performs update operations, it propagates the incremental state changes to its followers so that the consistency is guaranteed. Note that there is an odd number of servers ($2n + 1, n = 1, 2, 3, \dots$), and the number of active servers is never less than $n + 1$, which avoids evenly splitting votes.

3.2 Problem Definition and Notation

In this section, we describe the coordination model and formulate the optimization problem in ZooKeeper. Table 1 summarizes the notations used in this paper.

First, the client connects to one server to create a client/server session. Then, the client can submit requests to the server. Next, the server receives the client's requests and judges the type of requests. Based on the request type, it determines whether the request must be transferred to the leader. *Search* and *read* operations are not forwarded, while *setData*, *create* and *delete* requests among others are forwarded to the leader. For the latter, the leader organizes voting among its active followers and returns the results to all followers as a *commit* operation. Clients receive their results from the followers. Thus, the challenge is rising from this procedure. The amount of requests are received by the follower, and the more requests need the followers with more capacity to deal with such as increasing the number of followers, so that the burden of followers could be afforded and guarantee the speed of dealing with the requests. However, the effect of the leader is opposite from the followers due to the semi-distributed manner of ZooKeeper. The more requests and actions would result in the more burden of the leader, and the performance would be influenced by the leader. If the leader is also responsible for the work of the followers, it would be worse for the performance. Generally, the leader is not set to receive the users' requests as default. Thus, the scaling point of the cluster could be considered from the two angles that is dealing with the requests by the followers and the leader.

The network topology is represented as a graph $G = (V, E)$. V contains n nodes connected by the edges from set E . Based on the definition of a Poisson process, we assume that the arrival rate of clients λ follows a Poisson distribution, which has been discussed in many papers. For the departure rate of the clients μ , the instances can be separated into two different situations. The first case is simple, and client requests are served at rate ν under ideal conditions. In the second case, we consider a realistic scenario in which failure is introduced. We denote the failure rate of the leader as f_{sL} and the failures caused by clients is denoted as γ , where γ is a part of μ . At the physical level, many incidents can cause the system to become unavailable, such as power outages, wire disconnections, routing inaccuracies, link errors, and so on. Except for special and infrequent situations, we focus on the link error rate f_l .

Next, we set the network parameters. M defines the total number of servers, including the leader and the followers, while m denotes the number of followers. Considering the limited real-world conditions, we assume that client bandwidth is all the same, and the upper bound of the bandwidth is c_1 . The followers have a maximum bandwidth of c_2 , while the upper bound of the leader bandwidth is c_3 . In this paper, we also add the servers' attributes to this model. Combined with the theory of load balancing, FW stands for the load capacity of the follower, and LW denotes the leader's load capacity. Load capacity is reflected by the CPU idle time V_1 , the surplus storage space V_2 , and the remaining network bandwidth V_3 . However, the two types of requests, *read* and *write*, have different effects on server

load capacity. Thus, we denote that the number of requests for two types as rw_1 and rw_2 to show their respective influences on ZooKeeper.

NOTATION	DEFINITION
λ_{si}	Client arrival rate to server i , $i = 1, 2, 3, \dots$, represents the followers, and $i = L$ is the leader
ν	Client departure rate under ideal conditions
μ	Actual client departure rate
ρ	The intensity of service, $\rho = \lambda/\mu$
γ	The portion of the departure rate due to a client fault
f_{sL}	The leader sL failure rate
f_l	The link l failure rate
W	The average client waiting time (response time)
N_{si}	The number of operations that pass on server i , where $i = L$ represents the leader
rw_i	The number of requests, $i = 1$ denotes the type of requests <i>read</i> , $i = 2$ denotes the type of requests <i>write</i>
P_n	$P_n = P\{N = n\}$ ($n = 0, 1, 2, \dots$) denotes the event probability when the number of clients n is in a stable state.
$D(x)$	Throughput is counted as the number of packets that pass through a server in the interval x
S_x	The rate of transmission in the interval x
LW_{sL}	The performance of server L , referring to the leader using factors v_i to describe them
FW_{si}	The performance of server i , referring to the follower using factors v_i to describe them
c_i	The bandwidth constraint. The assigned bandwidth of the clients is expressed as $i = 1$, $i = 2$ denotes the maximum follower bandwidth, and $i = 3$ denotes the maximum leader bandwidth
v_i^{si}	The measurement of server i 's performance concerning factor i

Table 1. The major symbols used in this paper

4 COORDINATION MODEL

In this section, we develop the strategy to adjust ZooKeeper deployment and provide two models for analyzing the coordination process.

4.1 Simple Model

We first consider a simple model for ZooKeeper without considering the unexpected accidents that occur in the real conditions. Based on the overall situation, the coordination model is assumed to be M/M/1. In addition, FCFS (first-come-first-serve) is the queuing discipline model in ZooKeeper.

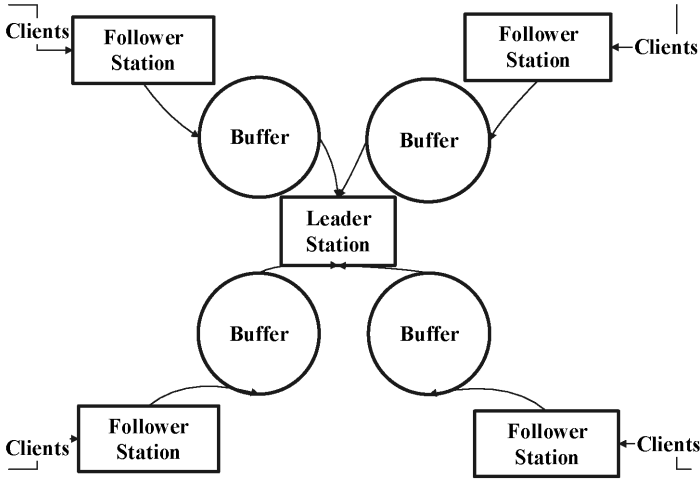


Figure 2. The simple architecture used for the queuing theory

As shown in Figure 2, the leader is the central pivot in the system and is also the organizer when refereeing a proposal. Thus, the buffer of each follower is filled with requests that the leader must address. According to queuing theory, the arrival rate for the followers i is λ_{si} ($i = 1, 2, 3, \dots, m$). Requests from follower i follow the probability σ_{si} ($i = 1, 2, 3, \dots, m$) to the leader, and their arrival rate at the leader λ_{sL} is defined in Equation (1):

$$\lambda_{sL} = \frac{rw_2}{rw_1 + rw_2} \times \sum_{i=1}^m \sigma_{si} \lambda_{si}. \tag{1}$$

According to Little’s law [26], we consider the average waiting time for one follower in Equation (2):

$$W = \frac{1}{\nu - \lambda}. \tag{2}$$

We assume that a client submits a request to the server and does not submit another request until the result is returned to the client. Therefore, the number of operations N_{sL} , as shown in Equation (3), means that the operation has passed through the leader. The number of operations assigned to the followers is denoted by N_{si} in Equation (4). We also assume that each request occupies identical bandwidth and that bandwidth capacity is limited, which means that the total bandwidth used between the clients and followers during a given interval should be less than the followers could provide in practice. We apply the constraint shown in Equation (5):

$$N_{sL} = \lambda_{sL} \times W, \quad (3)$$

$$N_{si} = \lambda_{si} \times W, \quad (4)$$

$$c_1 \times N_{si} \leq c_2. \quad (5)$$

4.2 General Model

In practice, failure cases are considered in the general model. This model supplements the simple model and is closer to reality. There are three types of failure: physical failure, leader-caused failure and client-caused failure. The physical failure rate f_l denotes the link error rate. The leader's failure rate f_{sL} results in reelection. When the leader is out of control, the followers first terminate their connections with the clients. Then, the followers vote for the role of a new *leader*. Each follower has at least one connection with the others. The votes are marked with *zxid* to represent the transaction state, and the votes also have the *server id* that reflects the source. When a server gets a majority of votes (more than half), that server is chosen to be the new leader and reestablish the coordination process in ZooKeeper. The system may experience less throughput during the voting process. In addition, we account for the client's failure rate γ , which reduces the departure rate μ . The follower may cause the failure, but it will not influence the performance. Information about a client's session is stored by both the followers and the leader. When a follower failure occurs, the connected client receives a message from the follower or the TTL times out. Then, the client chooses a new active follower from the set of followers. Recovery from this type of failure is both simple and fast. Therefore, we ignore it in the remainder of this paper. Due to the coordination process, failure will be detected whatever it occurs among the *leader*, its *followers*, and their *clients*. Assuming that the starting time is $x = 0$, we define the number of operations assigned to followers in Equation (6):

$$N_{si} = \lambda \times W - \int_{x=0}^{x=t} \gamma dx - \int_{x=0}^{x=t} f_l dx - f_{sL} \times \lambda_{si} \times W. \quad (6)$$

The two types of requests, *read* and *write*, reflect the different coordination process shown in Figure 1. Therefore, we must compute the number of packets for the two request types differently. The number of packets $D(x)$ ($x = t_1, t_2, t_3, \dots, t_n$) in an interval is defined in Equation (7). Using the number of packets $D(x)$, the rate of transmission S_x can be measured indirectly, as shown in Equation (8):

$$D(x) = \frac{rw_2}{rw_1 + rw_2} \times N_{si} \times [6 + 3 \times (M - 2)] + \frac{rw_1}{rw_1 + rw_2} \times N_{si} \times 2, \quad (7)$$

$$S_x = \frac{dD(x)}{dt}. \quad (8)$$

Then, we introduce the Markov birth-death process to describe the dynamic client changes. Assumed that $\lambda_{n-1} = \lambda_{n-2} = \dots = \lambda_0$, and $\mu_n = \mu_{n-1} = \dots = \mu_1$. Then, the probability of an empty queue can be derived under the constraint of $\sum P = 1$ as shown in Equation (9):

$$P_0 = \frac{1}{1 + \sum_{n=1}^{\infty} \rho^n}. \tag{9}$$

The variable ns denotes the probability of success for completing the coordination process. Moreover, the bandwidth limit c_{max} should not be negligible. Furthermore, if the type of request is *read*, then the M/M/1 queuing system transforms into M/M/1/ c_{max} during the read. However, the *write* type rarely reaches the upper bound of the bandwidth because of its order in the leader. Thus, we assume that *write* type requests still use the M/M/1 queuing system. Then, we have the following:

$$ns = \frac{rw_2}{rw_1 + rw_2} \times (1 - f_i) \times (1 - f_{sL}) \times (1 - P_0^{rw_1}) + \frac{rw_1}{rw_1 + rw_2} \times (1 - f_i) \times (1 - f_{sL}) \times (1 - P_0^{rw_2}). \tag{10}$$

In order to simplify the probability of an empty queue, we transform the P_0 as (10) into (11) using the queuing theory of M/M/1.

$$P_0^{rw_1} = \sum_{n=1}^{\infty} P_n = 1 - P_0 = 1 - \rho. \tag{11}$$

Then, we separate the formula (10) into two parts according to the different types of requests. By using the simplified formulation (11), we conclude the probability of success for completing the type of request *write* in Equation (12):

$$ns_1 = \frac{rw_2}{rw_1 + rw_2} \times (1 - f_i) \times (1 - f_{sL}) \times \rho. \tag{12}$$

The probability of an empty queue for the type of request *read* obey with the theory of M/M/1/ c_{max} , and the probability in (10) is simplified into (13). Thus, the probability of success for completing the type of request *read* is computed as

$$P_0^{rw_2} = \rho^i \times \frac{1 - \rho}{1 - \rho^{c_{max}+1}} (i = 0, 1, 2, \dots, c_{max}), \tag{13}$$

$$ns_2 = \frac{rw_1}{rw_1 + rw_2} \times (1 - f_i) \times (1 - f_{sL}) \times \frac{1 - \rho}{1 - \rho^{c_{max}+1}}. \tag{14}$$

Next, we consider how to achieve the load balancing in the system. More specifically, the follower is mainly responsible for dealing with the requests from the clients. Due to the increasing number of requests, we assume the leader is mainly responsible

for the decision part without receiving the requests directly from the client. Thus, we define the capacity of followers and leader, respectively. We choose the metrics to give a comprehensive understanding of the capacity of the followers in (15), which including the CPU idleness V_1 , the surplus storage space V_2 , and the surplus bandwidth of access for the server V_3 . Then, we set the weights k_1 , k_2 , and k_3 for the three metrics. Specific to ZooKeeper, the capacity for dealing with the requests would mainly influence the CPU, the storage for requests would mainly influence the memory metric, and the bandwidth would be used for communication and delivery. Thus, we take the three important metrics and set the weight of them almost the same. The constraint of the bandwidth is shown in Equation (16), which defines the minimal surplus bandwidth.

$$FW_{si} = k_1 \times V_1^{si} + k_2 \times V_2^{si} + k_3 \times V_3^{si}$$

$$(k_1 + k_2 + k_3 = 1, V_1^{si} \in (0, 1), V_2^{si} \in (0, 1), V_3^{si} \in (0, 1)), \quad (15)$$

$$\min(V_3^{si}) = \frac{c_2 - N_{si} \times c_1}{c_2}. \quad (16)$$

Load balancing is the global theory for evaluating the system. For the assignment of workload, the regulation is to distribute jobs to the servers according to the capability. If the server has a higher capability, it could do more work than the servers which have lower capability. This arrangement can achieve the balance for the system, and such that it could improve the utilization of the resource. Thus, we give the formula (17) as follows:

$$\frac{N_{si}}{FW_{si}} \approx \frac{N_{sj}}{FW_{sj}}. \quad (17)$$

The system maintains the state shown in Equation (17) to achieve the best use of the resources for each server. We observe the indicator b_1 and show the relative load of follower i in (18). When b_1 exceeds its upper limit, *bound1*, the current number of servers in the ZooKeeper cluster is insufficient to service the requests. From a follower viewpoint, the cluster needs to scale as the number of requests increases. Otherwise, the performance will degrade:

$$b_1 = \frac{N_s}{N_{si} \times FW_{si}} \geq bound1. \quad (18)$$

The definition for the leader capacity is similar to that of the followers. We apply the constraints shown below in Equations (19) and (20):

$$LW_{sL} = k_4 \times V_1^{sL} + k_5 \times V_2^{sL} + k_6 \times V_3^{sL}$$

$$(k_4 + k_5 + k_6 = 1, V_1^{sL} \in (0, 1), V_2^{sL} \in (0, 1), V_3^{sL} \in (0, 1)), \quad (19)$$

$$\min(V_3^{sL}) = \frac{c_3 - m \times c_2}{c_3}. \quad (20)$$

By checking the relative load of the leader b_2 , as shown in Equation (21), we judge the timing for adjusting the deployment with b_1 . For example, if we add some quantity of followers, the leader load will increase as the number of requests increases. A heavy leader load results indirectly in slower responses. When the load exceeds the leader’s upper limit $bound2$, the number of followers in the system should be correspondingly reduced.

$$b_2 = \frac{D(x)}{LW_{sL} \times \bar{b}_1} \geq bound2 \tag{21}$$

Through the formulations in Equations (18) and (21), whether to scale the system and when to adjust the followers are the problems that we must address. We synthesize two aspects to consider this problem. After trading off the two aspects, we obtain the formulation in Equation (22):

$$b_3 = k_7 \times b_1 + k_8 \times b_2$$

$$(k_7 + k_8 = 1, b_1 \in (0, 1), b_2 \in (0, 1)) \tag{22}$$

where the threshold is represented by b_3 . The case in which $bound1$ and $bound2$ are both achieved is out of our scope because that condition denotes that the leader capacity needs to be improved to guarantee the performance. Instead, we focus on the utility when sufficient resources are available. When b_3 is achieved, the number of followers should be adjusted, either maintained, increased or decreased. The adjustment of followers could be simply described as follows:

$$b_3 = \begin{cases} \text{increase the number of followers,} & \text{beyond } bound1 \ \&\& \text{equals } b_3 \\ \text{keep status,} & \\ \text{decrease the number of followers,} & \text{beyond } bound2 \ \&\& \text{equals } b_3 \end{cases} \tag{23}$$

5 NUMERICAL RESULTS

In this section, we use a realistic platform to demonstrate follower reassignment as the basis for load balancing. We observe the three metrics of load balancing and the throughput in two related deployments. Then, we compute the relative loads for the followers and the leader to explore the threshold b_3 . Complementary to the theoretical analysis, the experimental results both allow us to gain a better understanding of the reassignment operation and provide guidance to determine approaches which will improve the system’s performance and utility.

5.1 Experimental Setup

Under the ZooKeeper platform, we evaluate our model using a random local area network topology, which means that we pick the servers randomly from the set of local network. Every server runs on Intel Core i3 and has 2 GB of RAM. We assume

that the number of servers is s and configure three different deployments (' $s = 3$ ', ' $s = 5$ ', and ' $s = 7$ ') for the experiment. Then, we could make the adjustment from two groups, one group is between ' $s = 3$ ' and ' $s = 5$ ', the other is between ' $s = 5$ ' and ' $s = 7$ '. We execute each experiment 50 times to obtain average values, and the experiments mainly involve the actions both the leader and followers take part in. The arrival rate increases by a factor of 10 until it reaches the upper limit of the servers' capacity. Although the load of each server reflects the latency, we still set an acceptable latency range of 100 ms. Additionally, in this paper, we define the latency that starts when the clients submit a request and ends when the clients obtain the results from the followers. The ramp-up period is set at 1s, during which each factor is monitored until it reaches a stable state. We intentionally set the weights $k_1:k_2:k_3 = 4:3:3$ for evaluating the followers' loads, and to represent the similar importance for the three metrics, as well as that of the leader. The upper limit of the bandwidth c_2 for the followers is fixed, and the upper limit for the leader c_3 is also fixed. The metrics for the failure part (e.g. the client's failure rate γ , the link error f_l , the leader's failure rate f_{sL}) are set as the realistic pattern rather than the assumed value. Thus, the throughput S_1 and the number of packets $D(x)$ could be derived. Meanwhile, the related information of CPU, memory and bandwidth would be required through monitoring with the input metrics, and the adjustment is performed after computing the bound.

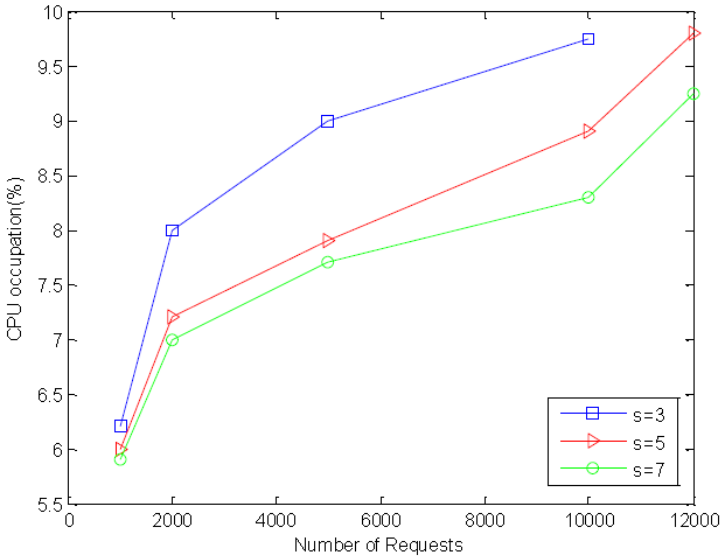
5.2 Evaluation Results

5.2.1 The Analysis for Followers

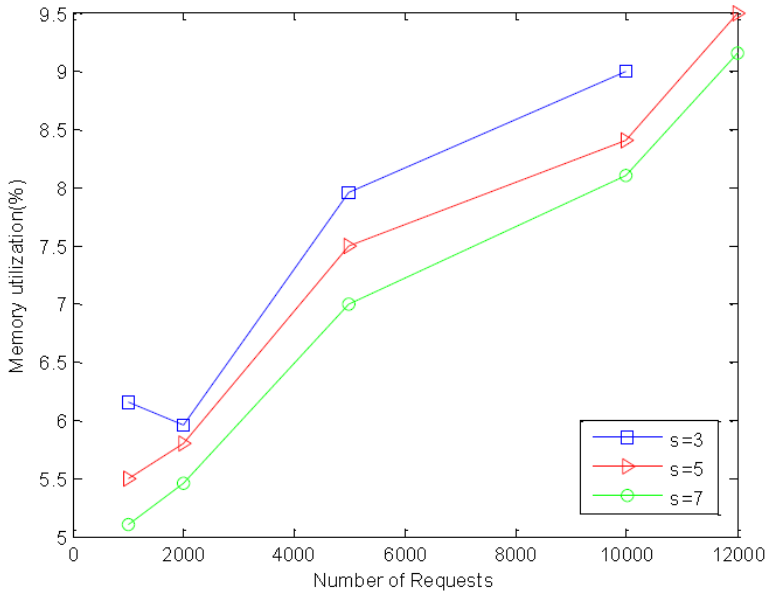
We first demonstrate the trend of three metrics to evaluate the server loads as the number of requests from each follower increases: CPU utilization (%), memory utilization (%), and network bandwidth (%). In addition, the throughput measures the number of requests in a millisecond.

The definition of x axis means that the number of requests would be submitted to the system. The results in y axis are collected from each follower and computed averagely. The results, presented in Figure 3 a), show that the trend of follower's CPU utilization as the number of requests increases. A similar trend occurs not only in the deployment as fewer servers but also more servers. The CPU utilization trend is related to the follower workloads. Thus, the CPU utilization increases with the heavier load of followers accordingly. However, the two deployments are different that more followers cause the CPU utilization of each follower to decrease with the same number of requests. When the number of requests reaches a certain point, the followers nearly achieve maximal capacity. The utility of the CPU in the system approximately achieves its extreme. This is reason for scaling the system to serve more requests.

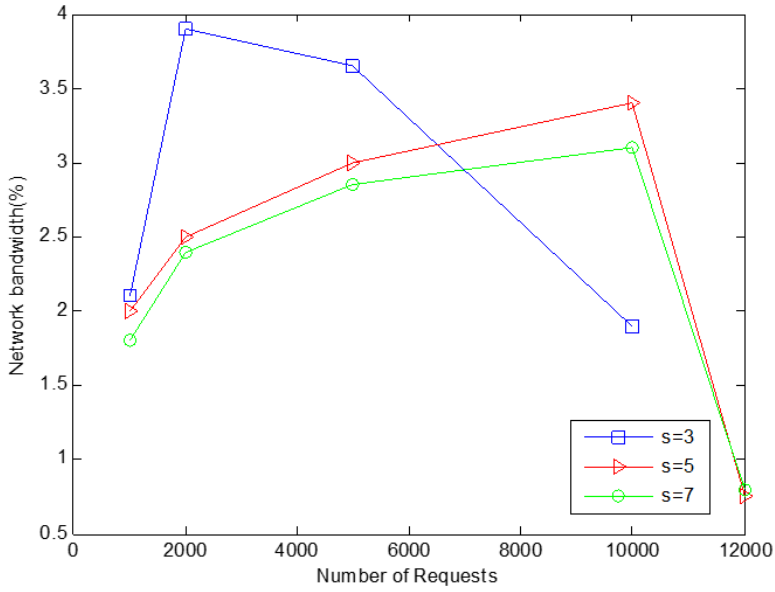
In Figure 3 b), the available memory is sufficient for the experiment. The trend of memory utilization also changes with the number of requests. The more followers there are, the total required storage space could be divided more. Moreover, the



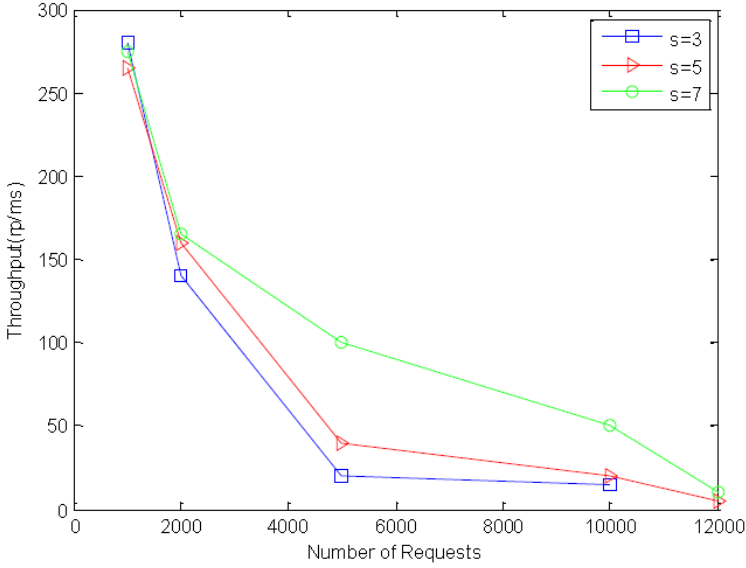
a) The trend of CPU_utilization for the followers



b) The trend of Memory_utilization for the followers



c) The trend of Network_bandwidth for the followers



d) The trend of Throughput for the followers

Figure 3. The capacity of the follower with the increasing number of requests

trend when ' $s = 3$ ' changes more obviously due to the more requests of each follower. However, the dynamic states of network bandwidth are not only related to the number of requests, but also influenced by the network condition (e.g. congestion). The finite bandwidth capacity is one of the main reasons that caused the congestion. Another is the queuing length of each follower. Thus, as shown in Figure 3 c), when the number of requests is within a certain range, the utilization of network bandwidth expands as the number of requests increases, and the same trend occurs in network I/O. However, when the number of requests exceeds the bandwidth capacity, a decreasing trend appears due to the capacity limit. Moreover, as this trend approaches zero, the system breaks down. The peak point for the deployment ' $s = 3$ ' comes up earlier than the other two deployment due to the more number of requests from followers, and it also means that the limit also appears earlier. The deployment ' $s = 5$ ' and ' $s = 7$ ' has the similar tendency in this figure. However, the average number of requests for ' $s = 7$ ' is less than the deployment ' $s = 5$ ', and the limit point is almost the same mainly due to the conflicting requests.

With the increasing number of requests, the trend of the throughput is shown in Figure 3 d). The figure shows that the throughput decreases when the number of requests is 1000, 2000 and 5000. After that, the trend gradually stabilizes. This result is indirectly related to the resource utilization. When the received load is well below the capacity, the throughput is high. As the load of followers increases, the throughput decreases sharply. Firstly, the deployment ' $s = 3$ ' has not been achieved the turning point, the throughput keeps higher because of the largest number of requests. Then, the throughput would decrease with some reasons (e.g. the capacity of servers, congestion, and the conflicting requests). Thus, the deployment ' $s = 7$ ' has the highest throughput after the first point of the deployment ' $s = 3$ ' until the limit point. When the number of requests achieves the limit point, the deployment ' $s = 5$ ' and ' $s = 7$ ' would converge to similar throughput.

In summary, the first time each experiment executes, there are small deviations due to the adaptive phase. The four graphs with dotted points are superimposed to simply analyze the capacities of followers and the system's performance under the three deployments. We can make the following conclusions from the findings above. First, CPU utilization is steady in each experiment. Next, memory utilization fluctuates within 20% of full utilization. Bandwidth utilization is lower than memory utilization, which means that storage is more likely to be a bottleneck than the bandwidth in ZooKeeper.

In the ' $s = 3$ ' deployment, the system can serve approximately ten thousand requests, but it appears to reach a non-working state when the number of requests exceeds ten thousand. This result indicates that a bound exists in ZooKeeper when served requests beyond the extreme point, and the adjustment of deployment should be considered. The deployment of ' $s = 5$ ' has more servers to deal with the requests. Therefore, the limit expands to 12000 requests the same as the deployment ' $s = 7$ '. Although the deployment ' $s = 5$ ' and ' $s = 7$ ' has the similar extreme point, the throughput for the deployment ' $s = 7$ ' has the advantage than the deployment ' $s = 5$ ', as shown in Figure 3 d). When the capacity of the follower in

the system is exceeded, an adjustment must be considered to guarantee the performance.

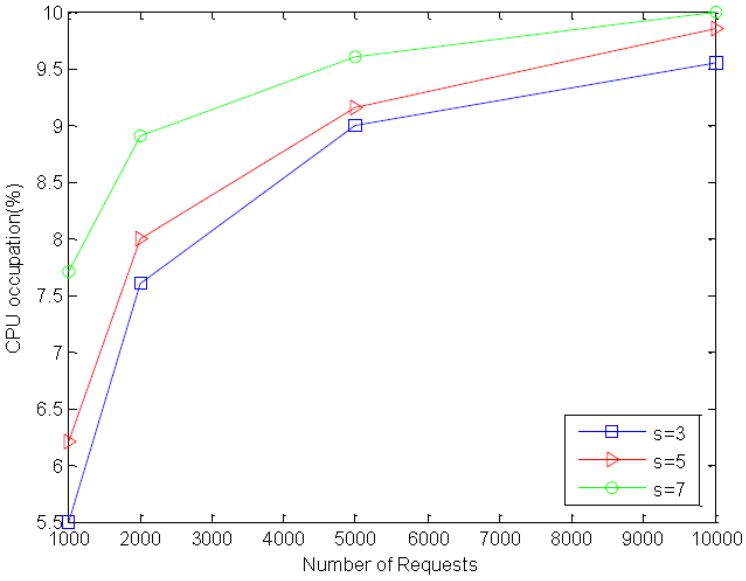
5.2.2 The Analysis for the Leader

The results presented in Figure 4 show the relative load of the leader using the three metrics. The x axis represents for the number of requests that the leader received. With regard to CPU utilization, the trend for the leader is almost similar. The deviations between the three deployments in Figure 4a) are caused by the overhead, which produces through the different numbers of requests. With the number of followers increasing, the utilization of CPU is higher gradually. Thus, the deployment ‘ $s = 7$ ’ has the highest utilization than the other two deployment. For the memory utilization in Figure 4b), the ‘ $s = 7$ ’ deployment needs more space than the ‘ $s = 5$ ’ deployment, and the ‘ $s = 5$ ’ deployment needs more space than the ‘ $s = 3$ ’ deployment. The situation is directly opposite to the trend of followers. From the leader’s point of view, the reason for this disparity is that the number of followers results in the different total number of requests. Thus, this situation means that the leader must maintain more information to manage and control the system. With the requests swarming in, congestion becomes severe and leads to a lower bandwidth utilization as shown in Figure 4c). Thus, the deployment ‘ $s = 7$ ’ has the lower utilization. When the number of requests from each follower achieves 5 000, the two deployments ‘ $s = 5$ ’ and ‘ $s = 7$ ’ would reach the limit, and the derivation between the two deployments is around 0.5% utilization. When the number of requests from each follower achieves 10 000, the bandwidth for deployment ‘ $s = 5$ ’ and ‘ $s = 7$ ’ have slight increasing due to the higher number of requests. The deployment ‘ $s = 5$ ’ has risen around 0.04%, and the deployment ‘ $s = 7$ ’ has risen around 0.02%. Thus, the swarming does not influence much to the limit situation.

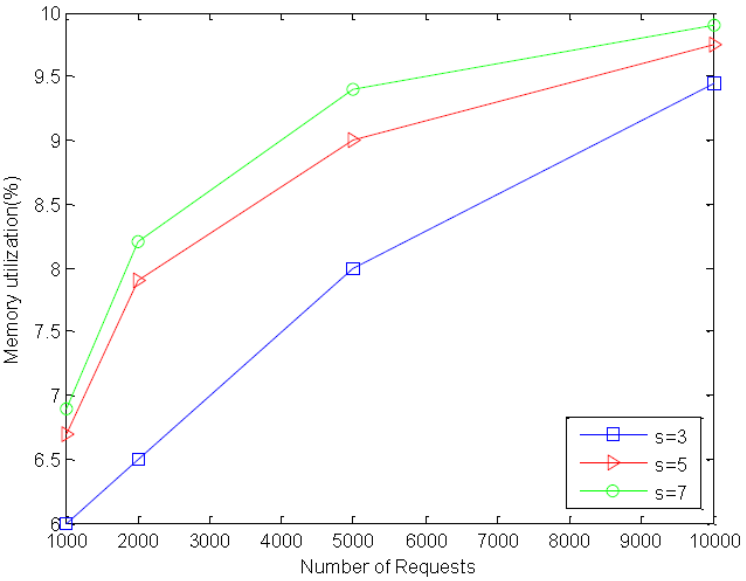
In Figure 4d), firstly, the deployment ‘ $s = 5$ ’ has higher throughput than the deployment ‘ $s = 3$ ’ due to the higher number of requests. Then, the throughput would decrease because of the increasing load of leader, and the deployment ‘ $s = 7$ ’ decreases sharply from the number of requests 1 000 to the number of requests 2 000. Until the limit point which the number of requests from each follower is 5 000, the deployment ‘ $s = 5$ ’ and ‘ $s = 7$ ’ has similar throughput. Thus, when we are ready to increase the number of followers, we also need to consider the leader’s state. Moreover, when the leader approaches its breakdown point, it also influences overall system performance. Thus, the adjustment timing should consider the two conflicting roles as follows.

5.2.3 The Adjustment for the System

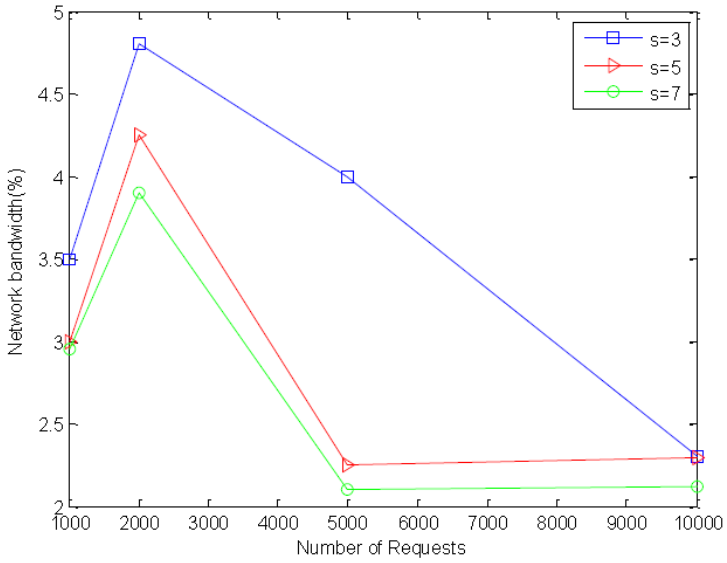
We first consider maximizing the utility in this paper. Load balancing is intended to take full advantage of the resource rather than to only consider the limits of hardware capacity. For dealing with more requests, the system needs to expand the cluster of followers. However, the more burden on the leader would cause the



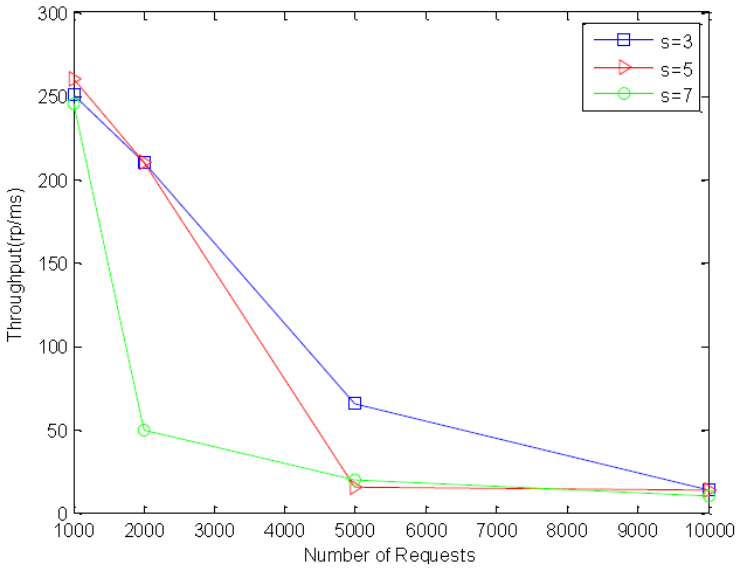
a) The trend of CPU_utilization for the leader



b) The trend of Memory_utilization for the leader



c) The trend of Network_bandwidth for the leader



d) The trend of Throughput for the leader

Figure 4. The capacity of the leader with the increasing number of requests

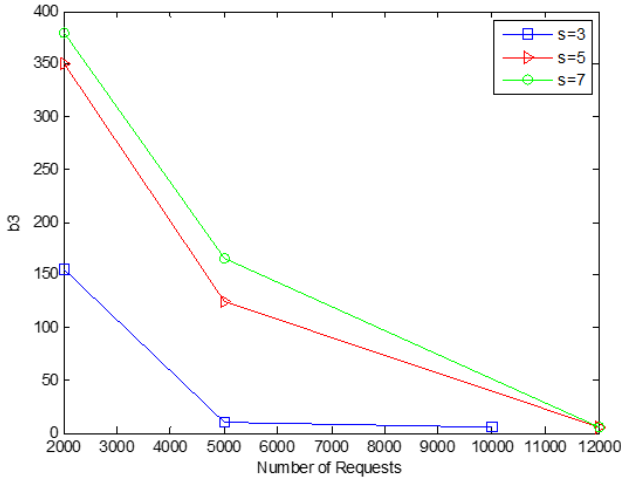


Figure 5. The tendency of the bound as the number of requests increases in ZooKeeper

worse performance. Thus, we combine the two conflicting situations to adjust the deployment. The timing for adjust could depend on various needs of applications, such as the requirement of throughput, bandwidth and memory.

Figure 5 depicts the trend of *bound3*, which is derived from *bound1* and *bound2*. We set the proportions of k_7 and k_8 according to the proportions computed from the communication steps between the follower and the leader. During the complete

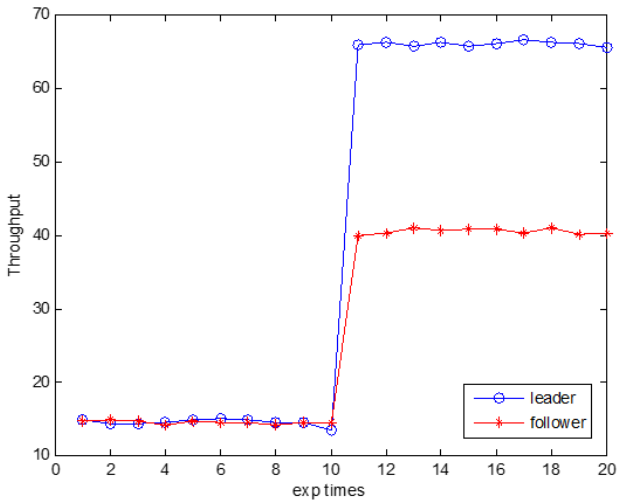


Figure 6. The throughput of the leader and followers before and after adjustment

experimental procedure, *bound1* of the followers decreases as the system scales because of the increasing load. Then, the *bound2* of the leader also decreases because of the realistic platform. Under the same number of requests, *bound1* is smaller when there are fewer followers. When the number of requests is below 5 000, the trend of the ‘ $s = 5$ ’ and ‘ $s = 7$ ’ deployment decreases more sharply than that of the ‘ $s = 3$ ’ deployment. As the deployment of ‘ $s = 3$ ’, before the 5 000-request point, the number of followers could also be decreased. The trend remains stable, decreasing gradually between 5 000 requests and the extreme. This result denotes the timing that the resource utility achieves optimality. Thus, we assume that our actions should be taken after the extreme. Namely, the extreme is the point at which to adjust the followers to maintain a specific throughput volume. For example, the strategy could adjust from the deployment ‘ $s = 3$ ’ to ‘ $s = 5$ ’ as shown here. The similar adjustment could be from the deployment ‘ $s = 5$ ’ to ‘ $s = 7$ ’. Note that the specific adjustment strategy should be further studied according to different objectives. By eliciting the leader’s bound, *bound2*, and the followers’ bound, *bound1*, we deduce the threshold $bound3 \approx 5.30$ for ZooKeeper in our cases. Figure 6 compares the throughput before and after the adjustment. The result demonstrates that the leader and followers are still in the steady phase and that the performance has been improved through the adjustment.

6 CONCLUSIONS

The research here provides a novel perspective for the analysis of ZooKeeper performance. In the emerging large platform, coordination services offer more capacity for the new challenge. In this paper, we developed a stochastic model to describe the coordination process. Based on this model, we derived an optimal strategy for adjusting server deployments in ZooKeeper and observed the results using a real environment. The experimental results demonstrated the trend of various server metrics, and the results revealed the sensitivity of each metric with providing insights for control schemes. Meanwhile, the influences of the performance we quantified could aid in making deployment decisions. Considering the conflicting roles involved in load balancing, thresholds are given to maximize the resource utilization. In the future, we can potentially improve system scalability, reliability and availability based on specific multimedia applications using the distributed platform.

Acknowledgement

This work is funded by: National Key R&D Plan of China under Grant No. 2017Y-FA0604500, National Sci-Tech Support Plan of China under Grant No. 2014BAH02-F00, by National Natural Science Foundation of China under Grant No. 61701190, by Youth Science Foundation of Jilin Province of China under Grant No. 20160520011-JH and 20180520021JH, by Youth Sci-Tech Innovation Leader and Team Project of Jilin Province of China under Grant No. 20170519017JH, by Key Technology Innovation Cooperation Project of Government and University for the whole Industry

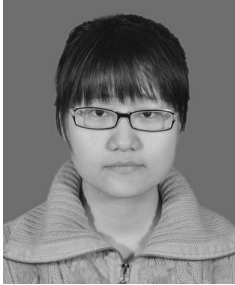
Demonstration under Grant No. SXGJSF2017-4, and by Key Scientific and Technological R & D Plan of Jilin Province of China under Grant No. 20180201103GX.

REFERENCES

- [1] KREUTZ, D.—RAMOS, F. M. V.—VERÍSSIMO, P. E.—ROTHENBERG, C. E.—AZODOLMOLKY, S.—UHLIG, S.: Software-Defined Networking: A Comprehensive Survey. *Proceedings of the IEEE*, Vol. 103, 2015, No. 1, pp. 14–76, doi: 10.1109/JPROC.2014.2371999.
- [2] SAHOO, J.—SALAHUDDIN, M. A.—GLITHO, R.—ELBIAZE, H.—AJIB, W.: A Survey on Replica Server Placement Algorithms for Content Delivery Networks. *IEEE Communications Surveys and Tutorials*, Vol. 19, 2017, No. 2, pp. 1002–1026, doi: 10.1109/COMST.2016.2626384.
- [3] HUNT, P.—KONAR, M.—JUNQUEIRA, F. P.—REED, B.: ZooKeeper: Wait-Free Coordination for Internet-Scale Systems. *USENIX Annual Technical Conference*, Vol. 8, 2010, p. 9.
- [4] Apache HBase. Available at: <http://hbase.apache.org/>.
- [5] MARZ, N.: Storm: Distributed and Fault-Tolerant Realtime Computation. Available at: <https://www.storm-project.net/>, 2015.
- [6] CAI, M.—LIANG, C.—CHEN, W.—SU, H.: Realtime Vehicle Routes Optimization by Cloud Computing in the Principle of TCP/IP. *10th International Conference on Service Systems and Service Management (ICSSSM '13)*, IEEE, 2013, pp. 113–118, doi: 10.1109/ICSSSM.2013.6602650.
- [7] GOEL, L. B.—MAJUMDAR, R.: Handling Mutual Exclusion in a Distributed Application Through ZooKeeper. *International Conference on Advances in Computer Engineering and Applications (ICACEA '15)*, IEEE, 2015, pp. 457–460, doi: 10.1109/ICACEA.2015.7164748.
- [8] SKEIRIK, S.—BOBBA, R. B.—MESEGUER, J.: Formal Analysis of Fault-Tolerant Group Key Management Using ZooKeeper. *2013 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, 2013, pp. 636–641, doi: 10.1109/CCGrid.2013.98.
- [9] OKORAFOR, E.—PATRICK, M. K.: Availability of JobTracker Machine in Hadoop/MapReduce ZooKeeper Coordinated Clusters. *Advanced Computing*, Vol. 3, 2012, No. 3.
- [10] JIANG, C.—DING, Z.—WANG, P. et al.: An Indexing Network Model for Information Services and Its Applications. *2013 IEEE 6th International Conference on Service-Oriented Computing and Applications*, 2013, pp. 290–297.
- [11] KALANTARI, B.—SCHIPER, A.: Addressing the ZooKeeper Synchronization Inefficiency. In: Frey, D., Raynal, M., Sarkar, S., Shyamasundar, R. K., Sinha, P. (Eds.): *Distributed Computing and Networking (ICDCN 2013)*. Springer Berlin Heidelberg, Lecture Notes in Computer Science, Vol. 7730, 2013, pp. 434–438.
- [12] JUNQUEIRA, F. P.—REED, B. C.—SERAFINI, M.: Zab: High-Performance Broadcast for Primary-Backup Systems. *2011 IEEE/IFIP 41st International Confer-*

- ence on Dependable Systems and Networks (DSN '11), 2011, pp. 245–256, doi: 10.1109/DSN.2011.5958223.
- [13] PHAM, C. M.—DOGARU, V.—WAGLE, R. et al.: An Evaluation of ZooKeeper for High Availability in System S. Proceedings of the 5th ACM/SPEC International Conference on Performance Engineering (ICPE '14), 2014, pp. 209–217.
- [14] ABDEL-RAHMAN, M. J.—MAZIED, E. D. A.—TEAGUE, K. et al.: Robust Controller Placement and Assignment in Software-Defined Cellular Networks. 26th International Conference on Computer Communication and Networks (ICCCN '17), IEEE, 2017, pp. 1–9.
- [15] MAROTTA, A.—D'ANDREAGIOVANNI, F.—KASSLER, A.—ZOLA, E.: On the Energy Cost of Robustness for Green Virtual Network Function Placement in 5G Virtualized Infrastructures. *Computer Networks*, Vol. 125, 2017, pp. 64–75.
- [16] CHEN, J.-B.—CHEN, C.-C.: Using Particle Swarm Optimization Algorithm in Multimedia CDN Content Placement. Fifth International Symposium on Parallel Architectures, Algorithms and Programming (PAAP '12), IEEE, 2012, pp. 45–51, doi: 10.1109/PAAP.2012.15.
- [17] JAYASUNDARA, C.—NIRMALATHAS, A.—WONG, E.—CHAN, C.: Improving Energy Efficiency of Video on Demand Services. *IEEE/OSA Journal of Optical Communications and Networking*, Vol. 3, 2011, No. 11, pp. 870–880, doi: 10.1364/JOCN.3.000870.
- [18] LLORCA, J.—TULINO, A. M.—GUAN, K. et al.: Dynamic In-Network Caching for Energy Efficient Content Delivery. *INFOCOM*, 2013, pp. 245–249, doi: 10.1109/INFOCOM.2013.6566772.
- [19] CHOI, N.—GUAN, K.—KILPER, D. C.—ATKINSON, G.: In-Network Caching Effect on Optimal Energy Consumption in Content-Centric Networking. 2012 IEEE International Conference on Communications (ICC '12), 2012, pp. 2889–2894, doi: 10.1109/ICC.2012.6364320.
- [20] LANGE, S.—GEBERT, S.—ZINNER, T. et al.: Heuristic Approaches to the Controller Placement Problem in Large Scale SDN Networks. *IEEE Transactions on Network and Service Management*, Vol. 12, 2015, No. 1, pp. 4–17.
- [21] RAMESH, S.—RHEE, I.—GUO, K.: Multicast with Cache (MCache): An Adaptive Zero-Delay Video-on-Demand Service. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 11, 2001, No. 3, pp. 440–456, doi: 10.1109/INFCOM.2001.916690.
- [22] GUO, M.—AMMAR, M. H.—ZEGURA, E. F.: Selecting Among Replicated Batching Video-on-Demand Servers. Proceedings of the 12th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '02), 2002, pp. 155–163, doi: 10.1145/507670.507692.
- [23] CHANG, C. W.—HUANG, G.—LIN, B. et al.: LEISURE: Load-Balanced Network-Wide Traffic Measurement and Monitor Placement. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 26, 2015, No. 4, pp. 1059–1070.
- [24] KIM, H.-Y.: An Energy-Efficient Load Balancing Scheme to Extend Lifetime in Wireless Sensor Networks. *Cluster Computing*, Vol. 19, 2006, No. 1, pp. 279–283.

- [25] BUI, H.—JOHNSON, A.—JACOB, R. et al.: Multipath Load Balancing for $M \times N$ Communication Patterns on the Blue Gene/Q Supercomputer Interconnection Network. 2015 IEEE International Conference on Cluster Computing (CLUSTER 2015), 2015, pp. 833–840, doi: 10.1109/CLUSTER.2015.140.
- [26] SIMCHI-LEVI, D.—TRICK, M. A.: Introduction to Little’s Law as Viewed on Its 50th Anniversary. *Operations Research*, Vol. 59, 2011, No. 3, p. 535.



Pingting HAO received her B.E. degree in 2013 from Jilin University. She is currently pursuing her Ph.D. in the Department of Computer Science, Jilin University, China. Her research interests include distributed computing, content delivery networks and software defined networks.



Liang HU received his B.Sc. degree from the Harbin Institute of Technology (HIT), Harbin, and M.Sc. and Ph.D. degrees from the College of Computer Science and Technology, Jilin University (JLU), Changchun, China. He has been Professor since 2002 and Ph.D. supervisor since 2003 with the School of Jilin University. His current research interests include distributed computing, network computing and security, data security and privacy.



Jingyan JIANG received her B.E. degree in 2012 from Jilin University. She is currently pursuing her Ph.D. in the Department of Computer Science, Jilin University, China. Her research interests include distributed computing, multimedia networks and software defined networks.



Xilong CHE received his M.Sc. and Ph.D. degrees in computer science from Jilin University, in 2006 and 2009, respectively. Currently, he is Associate Professor and Master Supervisor at the College of Computer Science and Technology, Jilin University, China. His current research areas are parallel and distributed computing, machine learning, and related applications. He is also a member of the IEEE and the corresponding author of this paper.

EXTRA: EXPERTISE-BOOSTED MODEL FOR TRUST-BASED RECOMMENDATION SYSTEM BASED ON SUPERVISED RANDOM WALK

Farshad BAKHSHANDEGAN MOGHADDAM

AIFB, Karlsruhe Institute of Technology

KIT, Karlsruhe, Germany

e-mail: farshad.bakhshandegan-moghaddam@kit.edu

Bahram SADEGHI BIGHAM*

Department of Computer Science and Information Technology

Institute for Advanced Studies in Basic Sciences, Zanjan, Iran

e-mail: b_sadeghi_b@iasbs.ac.ir

Abstract. The quality of recommendations based on any class of recommender systems may become poor if no or low quality data has been provided by users. This is a situation known as *Cold Start* problem, which typically happens when a new user registers to the system and no preference data is available for that user. Trust-Aware Recommendation Systems can be considered as a solution for the cold start problem. In these systems, the trust between users plays an import role for making recommendations. However, most of the Trust-Aware RSs consider trust as a context independent phenomenon which means if user a trusts user b to the degree k then user a trusts user b to the degree k in all the concepts. However, in reality, trust is context dependent and user a can trust user b in context X but not in Y . Moreover, most of the trust-aware RSs do not consider an expertise concept for users and all the users are considered as same in the recommendation process. In this paper we proposed a novel approach for detecting expert users just based on their ratings (unlike previous systems which consider the separate profile and extra information for each user to find an expert). In this model a supervised random walk is exploited to search the trust network for finding experts. Empirical experiments on the Epinions dataset shows that EXTRA can outperform previous models in terms of accuracy and coverage.

Keywords: Recommendation systems, trust, supervised random walk, expertise

* Corresponding author

1 INTRODUCTION

With the rapidly growing amount of information available on the Internet and availability of large amounts of data, finding the information that exactly matches the user's requirements is very difficult. This problem which is known as "Information Overloading" has led to a system which can automatically select the most relevant information and suggest them to users. These systems which are known as recommender systems, find their way to E-Commerce; e.g. there are popular recommender systems for movies¹, books², and so on.

The general task of a *Recommender System (RS)* is to find a list of items which a user may like to find them in the future. Generally, there are two entities which play the main roles in recommender systems, i.e. users and items. Each user can rate a list of items. So, the input of a RS is a User-Item $n \times m$ matrix (n users and m items). This matrix usually is very sparse and the main task of the recommender system is to predict the rating for user u on a non-rated item i or to generally recommend some items for the given user u based on the ratings that already exist. In RS literature, the user u is called "active user" and the item i is called "target item".

One of the most popular and yet effective techniques which are used in recommender systems is the Collaborative Filtering (CF) method [1]. In this method, the system tries to find the same users with the same preferences and aggregate their opinion for predicting the rating for the user u on an item i . The intuition behind the CF is: people often get the best recommendation from someone with a similar taste. Computing the similarity between users is the main bottleneck of this approach. To calculate the similarity of user i and j , most of the similarity functions needs common items which are rated by both user i and j . However, since most users cannot rate (and even are not interested in rating) many items, finding two users with commonly rated items is not easy. Moreover, it is also very challenging to calculate the similarity for users who are new to the system (i.e., cold start users), as they have not rated a significant number of items and hence cannot be properly linked with other similar users.

With the advent of social networks, the new techniques have emerged in recommender systems which are called "trust-aware recommendation systems". In these systems, the inputs is a trust network (a social network which expresses how much the members of the community trust each other) as well as the User-Item matrix. The system uses information about users' profiles and relationships between them to predict the rating of the active user for the target item. In these systems, the trust value between users is exploited instead of the similarity value. However, the trust network is also very sparse and we need a mechanism to calculate the trust value between two unconnected users.

¹ <http://www.netflix.com>

² <http://www.amazon.com>

Many models have emerged so far for calculating the trust value between two unconnected users and incorporating this value in the recommendation phase [2, 3, 4, 5]. The experimental results show that using trust-aware recommendation systems does not significantly alter the accuracy of the system but improves its coverage [2, 3, 6]. However, due to the complex nature of the trust, it is very difficult to calculate the exact trust value between two arbitrary users. Moreover, all the proposed methods for calculating the trust consider the trust as a context-independent phenomenon. It means that if user a trusts user b to the degree k then user a trusts user b to the degree k in all the concepts. However, in the real-world scenarios trust is context dependent and user a can trust user b in concept X but not in Y .

In this paper, we present a model for mining the trust network based on the supervised random walk which, firstly, rectifies the need of calculating the exact trust value between two users; secondly, considers the trust as a context-dependent phenomenon; and thirdly, exploits the expertise of the users to calculate the recommendation. In most of the previous methods, which have been introduced for finding experts in social networks [7, 8, 9], each user has his/her own profile and system; by comparing the profiles (usually by using text-mining techniques) he/she is able to find experts in the context X . In this paper, we propose a new method for finding experts without the need of any metadata for users, and just based on users ratings.

The rest of the paper is organized as follows. We begin our discussion by reviewing the related work in Section 2. Section 3 describes the problem statements. We discuss the details of our proposed model in Section 4. In Section 5 we introduce some properties of our model. The results of an empirical analysis are presented in Section 6 and in Section 7 we conclude the paper and introduce some directions for future research.

2 RELATED WORK

In this section, several major approaches for recommender systems, especially for collaborative filtering and trust-aware recommendation systems, will be reviewed. Two types of collaborative filtering approaches are widely studied: model-based [10, 11] and memory-based [2, 3, 5, 12]. In the model-based approaches, the system tries to learn a model which captures the rating behavior of users. Examples of model-based approaches include the clustering model [13], aspect models [14, 15, 16] and the latent factor model [17]. In the model-based approaches, training datasets are used to train a model. Although the training phase in this approach is very time consuming, it can be performed as an offline preprocess. In these systems, the prediction phase is very fast.

Unlike model-based systems, no learning phase exists in the memory-based systems. The most analyzed examples of memory-based collaborative filtering include user-based approaches [18, 19, 20, 13] and item-based approaches [21, 22, 12]. User-

based approaches predict the ratings of active users based on the ratings of similar users, and item-based approaches predict the ratings of active users based on the computed information of items similar to those chosen by the active user. In the memory-based approach, the system stores the entire User-Item matrix in the memory and tries to use it for prediction phase. In these systems, there is no learning phase but the prediction phase is very time-consuming.

Trust metrics could primarily be divided into ones with a global and ones with a local scope. Global trust metrics can take into account all users and trust relationships between them when computing a trust estimation. Examples can be found in [23, 24, 25] which borrow their ideas from the renowned PageRank algorithm [26]. However, the local trust metric only relies on a part of the trust network, hence taking into account the personal bias. The intuition behind the local trust metrics is that user a may trusts user b but user c may not trust user b [27, 28].

[3] deduced some form of trust graph properties from real networks and proposed an algorithm that has been called TidalTrust [3]. TidalTrust is a modified breadth-first search and looks at the trust values along paths connecting two users. Basically, it tries to find all the raters which are in the shortest path from the source user. Finally returns the aggregated rating of users weighted by the trust value. For calculating the trust value, the source user a asks from its neighbors about the trust value of the user b and finally aggregates responses, weighted by trust values of itself and its direct neighbors.

Massa in [2] introduced MoleTrust. The idea used in the MoleTrust algorithm is similar to the TidalTrust algorithm. MoleTrust also uses breadth-first search. However, it considers maximum-depth for the algorithm and just considers all raters up to a given threshold. The trust value from the source user a to the sink user b is the aggregation of trust values between a and users directly connected to b weighted by the trust values [6]. Figure 1 explains the differences between these two methods.

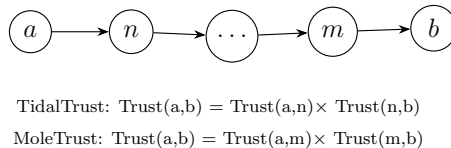


Figure 1. A trust network between user a and b and difference between Tidal Trust and Mole Trust

TrustWalker model has been introduced in [6] as a random walk method to combine the trust-aware approach and the item-based CF approach for predicting the rating of a single item. TrustWalker considers not only the ratings of the target item but also similar items. Research by Jamli et al. [6] proved that TrustWalker outperforms both TidalTrust and MoleTrust models which were introduced before.

More information about the trust-aware recommender systems can be found in [32, 33, 34].

3 PROBLEM STATEMENT

In this section we will formalize the recommendation problem. Typically in a recommendation system there is a set of users $U = \{u_1, \dots, u_N\}$ and a set of items $I = \{I_1, \dots, I_M\}$. Each user can rate a subset of I . This subset is denoted by $RI_u = \{r_{u,i_1}, \dots, r_{u,i_k}\}$. RI can be null (something that occurs in many datasets). The rating of the user u on the item i is denoted by $r_{u,i}$. The $r_{u,i}$ can be classified as unary (like), binary (thumbs up/down) or often a scalar range of $[m, M]$. Usually m represents that the user is completely unsatisfied with the item, and M represents that the user is completely satisfied with the item. Also, in a trust-aware system, there is a trust network among users. We show this network by TN . $t_{a,b}$ denotes the trust value between users a and b . In binary trust networks (such as Amazon, Epinion and eBay), $t_{a,b}$ can be $[0, 1]$. Zero means no trust and one means full trust. We define $N_a = \{b \in U | t_{a,b} = 1\}$ where N_a denotes the set of users directly trusted by a and being neighbors of a . The trust network will currently be outlined as a graph $G = \langle V, E \rangle$ where V represents the users and $E = \{(a, b) | a \in V \& b \in N_a\}$ represents the edges. Each node corresponds to a user and each edge corresponds to a trust statement.

The recommender system is responsible for predicting the rating of an active user $u \in U$ on the target item $i \in I$. The predicted rating is denoted by $P_{u,i}$. Traditional collaborative filtering systems use similar users ratings for obtaining $P_{u,i}$. These systems use a similarity function such as Pearson Correlation, Cosine or Adjusted Cosine for finding similar users. However, as mentioned in Section 1, if two users do not have any rated items in common, the system will fail to calculate the similarity value.

In some of the trust-aware recommender systems techniques, there is no need for calculating the similarity between two users (as the trust value is used instead of the similarity value). Equations (1) and (2) show the difference between these two approaches:

$$P_{u,i} = \frac{\sum_{v \in U, i \in RI_v} sim(u, v) \times r_{v,i}}{\sum_{v \in U, i \in RI_v} |sim(u, v)|}, \quad (1)$$

$$P_{u,i} = \frac{\sum_{v \in U, i \in RI_v} t_{u,v} \times r_{v,i}}{\sum_{v \in U, i \in RI_v} |t_{u,v}|}. \quad (2)$$

Here, $sim(u, v)$ is the degree of similarity between user u and v and $t_{u,v}$ is the degree of trust between two users. Trust-aware recommender systems have adequate performance due to two important principles of sociology. These principles are *Selection* and *Social Influence* effect. The selection means that people prefer to relate to people with similar preferences, and the social influence means that related people in a social network influence each other to become more similar [29, 6].

Various experiments show that using the trust network for recommendation does not improve the accuracy but it can dramatically increase the coverage. Coverage

is the percentage of $\langle u; i \rangle$ pairs which system can compute $P_{u,i}$ for that. Moreover, using the trust network protects the system against attacks like fake profiles due to faked profiles are not being trusted.

4 EXTRA MODEL

The main issues with the current trust-aware systems are:

- Need of having an accurate model to calculate the trust value between two arbitrary users u and v ($t_{u,v}$).
- Unable to capture the real world scenarios due to considering the trust as a context-independent phenomenon.

Moreover, another challenge is to decide how far to go in exploring the trust network. The further we go, the more likely to find raters, but the less trustworthy their ratings become. The random walk model which has been introduced in the [6] can prepare a good trade-off between the precision and the coverage. Although the random walk which we use in EXTRA is identical to the random walk which has been introduced in [6], however, our random walk model differs from that in two ways. First, our random walk is supervised while [6] uses an ordinary one. Second, the stopping criteria of our random walk is different from [6]. The main motivation for proposing this model is that all the trusted users cannot necessarily produce a dependable opinion about the target item i , and the reliability of an opinion must be evaluated based on how much a user u is expert in the field of item i . Based on this fact, the expert user's opinion must be weighted much more than an ordinary user. In this paper, we proposed a new method for finding how much a user u is expert in the item i , solely based on the user u ratings.

To recommend a rating for an active user u_0 on the target item i , we perform random walks on the trust network and select one of the direct neighbors of $u_0(v)$. It must be noted that the probability of selecting each neighbor is different from another and this probability depends on the level of expertise of each neighbor in the item i . The details of how much a user is expert will be discussed later in this section. Each random walk returns a value. We perform several random walks, and the aggregation of all values, returned by different random walks, are considered as the predicted rating $P_{u,i}$.

4.1 Who Is Expert?

Most of the models which exploit expertise concept, consider a profile for their users [7, 8, 9]. This profile encompasses the abilities and interests of each user. When the system is faced with a question (query), it tries to find a user with the highest level of expertise in the question by using text mining techniques (most of the profiles are in the text format) [9]. Also, in [7] authors considered some specific fields and users must promulgate their level of expertise with respect to any of the

fields to the system. Then the entered question is converted to a field vector by the system and comparison of the question vector and the user vector will reveal who is an expert about the question.

To the best of our knowledge there is no system which uses just ratings for defining expertise and finding experts. The authors in [30] showed that there is a relation between the rating similarity and the content similarity. They used MovieLens³ (1 M) dataset and exploited Wikipedia website for extracting the content of each movie in this dataset. Experimental results showed that the correlation between the content of items and item ratings is 14%. This statement can be interpreted in the following way:

If two products have the same rating pattern then their content will be the same with probability of 0.14.

We use this fact to define expertise and present the following definitions:

1. A user is expert in an item i , if he/she rates this item or rates large number of similar products to the item i .
2. A user is expert in an item i , if he/she rates this item or rates a very similar item to the item i .

We consider a user as a fully expert user on an item i if he/she rates item i . The reason is in trust-aware recommender systems, for predicting $P_{u,i}$ the system looks for users who rate the item i in the trust network. For modeling these definitions we need a function to calculate the similarity of 2 items. In this paper, the Pearson Correlation function is used for calculating the similarity between two items:

$$\text{corr}(i, j) = \frac{\sum_{u \in CU_{i,j}} (r_{u,i} - \mu_u)(r_{u,j} - \mu_u)}{\sqrt{\sum_{u \in CU_{i,j}} (r_{u,i} - \mu_u)^2} \sqrt{\sum_{u \in CU_{i,j}} (r_{u,j} - \mu_u)^2}}. \quad (3)$$

$CU_{i,j}$ is the set of common users who have rated both items i and j , and μ_u denotes the average of ratings expressed by u . $\text{corr}(i, j)$ denotes the correlation of items i, j . Values of this function are in the range $[-1, 1]$. Based on the results of [30], if two items have a Pearson correlation to the degree p , then with the probability $0.14 \times p$ their contents will be the same. Obviously this value is very small, however, our experiments show that exploiting this small value increases the accuracy and coverage of recommender systems. We should note that, if p is a positive number, then $0.14 \times p$ will present how similar are two items and, on the contrary, if p is a negative number, then the $0.14 \times |p|$ will present how dissimilar are two items. Now we can formally define the expertise concept.

Definition 1. Considering the number of items which have been rated by user u and have positive correlation with the target item i as expertise of user u in the item i .

³ <http://grouplens.org/datasets/movielens/>

$$\psi(u, i) = \begin{cases} 1, & \text{if } r_{u,i} \neq \emptyset, \\ \frac{|\{j \in RI_u \mid Corr(i,j) > 0\}|}{\sum_{v \in N_u} |\{j \in RI_v \mid Corr(i,j) > 0\}|}, & O.W. \end{cases} \quad (4)$$

In this equation $\psi(u, i)$ denotes the expertise of user u in the item i and RI_u denotes all the items which have been rated by u .

Definition 2. Considering the maximum similarity of items which have been rated by user u and the target item i as the expertise of user u in the item i .

$$\psi(u, i) = \begin{cases} 1, & \text{if } r_{u,i} \neq \emptyset, \\ \max_{j \in RI_u} Corr(i, j), & O.W. \end{cases} \quad (5)$$

The values of Definition 2 are in the range $[0, 1]$, but the values of Definition 1 must be normalized to be usable as a probability for the random walk.

4.2 Random Walk

In this section we describe one step of a random walk and show how the probability of each edge is calculated. Although the trust network which is used in this paper is a binary network, but we exploit a supervised random walk where the probability of each edge is different from another.

We perform the random walk from the source user u_0 (line 3 of Figure 2). At first step, we have to select one of the directly trusted neighbors of u_0 (line 6 of Figure 2). But first, the probability of edges must be calculated (line 5 of Figure 2). In this model, instead of selecting neighbors uniformly, we assign the expertise of each neighbor in item i as the edge probabilities. This makes it possible to visit the experts in item i more than other users. We define S_u as a random variable for selecting a user v from N_u :

$$p(S_u = v) = \frac{\psi(v, i)}{\sum_{w \in N_u} \psi(w, i)}. \quad (6)$$

Here, N_u is the set of all the neighbors of u and $\psi(u, i)$ denotes the expertise of user u in the item i .

Let us assume the user u is selected. If u already has the rating on the target item i (u is fully expert in the item i), then we stop the process and return $r_{u,i}$ as a result (lines 7–9 of Figure 2). However, if u does not have a rating on i , then we have two options:

- With probability α stop the random walk and return the weighted mean of ratings of items which are rated by u (lines 12–14 of Figure 2).
- With probability $1 - \alpha$ continue the random walk and select one of the direct neighbors of u (lines 16–17 of Figure 2).

α is the probability of stopping the random walk at the node u . As mentioned earlier, by going far from the active user, the data will be less trustworthy. Thus the number of random walk steps (k) must be involved in the calculation of α to avoid and penalize long random walks. Sigmoid function can satisfy this constraint [6]. This function gives values 1 for big values of k , and 0 for small values of k . Thus, α is defined by Equation (7):

$$\alpha = \psi(u, i) \times \frac{1}{1 + e^{-k}}. \quad (7)$$

As each edge has a probability, there is a chance for a single random walk to continue forever. Therefore, we consider a threshold for going far from the active user (line 16 of Figure 2). We set this threshold to 6 based on the idea of “six-degrees of separation” [6, 31] and terminate the random walk when $k > 6$.

4.3 Termination Condition

Due to the probabilistic nature of a Random Walk (RW), the result of one RW cannot be accurate. Several random walks must be performed to produce a more reliable prediction. In this model, we use the TrustWalker termination condition model which uses the variances of obtained result of several random walks to stop the overall algorithm. As stated in [6], the variance of the results of all the random walks is used as follows:

$$\sigma^2 = \frac{\sum_{i=1}^T (r_i - \mu)^2}{T}. \quad (8)$$

Here, r_i is the result of i^{th} random walk, and μ denotes the average of the ratings returned by random walks. T is the number of random walks which is performed to compute the prediction. Since ratings are in a finite range of $[1, 5]$, it has been proved in [6] that σ^2 converges to a constant value. Moreover, σ_i^2 is defined as the variance of the results of the first i random walks. So EXTRA terminates if $|\sigma_{i+1}^2 - \sigma_i^2| \leq \epsilon$ (line 4 of Figure 2). In this algorithm, we consider the threshold of 100 for the maximum number of unsuccessful random walks, and after that, we consider the pair $\langle user, item \rangle$ as non-covered [6] (lines 18–19 of Figure 2). The quality of the output depends on the value of ϵ . The smaller ϵ the more accurate results. But we should note that by setting ϵ to small values, the running time of the algorithm will be increased. Thus, we set $\epsilon = 0.001$ for the termination condition (the reason will be explained in Section 6). The complete algorithm of the EXTRA model can be studied in Figure 2.

5 PROPERTIES OF EXTRA MODEL

The EXTRA model (same as TrustWalker) has the potential to be converted to the previous models and this shows the generality of this model. If $\alpha = 1$ for all $u \in U$, then our random walk will never start, and it will return weighted average of the

```

1: Initial  $\sigma^2$ ,  $\epsilon$ , Max_Steps
2: for All  $U \in \text{Users}$  do
3:   Active User =  $U$ ;
4:   while  $\sigma^2 > \epsilon$  do
5:     Assign a value to edges base on users expertise;
6:      $S = \text{Select one of the Active User neighbors}$  ( $S \in N_{ActiveUser}$ );
7:     if  $S$  has a rating for target item then
8:       Return the rating value;
9:     end if
10:    Calculate  $\alpha$  by Equation (7);
11:     $R = \text{Random Number}$ ;
12:    if  $R < \alpha$  then
13:       $SI = \text{weighted-mean of } S' \text{ ratings list}$ ;
14:      Return  $SI$ ;
15:    else
16:      if Steps < Max_Steps then
17:        Continue the Random Walk;
18:      else
19:        Return "Cannot cover";
20:      end if
21:    end if
22:    Calculate  $\sigma^2$ ;
23:  end while
24: end for

```

Figure 2. The EXTRA algorithm

ratings on items in RI_{u_0} . This is the same as the result of Item-based collaborative filtering proposed in [12].

On the other hand, if we consider that the expertise of all users is equal ($\forall u, v \in U, \psi(u, i) = \psi(v, i)$) and change the termination condition as $\alpha = 1$ if $r_{u,i} \neq \emptyset$, then all random walks will continue until they have found a rating for the exact target item i . These changes will convert the EXTRA to existing methods which are explained in [3, 2].

Moreover, if we consider that the expertise of all users is equal ($\forall u, v \in U, \psi(u, i) = \psi(v, i)$) and change the termination condition as using maximum similarity instead of ψ ($\alpha = \max sim(i, j) \times \frac{1}{1+e^{-\frac{k}{2}}}$), then the EXTRA will be converted to the TrustWalker model [6].

6 EXPERIMENTS

In this section the result of our experiments is reported. In our experiments the EXTRA algorithm is compared against some trust-aware methods such as Trust-

Walker [6] and pure random walk and some fundamental similarity based recommendation methods such as User-Based Collaborative Filtering [1] and Item-Based Collaborative Filtering [12].

6.1 Dataset Description and Experimental Design

Most of the datasets in the recommender system have no social network among users. We choose Epinions⁴ as the data source for our experiments which has both a trust network and ratings expressed by users. Epinions.com is a consumers opinion site where users can review items and also assign numeric ratings in the range of [1–5]. Every member of Epinions maintains a “trust” list which presents a network of trust relationships between users, and a “block (distrust)” list which presents a network of distrust relationships. Inserting a user in the Web of Trust equals to issuing a trust statement with value 1 while inserting a user in the Block List equals to issuing a trust statement of value 0.

The dataset used in our experiments consists of 49 290 users who rated a total number of 139 738 different items at least once. The total number of reviews is 664 824. The total number of issued trust statements is 487 181. The rating matrix sparsity is 99.99135 %. In this dataset 52.82 % of users are cold start users (users with less than 5 ratings (same as [22]) which is a huge portion of users. So, considering the performance of recommendation for cold start users is very important. Table 1 shows the details of this dataset. Figure 3 shows the rating distribution over users in Epinion dataset. It is notable that 5 is the dominated rating in this dataset.

Number of users	49 290
Number of items	139 738
Number of trust statements	487 181
Number of reviews	664 824
Average number of out-degree	9.8768
Average number of item each user rate	13.4883
Number of users who do not have any neighbor	15 347
Number of users who rate less than 5 items	26 037

Table 1. Epinions dataset features

The main parameters of the EXTRA algorithm are the number of steps for random walks and the ϵ . Based on Milgram experiment [31], we set 6 for the longest random walks. However, choosing the ϵ is an important part in the EXTRA algorithm. The smaller ϵ we choose, the higher precision we achieve, but the time complexity increases dramatically. So, different values of ϵ were tested with EXTRA-Num (all the values have been normalized to [0, 1]). We considered the number of steps as 2. Based on Table 2 and Figure 4, we set $\epsilon = 0.01$, because in this case the $\frac{\text{Precision}}{\text{TimeComplexity}}$ is maximal.

⁴ http://www.trustlet.org/wiki/Downloaded_Epinions_dataset

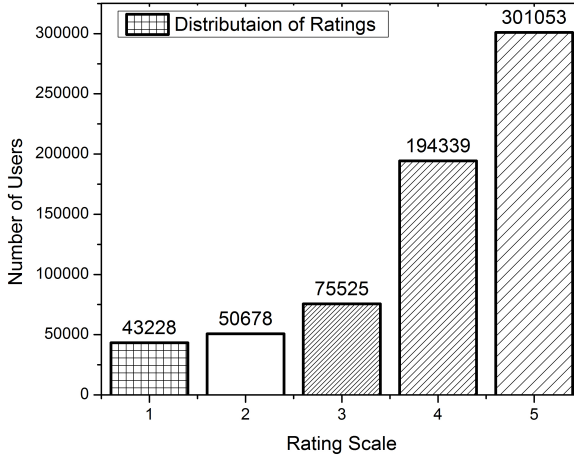


Figure 3. Rating distribution of Epinions dataset

ϵ	100	10	1	0.1	0.01	0.001
<i>Precision</i>	0.60	0.61	0.62	0.70	0.75	0.76
<i>Time(Sec)</i>	25 002	25 920	27 012	30 121	30 310	41 002
$\frac{Precision}{Time}$	2.42×10^{-5}	2.35×10^{-5}	2.31×10^{-5}	2.32×10^{-5}	2.47×10^{-5}	1.85×10^{-5}
$\frac{Precision}{Time}$ (Normalized)	0.90	0.79	0.71	0.75	1	0

Table 2. Results for different values of ϵ

We implemented all methods in MATLAB 7.10. We used an Intel Core i5 2.4 GHz CPU with 4 GB RAM to run our experiments on a Windows 7 system.

In this article we will review the results of the various models as follows:

TrustWalker Model: This method is a random walk method based on the trust and the item similarity [6].

Random Walk (pure): This method is an extension of TrustWalker which stops the random walks whenever finds a user with rated target item.

User-Based Collaborative Filtering: This method is the classic user-based collaborative filtering method, which makes prediction based on the user similarity [1].

Item-Based Collaborative Filtering: This method is the classic item-based collaborative filtering method, which makes prediction based on the item similarity [12].

EXTRA-Num: This method in the proposed model which uses Definition 1 for the calculation of user expertise.

EXTRA-Max: This method in the proposed model which uses Definition 2 for the calculation of user expertise.

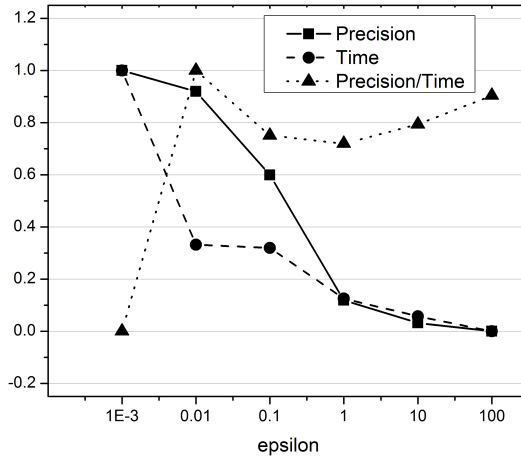


Figure 4. Results for different values of the ϵ

6.2 Accuracy Metrics

We use one-leave-out method to evaluate the recommender system. In this technique, one of the ratings has been masked and the system tries to predict it using the trust network [3, 2, 12]. Finally calculated value and the masked value are compared. For measuring the prediction quality we use the Root Mean Squared Error (RMSE) metric.

$$RMSE = \sqrt{\frac{\sum_{(u,i)} (r_{u,i} - P_{u,i})^2}{N}} \tag{9}$$

where $r_{u,i}$ denotes the rating of the user u on an item i , $P_{u,i}$ denotes the rating of the user u on an item i as predicted by the system, and N denotes the number of pairs (u, i) which u has a rating on i . The smaller the value of RMSE, the more precise the recommendation is.

Another important measure is the coverage. We define a coverage metric to measure the percentage of pairs of $(user, item)$ for which system can predict a rating.

To have a single evaluation metric and be able to compare the results with the previous models, we can combine RMSE and coverage. F-score can be used for this

purpose. However, first, RMSE should be converted to a precision in the range of $[0, 1]$. As the values of ratings are in the range of $[1-5]$, we define the precision as follows:

$$Precision = 1 - \frac{RMSE}{max\ error} \quad (10)$$

where $max\ error$ is $5 - 1 = 4$. We define F-score as follows:

$$F\text{-score} = \frac{2 \times Precision \times Coverage}{Precision + Coverage}. \quad (11)$$

6.3 Results

In this section, we summarize and discuss the most important results of the presented experiments. Since more than half of the users are cold start users, considering the performance of the recommendation for cold start users is very important. Table 3 shows the RMSE, Coverage, and F-score for cold start users. We consider the maximum step of the random walk to 6 for all the algorithms. It should be mentioned that because of the probabilistic nature of the random walk, all the methods which exploited random walk were performed 5 times and an average of the results has been reported. Another point is, as the EXTRA algorithm is a greedy method which works locally and helps random walks to choose the best option for the next step, we set $\psi(u, i) = \varepsilon$ in case $\psi(u, i) = 0$, in order to give a chance to non-expert users to be selected and help EXTRA to escape from the local optimums.

As shown in Table 3, the proposed EXTRA model outperforms other methods according to each three criteria for the cold start users.

Notice that most of the methods which exploit trust network outperform basic methods according to RMSE and coverage. Random Walk (pure) is the only method that has lower coverage than others. Because it tries to find a user who has a rating for the target item, and as it just considers limited steps, finding a user with rated target item is not easy. Among all these methods (except EXTRA) TrustWalker has better RMSE and better coverage. The EXTRA-Num model outperforms TrustWalker by exploiting supervised random walk and considering the rating of expert users. Figures 5 to 8 compare the results of different methods according to the RMSE by considering the step of random walks.

As shown in Figures 5 to 8, there is no significant change in the RMSE after step 2 in the models which use the trust network. It indicates that considering the friends and a friend of friends is enough for the recommendation process and there is no need to consider further users. Considering more users not only does not change the RMSE, it also increases the running time of the system.

The results for all users are shown in Table 4. We observe the similar relative performance of all methods as for cold start users. It should be noted that all methods perform significantly better over all users since there is more information available than cold start users. EXTRA-Num outperforms all other methods in terms of RMSE, Coverage, and F-score.

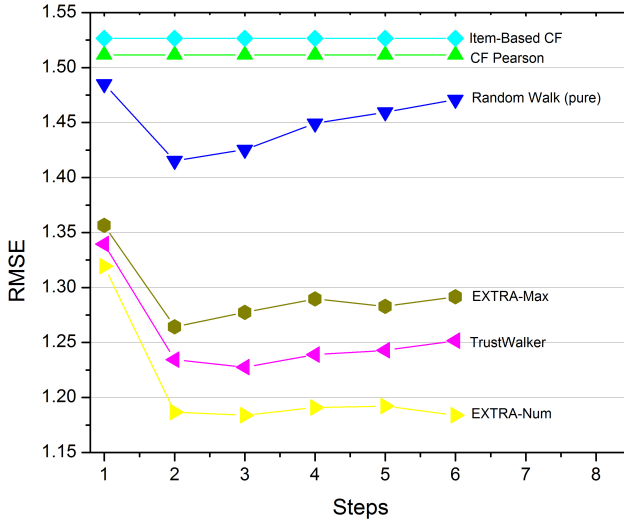


Figure 5. Comparison of RMSE for different methods on cold start users

Model	RMSE	Coverage	F-score
CF Pearson	1.51158	21.79916	0.32285
Random Walk (pure)	1.41527	17.25901	0.27242
Item-based CF	1.52669	28.17851	0.38714
TrustWalker	1.23438	68.62673	0.68883
EXTRA-Num	1.18678	74.76602	0.7248
EXTRA-Max	1.26438	74.21729	0.71185

Table 3. Results for cold start users

The results show that EXTRA-Num can better capture the expertise concept than EXTRA-Max. That is due to the fact that in Definition 2 we consider the maximum similarity of items as the user's expertise, however, imagine there is a user who rates 100 items which 99 of them have a negative correlation with target item but 1 has correlation 1. So, based on Definition 2, we consider this user as a fully expert user in item i , while it is not true.

In summary, EXTRA substantially not only improves the coverage of existing trust-aware approaches, it also improves the precision. This improvement is achieved by considering ratings of users which are expert in the target item, because ratings which are expressed by experts are more reliable than ratings expressed by ordinary users. For example, EXTRA-Num's RMSE is 1.18678 for cold start users while TrustWalker approaches have RMSE of 1.23438. Also, in the case of all

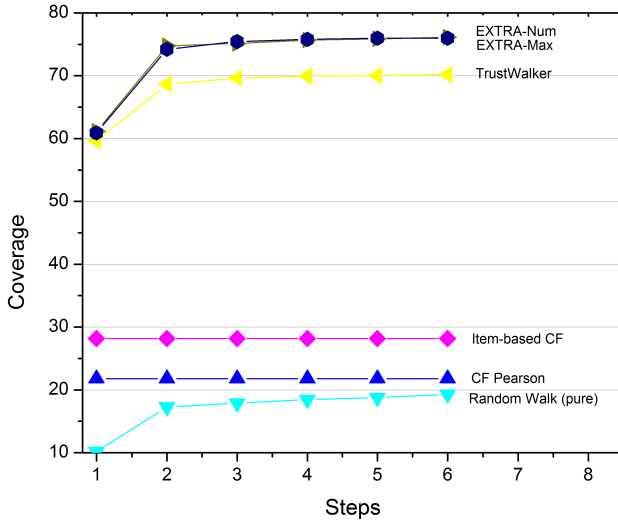


Figure 6. Comparison of Coverage for different methods on cold start users

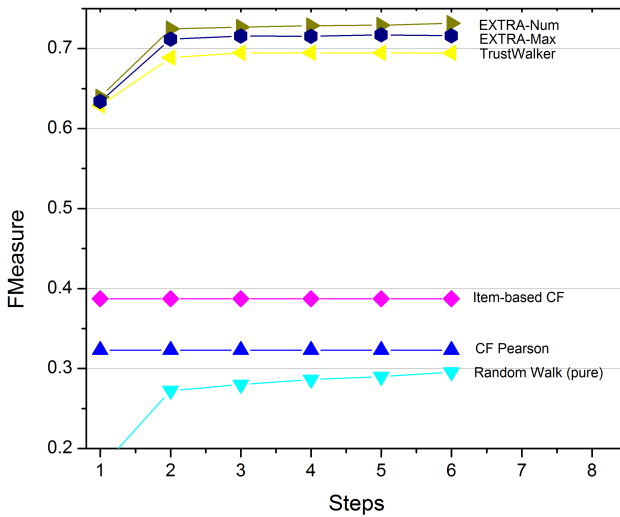


Figure 7. Comparison of F-score for different methods on cold start users

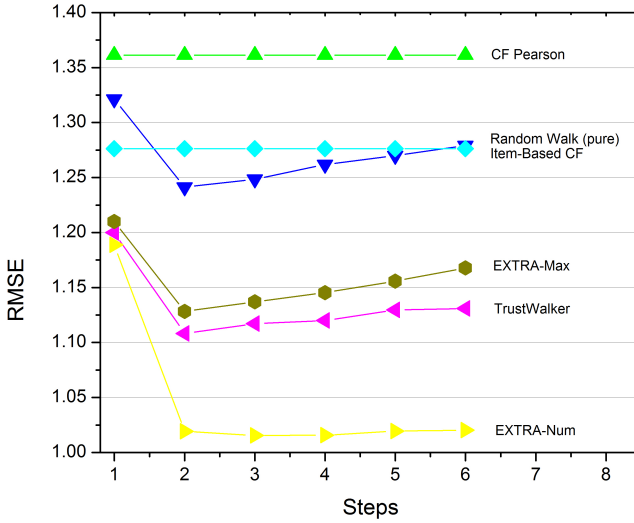


Figure 8. Comparison of RMSE for different methods on all users

Model	RMSE	Coverage	F-score
CF Pearson	1.36135	71.03196	0.68405
Random Walk (pure)	1.24128	60.3375	0.64365
Item-based CF	1.2762	70.0998	0.69083
TrustWalker	1.10822	92.8000	0.81274
EXTRA-Num	1.0193	94.76837	0.83432
EXTRA-Max	1.12837	94.76752	0.81694

Table 4. Results for all users

users, EXTRA-Num’s RMSE is 1.0193 while TrustWalker approaches have RMSE of 1.10822. The 9% RMSE reductions is an evidence for the significant improvement of accuracy of the trust-aware recommender systems.

7 CONCLUSIONS

In this paper, we presented a new approach for enhancing trust-aware recommender systems by exploiting expert users. We introduced a mechanism for finding experts just based on the ratings they make on items and consider just these users for prediction in the recommendation process.

The evaluation on the Epinions dataset showed that EXTRA outperforms both basic and pure trust-aware methods especially in terms of RMSE and coverage. Also, we showed that considering the friends and friends of a friend is adequate

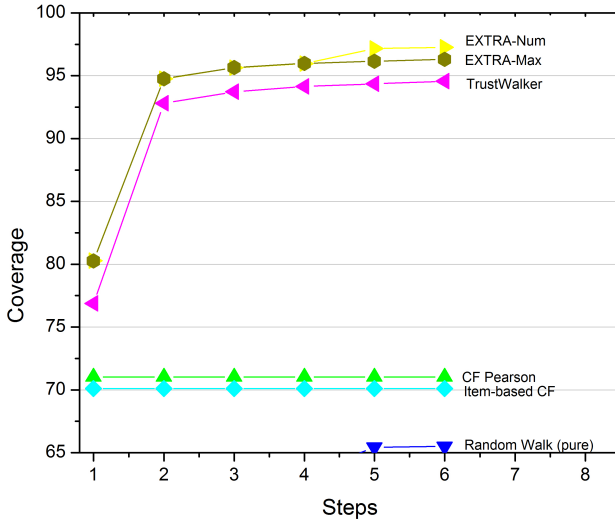


Figure 9. Comparison of Coverage for different methods on all users

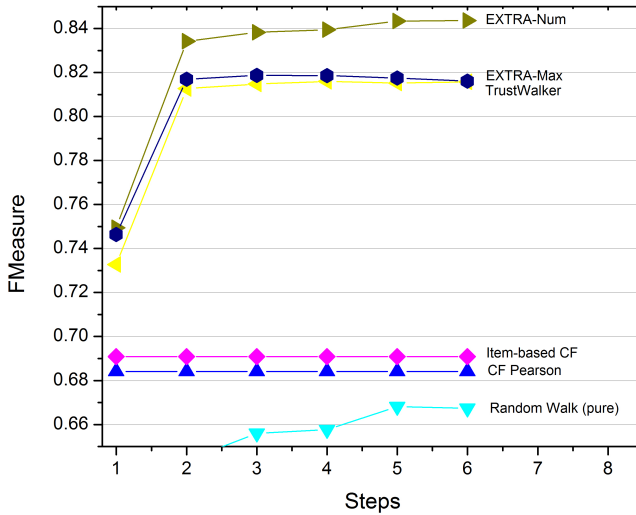


Figure 10. Comparison of F-score for different methods on all users

in trust-aware recommender systems and there is no need to go deeper into trust networks.

There are various ways to extend the EXTRA method for future works. This method can be evaluated against other datasets such as Flixster⁵ with different properties. Flixster is a social networking service in which users can rate movies and create a social network. It is bigger and denser than the Epinions dataset. Moreover, defining expertise in a way that properly captures the context of expertise in real-world scenarios is another future work. In this paper, we assumed that the ratings are stored in a centralized repository. So, we could easily calculate the similarity between the two items for finding expert users. However, decentralized applications can have multiple repositories. So calculating the similarity of items in distributed repositories is another future work.

REFERENCES

- [1] GOLDBERG, D.—NICHOLS, D.—OKI, B. M.—TERRY, D.: Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM*, Vol. 35, No. 12, 1992, doi: 10.1145/138859.138867.
- [2] MASSA, P.—AVESANI, P.: Trust-Aware Recommender Systems. *ACM Conference on Recommender Systems (RecSys'07)*, USA, 2007, pp. 17–24, doi: 10.1145/1297231.1297235.
- [3] GOLBECK, J. A.: Computing and Applying Trust in Web-Based Social Networks. Ph.D. thesis, University of Maryland at College Park, 2005.
- [4] LEVIEN, R.—AIKEN, A.: Advogato's Trust Metric. Available at: <http://advogato.org/trust-metric.html>, 2002.
- [5] ZIEGLER, C.-N.: Towards Decentralized Recommender Systems. Ph.D. thesis, University of Freiburg, 2005.
- [6] JAMALI, M.—ESTER, M.: TrustWalker: A Random Walk Model for Combining Trust-Based and Item-Based Recommendation. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, 2009, pp. 397–406, doi: 10.1145/1557019.1557067.
- [7] TRIAS MANSILLA, A.—DE LA ROSA ESTEVA, J. L.: Propagation of Question Waves by Means of Trust in a Social Network. In: Christiansen, H., De Tré, G., Yazici, A., Zadrozny, S., Andreasen, T., Larsen, H. L. (Eds.): *Flexible Query Answering Systems (FQAS 2011)*. Springer, Berlin, Heidelberg, *Lecture Notes in Computer Science*, Vol. 7022, 2011, pp. 186–197.
- [8] SHAH, P.: Expert Finding Using Social Networking. Master's Thesis, San Jose State University, 2009.
- [9] DAVOODI, E.—AFSHARCHI, M.—KIANMEHR, K.: A Social Network-Based Approach to Expert Recommendation System. In: Corchado, E., Snášel, V., Abraham, A., Woźniak, M., Graña, M., Cho, S. B. (Eds.): *Hybrid Artificial Intelligent*

⁵ <https://www.flixster.com/>

- Systems (HAIS 2012). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7208, 2012, pp. 91–102.
- [10] RETTINGER, A.—NICKLES, M.—TRESP, V.: A Statistical Relational Model for Trust Learning. Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems – Volume 2 (AAMAS '08), 2008, pp. 763–770.
 - [11] MA, H.—YANG, H.—LYU, M. R.—KING, I.: SoRec: Social Recommendation Using Probabilistic Matrix Factorization. Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM '08), 2008, pp. 931–940, doi: 10.1145/1458082.1458205.
 - [12] SARWAR, B.—KARYPIS, G.—KONSTAN, J.—RIEDL, J.: Item-Based Collaborative Filtering Recommendation Algorithms. Proceedings of the 10th International Conference on World Wide Web (WWW '01), 2001, pp. 285–295, doi: 10.1145/371920.372071.
 - [13] XUE, G.-R.—LIN, C.—YANG, Q.—XI, W.—ZENG, H.-J.—YU, Y.—CHEN, Z.: Scalable Collaborative Filtering Using Cluster-Based Smoothing. Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '05), 2005, pp. 114–121, doi: 10.1145/1076034.1076056.
 - [14] HOFMANN, T.: Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis. Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '03), 2003, pp. 259–266, doi: 10.1145/860435.860483.
 - [15] HOFMANN, T.: Latent Semantic Models for Collaborative Filtering. ACM Transactions on Information Systems, Vol. 22, 2004, No. 1, pp. 89–115, doi: 10.1145/963770.963774.
 - [16] SI, L.—JIN, R.: Flexible Mixture Model for Collaborative Filtering. Proceedings of the 20th International Conference on Machine Learning (ICML '03), 2003, pp. 704–711.
 - [17] CANNY, J.: Collaborative Filtering with Privacy via Factor Analysis. Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '02), 2002, pp. 238–245, doi: 10.1145/564376.564419.
 - [18] BREESE, J. S.—HECKERMAN, D.—KADIE, C.: Empirical Analysis of Predictive Algorithms for Collaborative Filtering. Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI '98), 1998, pp. 43–52.
 - [19] HERLOCKER, J. L.—KONSTAN, J. A.—BORCHERS, A.—RIEDL, J.: An Algorithmic Framework for Performing Collaborative Filtering. Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99), 1999, pp. 230–237, doi: 10.1145/312624.312682.
 - [20] JIN, R.—CHAI, J. Y.—SI, L.: An Automatic Weighting Scheme for Collaborative Filtering. Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '04), 2004, pp. 337–344, doi: 10.1145/1008992.1009051.
 - [21] DESHPANDE, M.—KARYPIS, G.: Item-Based Top-N Recommendation Algorithms. ACM Transactions on Information Systems, Vol. 22, 2004, No. 1, pp. 143–177, doi: 10.1145/963770.963776.

- [22] LINDEN, G.—SMITH, B.—YORK, J.: Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*, Vol. 7, 2003, No. 1, pp. 76–80.
- [23] KAMVAR, S.—SCHLOSSER, M.—GARCIA-MOLINA, H.: The EigenTrust Algorithm for Reputation Management in P2P Networks. *Proceedings of the Twelfth International Conference on World Wide Web (WWW'03)*, 2003, pp. 640–651, doi: 10.1145/775152.775242.
- [24] GUHA, R.: Open Rating Systems. Technical report, Stanford Knowledge Systems Laboratory, Stanford, 2003.
- [25] RICHARDSON, M.—AGRAWAL, R.—DOMINGOS, P.: Trust Management for the Semantic Web. In: Fensel, D., Sycara, K., Mylopoulos, J. (Eds.): *The Semantic Web (ISWC 2003)*. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2870, 2003, pp. 351–368.
- [26] PAGE, L.—BRIN, S.—MOTWANI, R.—WINOGRAD, T.: The Pagerank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.
- [27] LEVIEN, R.—AIKEN, A.: An Attack-Resistant, Scalable Name Service. Draft Submission to the Fourth International Conference on Financial Cryptography, 2000.
- [28] GOLBECK, J.—PARSIA, B.—HENDLER, J.: Trust Networks on the Semantic Web. In: Klusch, M., Omicini, A., Ossowski, S., Laamanen, H. (Eds.): *Cooperative Information Agents VII (CIA 2003)*. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2782, 2003, pp. 238–249.
- [29] WASSERMAN, S.—FAUST, K.: *Social Network Analysis*. Cambridge University Press, 1994, doi: 10.1017/CBO9780511815478.
- [30] BASKAYA, O.—AYTEKIN, T.: How Similar is Rating Similarity to Content Similarity? Workshop on Recommendation Utility Evaluation: Beyond RMSE (RUE 2012) held in conjunction with ACM RecSys 2012, Dublin, Ireland, 2012, pp. 27–29.
- [31] MILGRAM, S.: The Small World Problem. *Psychology Today*, Vol. 1, 1967, No. 1, pp. 61–67.
- [32] OZSOY, M. G.—POLAT, F. S.: Trust Based Recommendation Systems. *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM'13)*, 2013, pp. 1267–1274, doi: 10.1145/2492517.2500276.
- [33] BABY, B.—MURALI, S.: A Survey on Trust Based Recommendation Systems. *International Research Journal of Engineering and Technology (IRJET)*, Vol. 3, 2016, No. 10, pp. 1021–1023.
- [34] AFEF, S.—BRAHMI, Z.—GAMMOUDI, M. M.: Trust-Based Recommender Systems: An Overview. 27th IBIMA Conference, Milan, Italy, 2016.



Farshad BAKHSHANDEGAN MOGHADDAM is computer science Ph.D. candidate at Karlsruhe Institute of Technology (KIT). His research interests include various aspects of recommender systems, semantic web and natural language processing. He is currently focusing on word and network embedding models and their usage in recommendation systems. Prior to KIT, he was fortunate to be advised by Bahram Sadeghi as Master student at the Institute for Advanced Studies in Basic Sciences (IASBS), Zanzan, Iran.



Bahram SADEGHI BIGHAM is Associate Professor in computer sciences at the Institute for Advanced Studies in Basic Sciences (IASBS). His research interests are in the areas of robot motion planning, computational geometry, algorithms, AI and medical applications. Prior to arriving at IASBS, he worked as Postdoctoral Fellow at the University of Cardiff in the School of Computer Science. In June 2008, he completed his Ph.D. at Amirkabir University of Technology (Tehran Polytechnic), where he also completed an M.Sc. in 2000. His B.Sc. is from University of Birjand in mathematics.

ONTOLOGY-BASED RESOLUTION OF CLOUD DATA LOCK-IN PROBLEM

Darko ANDROČEĆ, Neven VRČEK

Faculty of Organization and Informatics

University of Zagreb

Pavlinska 2

42000 Varaždin, Croatia

e-mail: {dandrocec, nvrcek}@foi.hr

Abstract. Cloud computing is nowadays becoming a popular paradigm for the provision of computing infrastructure that enables organizations to achieve financial savings. On the other hand, there are some known obstacles, among which vendor lock-in stands out. Furthermore, due to missing standards and heterogeneities of cloud storage systems, the migration of data to alternative cloud providers is expensive and time-consuming. We propose an approach based on Semantic Web services and AI planning to tackle cloud vendor data lock-in problem. To complete the mentioned task, data structures and data type mapping rules between different types of cloud storage systems are defined. The migration of data among different providers of platform as a service is presented in order to prove the practical applicability of the proposed approach. Additionally, this concept was also applied to software as a service model of cloud computing to perform one-shot data migration from Zoho CRM to Salesforce CRM.

Keywords: Cloud data portability, data migration, platform as a service, software as a service, data type mappings, semantic web services

Mathematics Subject Classification 2010: 68-P20

1 INTRODUCTION

Many end users and corporations store parts of their data in clouds. For now, it is not easy to move these data from one cloud vendor to another. Due to different cloud

storage models and implementations, it is difficult for users to switch to another cloud storage solution in case of service dissatisfaction or change of users' needs. The cloud data lock-in problem is characterized by time-consuming and costly migration of data to alternative cloud solutions offered by different vendors. Currently, each cloud vendor offers its own tools, remote application programming interfaces (APIs), and cloud storage data models. The numerous heterogeneities among different vendors make cloud data portability an interesting and complex research and practical problem. The migration of data between different offers of platform as a service is the main focus of this work. The same approach was applied to software as a service model, where we have chosen to migrate data between two cloud customer relationship management (CRM) systems: Zoho CRM and Salesforce.

The presence of several coexisting cloud storage models introduces the need for translation techniques and tools. The main research questions of this paper can be stated as: How to leverage automatic data migration between cloud solutions? How to provide cross cloud data type mappings for different cloud storage systems used by various cloud offers? There are many data migration/interoperability problems among cloud providers, so the answer to the mentioned research questions is not trivial. For example, there exists a difference between data storage models of various commercial cloud providers; storage data types differ in name, value space, permitted range of values, precision of data; data import or export is often complex; some offers have a predefined standard data container or have some naming restrictions. The main contribution of this work is a flexible data migration among cloud storage systems that uses Semantic Web and AI planning to avoid point-to-point mappings. We have used a standard data model in the form of the OWL ontology representing data schema, data types, and data elements. The result of this work is the architecture design for automated data migration among different cloud providers.

This paper proceeds as follows. First, in Section 2, the related work is listed. Section 3 lists the main steps of our research. In Section 4 we present the chosen PaaS offers and storage types. Section 5 shows how we use OWL as an intermediate format for the migration of data structure and data from one PaaS storage to another. Section 6 briefly describes the developed PaaS ontology and our implementation of data type mappings between different PaaS storage offers. The following three sections show the approach we proposed, and explain the use of Semantic Web services, AI planning, and Apache CXF framework to migrate data between PaaS providers. In Section 10, the validation of our data migration approach was done by migrating a more complex set of PaaS and SaaS data. Our conclusions are provided in the final section.

2 RELATED WORK

Cloud storage interoperability problems are similar to interoperability conflicts among multiple independent database systems. These problems are well investi-

gated in the current literature. For example, Sheth and Kashyap [1] classified and defined the most important interoperability conflicts among multiple independent database systems. They listed the following main categories of incompatibilities [1]: domain definition incompatibility (attributes have different domain definitions), entity definition incompatibility (descriptors used for the same entity are partially compatible), data value incompatibility (inconsistency between related data), abstraction level incompatibility (the same entity is represented at different levels of abstraction), schematic discrepancy (data in one database correspond to schema elements in another). For each incompatibility category, Sheth and Kashyap listed possible concrete conflicts. Parent and Spaccapietra [2] listed the most relevant issues and the approaches to tackle data interoperability problems when integrating databases. They distinguished seven categories [2]: heterogeneity conflicts, generalization/specialization conflicts (different generalization/specialization hierarchies and different classification abstractions), description conflicts (types have different properties and/or their properties are described differently [2]), structural conflicts (different structures of related types), fragmentation conflicts (the same object is depicted by decomposition into different elements [2]), metadata conflicts, and data conflicts (data instances have different values for the same properties).

Park and Ram [3] conclude that semantic conflicts among databases can occur at two levels: data and schema. Data-level conflicts include data-value conflicts (the data value has different meaning in different databases), data representation conflicts (such as different representations of date and time), data-unit conflicts (different units are used in different databases), and data precision conflicts. All data-level conflicts can occur at the attribute level or at the entity level. Schema-level conflicts include [3]: naming conflicts, entity-identifier problems, schema-isomorphisms, conflicts of generalization, aggregation conflicts, and schematic discrepancies. Arenas et al. [4] tackled the problem of exchanging data between different relational databases that have different schemas. They mentioned three key problems of relational and XML data exchange: how to build target solutions; how to answer queries over target solutions; and how to manipulate with schema mappings (metadata management)[4]. Rocha et al. [5] presented their framework to support migrating from relational (MySQL) to NoSQL databases (MongoDB). Their framework consists of migration and mapping module. However, their work is not focused on cloud storage systems and how to access these storage systems. It also supports only MongoDB, a document-oriented NoSQL solution.

There are several cloud APIs and frameworks that act as intermediaries between different clouds. Apache Libcloud [6] is a Python library containing a unified API that can manage cloud resources of different providers. This library is focused on infrastructure as a service and supports cloud servers, block storage, cloud object storage, load balancers, and DNS as a service. Deltacloud API [7] contains a cloud abstraction API working as a wrapper around a large number of clouds to abstract their differences. It is also focused on IaaS providers and provides drivers for Amazon, Eucalyptus, GoGrid, OpenNebula, etc. Apache jclouds [8] is an open-source library offering blob (binary content) store and compute service abstraction

for 30 IaaS providers. There are also some commercial (industrial) approaches to tackle cloud portability and interoperability. For example, Cloutex can integrate and synchronize data between Salesforce, Quickbooks Online and Magento. A similar offer, Import2.com, enables the transfer of data between cloud applications such as Salesforce, Tumblr, Nimble, Pipedrive, SugarCRM, and Zoho CRM. Import2 is currently focused on CRM, helpdesk and blog migration of cloud data. These are all commercial offers with closed source code, and adding new cloud providers to their offers can be expensive or impossible.

There are some known approaches in the current literature to tackle cloud storage interoperability and migration problems. One of the first attempts to define cloud computing ontology to achieve cloud interoperability was introduced in Youseff et al. [9]. They presented an ontology which differentiates five main layers of cloud computing (applications, software environments, software infrastructure, software kernel and hardware). Ranabahu and Sheth [10] present the usage of semantic technologies to overcome cloud vendor lock-in issues. They distinguish four types of semantics for an application: data semantics (definitions of data structures, their relationships and restrictions), logic and process semantics (the business logic of the application), non-functional semantics (e.g. access control and logging) and system semantics (deployment descriptions and dependency management of the application).

Quinton et al. [11] proposed SALOON, a software product lines-based platform to select among cloud environments the best one for a specific purpose. Their platform automates the deployment of cloud environment configurations through the generation of executable configuration scripts. Tsai et al. [12] proposed a service-oriented computing architecture for cloud interoperation. However, the implementation or use case of the proposed architecture is still missing. Bastiao Silva et al. [13] developed a unified API for delivering services using cloud resources of multiple vendors with abstract layers for cloud blob stores, cloud columnar data (e.g. Azure Table), and Publish/Subscribe mechanism (Channel API of Google App Engine and Azure Queue). However, the focus of their work is to allow different applications to interoperate using a normalized API interface, and the authors did not tackle the issue of the cloud data migration.

Vision Cloud project [14] was primarily concerned with developing the architecture of a cloud-based infrastructure to provide a scalable and flexible framework for optimized delivery of data-intensive storage services. Its main aim is to solve the data management conflicts in cloud federations and multi-clouds. Five areas of innovation in the VISION Cloud platform [14] include: data objects are enriched with a detailed metadata, data lock-in should be avoided, computations are put close to the data, efficient retrieval of objects is enabled, and strong QoS guarantees, security and compliance with international regulations are guaranteed. Data objects are grouped into containers that can have associated metadata descriptions. Researchers working on Vision Cloud project used CDMI standard to achieve interoperability among CDMI-compliant cloud storage vendors. They also introduced the on-boarding federation to move data from one cloud storage provider to an-

other. Vision Cloud's approach uses a cloud storage container as the basic unit of the federation. Vision Cloud offers a RESTful API to manage data federation. However, commercial cloud vendors are rarely CDMI-compliant at the moment. Scavuzzo et al. [15] propose a migration system for columnar NoSQL databases (Google App Engine Datastore and Windows Azure Tables) that consists of intermediate metamodel and database-specific translators. Our approach is more comprehensive, because it includes different types of cloud storage systems (columnar NoSQL, relational and object databases), platform as a service and software as a service models, and it is tested on four cloud providers (Microsoft Azure, Google App Engine, Salesforce and Zoho CRM). Additionally, when migration is not successful, our system returns found interoperability problems and their descriptions. Data type mappings of different types of cloud storage systems are also addressed in our solution.

Shirazi et al. [16] provided a formal way for migrating data between HBase as a column family database to Neo4j graph database. However, their approach is specific to the two mentioned data storage types, it is not general nor can be extended to include other types of cloud data storage. Ali et al. [17] proposed a cloud broker solution for the data migration between different software as a service providers. Their approach has several steps, which include: collection and analysis of the meta-data, development of a mapping model, solution design, implementation and testing of the solution. The main drawbacks of their approach include the need for the definition of point-to-point data mappings, and the need to implement a broker for each specific mapping, so our approach is better since we have many different cloud storage providers and it is more flexible. Bansal et al. [18] proposed a NoSQL data migration meta-model driven framework to foster data portability across cloud-based heterogeneous NoSQL databases. Our work is more comprehensive, because it includes relational and object databases, and is tested on more cloud providers' storage systems, and their approach is tested only on Microsoft Azure (Azure Tables to MongoDB or Neo4j) and includes only NoSQL databases.

Definite solutions to interoperability and portability issues of platform as a service and software as a service remain elusive due to the technical complexity and a lack of accepted standards [19]. The vendor lock-in is omnipresent in cloud offers, and many clients have postponed their investment because they fear the significant costs if they decide to migrate to another provider. The gaps in the existing literature include the lack of data portability among different types of cloud storage systems, especially between relational and NoSQL cloud data storage systems. Furthermore, there is no existing work that solves the problem of data type mappings among different types of cloud data storage systems (NoSQL, relational database, object databases). Our work deals with the mentioned problems. Our approach aims to ease the migration of data between cloud data storage systems. Once wrappers for a platform have been built, it requires little manual intervention to migrate data between various cloud data storage systems. The big advantages of our approach arise when one has to migrate many different data types and data among multiple cloud storage systems, because it is easy to incrementally add new mapping rules. The main contribution of our work is an ontology-based architecture for the

automatic cloud data migration between cloud storage systems (including relational and NoSQL storage systems). The approach is general for cloud storage systems, and we have checked it using two use cases, one for platform as a service, and one for software as a service model of cloud computing.

3 RESEARCH METHODOLOGY

The basic steps in this research include: design and implementation of use cases, development of the ontology used for semantic annotations, definition and development of semantic web services, and identification of interoperability problems among different commercial providers of platform as a service. In the first step of the research, use cases are defined. These use cases are examined to determine technical and semantic interoperability problems among APIs of different providers of cloud data storage systems and how to detect and resolve interoperability problems. The migration of data among different providers of platform as a service (Google, Microsoft, Salesforce) is the first use case used to prove the practical applicability of the proposed approach. Additionally, the concept was also applied to software as a service model of cloud computing to perform one-shot data migration from Zoho CRM to Salesforce CRM. In the future, similar use cases can be defined. Many papers and research projects (e.g. FP7 or Horizon2020) in the computer science field use a similar research method, i.e. begin with a definition of use cases to explain requirements and to later test the approach.

The second step of this research is the development of the ontology for resources and operations. The aim is to clearly describe and categorize the existing functionalities, features and specificities of commercial platform as a service offers. Additionally, the ontology supports data mappings among the heterogeneous APIs and various data storage systems. The offerings of platform as a service often use proprietary and non-standard databases (relational and non-relational). Representing these data models by means of ontology can provide a common layer for information exchange.

The PaaS ontology developed in the previous step is used to create semantic web services that represent remote functions (APIs) of platform as a service offers. Every operation from the cloud vendor's API will be semantically described using a web application developed for this purpose. The aim of these semantic web services is to simplify determination and resolution to interoperability problems among the existing commercial vendors. All cloud providers offer APIs to access and manage their cloud storage systems, and to integrate or migrate data from multiple providers. We can use these services that are further semantically annotated in our approach to enable automatic or semi-automatic data migration. In the end, we try to determine the existing interoperability problems among the selected commercial cloud solutions by comparing their associated semantic web services. For this purpose, the AI planning methods were used. AI planning was chosen because it is one of the most promising techniques to solve the problem of web service composition.

4 CHOSEN PAAS OFFERS AND STORAGE SYSTEMS

Our approach was first applied to platform as a service model of cloud computing. There are many providers of platform as a service. We have chosen the following three prominent providers of platform as a service: Microsoft, Google, and Salesforce. These offers were chosen because they are currently among the leading offers on the platform as a service market with many current users. For example, in the magic quadrant for enterprise application platform as a service published in January 2014, Gartner [20] listed Microsoft and Salesforce.com as the only two market leaders, and Google as the only market challenger among the total of 18 reviewed commercial PaaS providers. Furthermore, the mentioned PaaS offers support different types of data storage systems that can possibly identify more data interoperability problems in comparison to moving data only among the cloud storage systems of the same types.

On the Force.com platform, data objects are called custom objects (similar to tables in databases). In Salesforce [21], an organization represents a database with built-in user identity and security. Objects are similar to tables in relational databases and they contain fields and records. Objects are related to other objects by using relationship fields instead of primary and foreign keys. There are two types of objects: standard objects (predefined, created automatically by Salesforce) and custom objects (objects that you create in your organization). Each custom object has some predefined, standard fields. Every custom object's name on Salesforce must finish with the postfix `_c` (e.g. `Customer_c`).

Next, Google App Engine has three options for data storage: App Engine Datastore, Google Cloud SQL and Google Cloud Storage. The App Engine Datastore [22] is a schema-less object datastore. The datastore holds data objects named entities; each entity has one or more properties of one of the supported data types; and each entity is identified by its kind and key. Google Cloud SQL [23] enables the usage of relational MySQL databases in Google's cloud. The Google Cloud Storage is an experimental service that provides storage for big objects and files (up to terabytes in size). The first option (App Engine Datastore) was selected because it is the only free option. Furthermore, it is a good example of key-value cloud storage. Datastore objects can be created programmatically by means of Java object classes, servlets, HTML and JavaScript.

There are three main storage offerings on the Azure platform [24]: Local Storage, Windows Azure Storage, and SQL Database. Local Storage provides temporary storage for a running application and it represents a directory that can be used to store files. Windows Azure Storage consists of blobs (storage of unstructured binary data), tables (a schema-less collection of rows such as entities, each of which can contain up to 255 properties) and queues (storage for passing messages between applications) that are accessible by multiple applications. SQL Database is based on SQL Server technology and provides a relational database for the Azure platform. For the purpose of these use cases, SQL Database option was chosen. To be better at detecting interoperability problems among different types of PaaS storage, this

relational storage option was chosen, because in the first two providers different types of PaaS storage were selected. More various interoperability problems can be detected if different types of PaaS storage were chosen, instead of choosing the same or similar storage types (such as key-value datastore, relational database-like storage, or object storage) for each PaaS provider. A database can be created by means of Microsoft Azure management portal (<https://windows.azure.com>). It can also be created programmatically.

Most API data operations deal with one data container (table, entity, or custom object), so if users want to migrate all data, they must first learn how to get names or identifiers of data containers. All three chosen PaaS providers enable CSV export, and these files can be used to obtain the required names or identifiers. The obtained basic structure can be used to call remote API functions to retrieve detailed information about the structure of data and data types from cloud storage systems.

5 TRANSFORMATION OF DATA STRUCTURES AND DATA TO ONTOLOGY

Data structures and data of each platform as a service's storage will be represented as the unified data model ontology, so OWL will be used as an intermediate format to migrate data between PaaS vendors. We have chosen OWL, because we also use it for semantic web services. The mappings could be implemented in any other format. This architecture is inspired by a direct mapping approach [25] proposed by the RDB2RDF Working Group. The transformation of data structures from cloud storage to ontologies is based on mapping rules that specify how to map PaaS data constructs to the ontological models. Astrova et al. [26] proposed an approach to automatic transformation of relational databases to ontologies. They listed the mapping rules [26] which inspired the rules presented later in this paper. Inevitably, some of the semantics captured in a relational database will be lost when transforming the relational database to the ontology [26], the same situation will certainly also happen when dealing with PaaS storage systems.

Due to many differences among cloud storage types supported by major commercial providers of platform as a service, the basic transformation rules were defined to build data model ontology's classes, data properties and instances (see Table 1). The mappings in the other direction (from OWL ontology to cloud storage) could also be defined, so representing these data models by means of the OWL ontology can provide a common layer for information exchange. The web services for reading and writing OWL data ontologies were created using the above specified transformation rules and the Apache Jena framework [27] for building Semantic Web applications in Java programming language. Jena provides an API to work with OWL and RDFS files and a rule-based reasoning inference engine.

Azure SQL	GAE Datastore	Salesforce	OWL
table	entity kind	object	OWL class
column	property	field	data type property
row	entity	record	instance
primary key	identifier from an entity key	identifier of an object (recognized as a field of Salesforce's ID data type)	data property identifier in an instance
foreign key	relationship between two entities	relationship between two objects (recognized as a field of Salesforce's reference data type)	object property hasLinkToObject with the appropriate domain and range in an instance

Table 1. Basic transformation rules to and from an intermediate OWL file

6 PAAS ONTOLOGY AND DATA TYPE MAPPINGS

Our architecture for PaaS data migration is using OWL as data intermediate format. Additionally, we have used PaaS ontology to define mappings between data types of different PaaS providers and to list all relevant providers' API operations for manipulating and management of underlying PaaS data. For the purpose of the development of PaaS ontology, the Ontology Development 101 [28] methodology was selected. This methodology was chosen among others, because it is the simplest and it is really focused on the results, i.e. building the first ontology version very fast and then refining it according to requirements. Web Ontology Language (OWL) was chosen because it has the needed expressive power and is the most widely used language for ontologies in the papers in the field of computer science and research projects related to this field of study. The aim of the ontology is to describe clearly and to categorize the existing functionalities and features of commercial providers of platform as a service. Our ontology is publicly available at <https://github.com/dandrocec/PaaSInterop/tree/master/PaaSOntology5>.

Initially, the concepts in this ontology were derived from the existing cloud ontologies (mostly from mOSAIC project), PIM4Cloud [29] metamodel from REMICS project, OASIS Reference Ontology for Semantic Service Oriented Architecture [30], relevant related works from literature [9], remote cloud functions specified in the API documentation of the most prominent commercial providers of platform as a service (Google App Engine, Microsoft Azure, Salesforce), standards for Semantic Web services such as OWL-S and WSMO, relevant cloud computing standards (OCCI, TOSCA, CDMI), and using personal experience in building applications for platform as a service. A total of 146 classes were defined that are organized in 17 top level classes (Table 2). The ontology is described in more detail in our previous work [31].

For this paper, the most important part of the PaaS ontology is its application for data type mappings which will be explained in the next chapter.

Each platform as a service provider supports its own set of data types. Data types differ in their name, value space, permitted range of values, precision of data, etc. Data types from the three chosen PaaS storage types (Google App Engine Datastore [32], Microsoft Azure SQL Database [33], Salesforce [34]) are mapped to XSD (because OWL uses Schema Data Types – [35] and [36]), more specifically to an OWL data property’s range of data model ontology. OWL (XSD) data types are chosen as a baseline system.

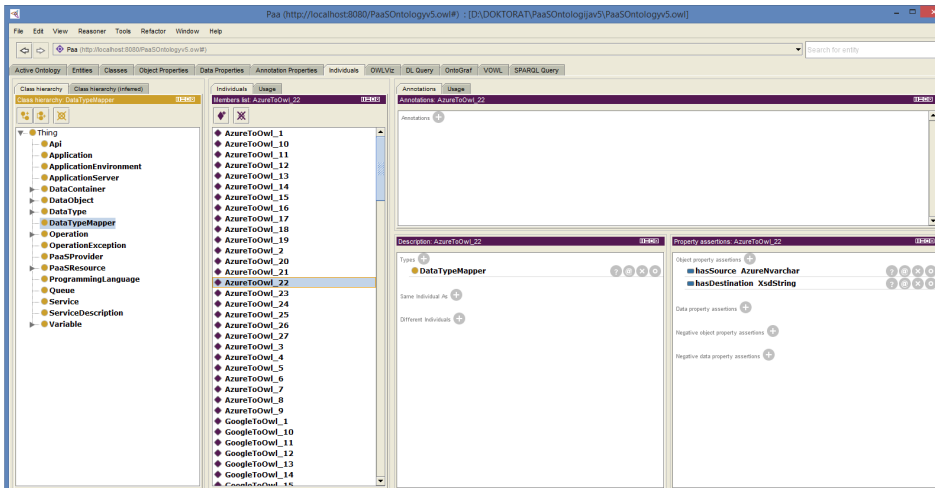


Figure 1. Instances of *DataTypeMapper* class for mapping different cloud providers’ data types

Data type mappings were implemented by means of instances of the ontology. Two classes deal with data types mappings between different PaaS storage systems: *DataType* and *DataTypeMapper*. The subclasses of the *DataType* class are OWL data types and data types of each platform as a service storage model (*AzureDataType*, *GoogleDataType*, *OWLDataType*, and *SalesforceDataType*). Each data type is represented by an individual (instance) of the associated class. As an illustration, *XsdDate* is an instance of the OWL class *OWLDataType* and it represents the *xsd:date* type. The second important class is *DataTypeMapper*. This class has two object properties (*hasSource* and *hasDestination*), and instances of this class are actually data type mappings (see Figure 1). For instance, *SalesforceToOwl_2* is an instance of the *DataTypeMapper* *hasSource* *SalesforceBoolean* and *hasDestination* *XsdBoolean*, so it shows that the Salesforce’s Boolean data type is mapped to the OWL Boolean data type.

Web services were created to handle these mappings automatically by reading the OWL ontology and performing the needed mappings and transformations.

Top Class	Description
Api	It represents vendors' Application Programming Interfaces (APIs).
Application	It contains all instances of applications that are deployed to a PaaS offer and run in the ApplicationEnvironment.
ApplicationEnvironment	PaaS application environment such as Google App Engine Java runtime environment.
ApplicationServer	Application server dedicated to efficient execution of cloud applications on vendor's servers.
DataContainer	This class is an abstraction of containers of data objects, e.g. tables, entities, objects, files directories.
DataObject	This class includes instances of data objects of various storage options such as NoSQL, relational database, object database and cloud file systems.
DataType	Data types in cloud storage systems or cloud services.
DataTypeMapper	Its instances are used for data type mappings between different storage systems of different PaaS vendors.
Operation	It represents all instances of remote operations defined in various vendors' APIs.
OperationException	It includes all instances of possible exceptions thrown by remote operations defined in vendors' APIs.
PaaSProvider	It includes instances of commercial vendors who offer platform as a service.
PaaSResource	A generic resource provided by a PaaS vendor.
Programming-Language	It contains instances of computer languages used for developing applications in vendor's environment.
Queue	It covers all instances of FIFO queues supported by commercial providers of platform as a service.
Service	It includes all kinds of services provided by commercial vendors of platform as a service.
ServiceDescription	A description of the functionality provided by service
Variable	Its subclasses include input, output, and results of APIs' web services.

Table 2. Top level classes of our ontology

For now, *DataTypeMapper* has approximately 150 instances (mappings). If some mappings are not correct, they can be fixed in the PaaS ontology and data type conversion will work. If another platform as a service provider is added, another subclass of *DataType* must be added, as well as instances for each data type of the new PaaS storage, and mapping instances from and to OWL data types must be created. Web services for data mapping to deal with the new storage provider also need to be slightly upgraded. This enables great flexibility regarding the mapping of data types supported by different PaaS providers. Some data types have unsupported mappings (for example, for Salesforce's *anyType* we can not find anything similar in OWL). In these cases, data migration will stop and error will be shown to the user suggesting that there is an interoperability problem connected to data types of different PaaS storage systems.

Ontology Development 101 methodology does not have an explicit evaluation step and it lacks evaluation procedure and recommendations, but evaluating the ontologies is useful to refine the ontologies and see whether they can be used in applications as expected. The ontology was evaluated by four human experts working in the field of cloud computing interoperability and related science projects. Their feedback was used to refine the ontology. After their initial feedback, the ontologies were revised and improved, and contact was kept (by email) with the experts who offered more comments on the newer versions of the ontologies. Several pitfalls were found by four experts and the ontology was improved.

7 SEMANTIC PAAS WEB SERVICES

All cloud providers enable developers to access their cloud storage systems using their application programming interfaces (APIs). They also provide custom tools or scripts, but these are specific for a certain cloud platform and not usable on another platform. If we want to integrate different cloud storage systems, the best solution is to use APIs (RESTful or SOAP service designed to be used exactly for this purpose, i.e. to integrate custom application or systems with cloud storage systems). Current web services provide only syntactical descriptions, so web service integration must be done manually. Semantic web services are the integration of Semantic Web and service-oriented architecture implemented in the form of web services. Semantic web services are aimed at an automated solution to the following problems: description, publishing, discovery, mediation, monitoring and composition of services. For this reasons, we have decided to use semantic web services. Web services that encapsulate remote API operations of three commercial providers (Google, Microsoft, and Salesforce) were developed to access these services in a unique way (providers offer their remote APIs in different forms - REST, SOAP or programming language libraries). These services directly call remote vendors' APIs. Some composite services (that call more than one cloud API operation and perform some additional tasks) were also developed (e.g., some of the services used for the data migration between PaaS storage types). Web services and all other parts of the authors' pro-

totype were implemented in Java. The source code of the tool is publicly available at <https://github.com/dandrocec/PaaSInterop>.

SAWSDL (W3C's Semantic Annotations for WSDL) [37] lightweight annotation was used to define semantic web services. SAWSDL was chosen due to its simplicity, its rich ontology-based data mediation mechanism for mapping inputs to outputs of web services and tool availability. The SOWER tool developed as part of the SOA4All FP7 project [38] was used to facilitate the manual annotation of WSDL service descriptions with the semantic information [38]. The web services that invoke API operations of the providers of platform as a service were developed, and each particular API operation with a term defined in this ontology of platform as a service can now be annotated. For instance, the Azure's *createTable* web service operation can be referenced to *CreateDataOperation* class of the OWL ontology.

8 AUTOMATED PAAS SERVICE COMPOSITION

As we have already mentioned in the previous section, what all cloud providers have in common is that they provide APIs for cloud storage management. To move data among different cloud providers, we have chosen to use semantically annotated web services representing cloud APIs. The next step is to automatically or semi-automatically compose semantic web services to retrieve data from one cloud storage, perform data type mappings, and store data into another cloud storage.

In the current literature, the automated composition of web services was performed using numerous methods, such as: Event Calculus, Petri Nets, Colored Petri Nets, Linear Logic theorem proving, AI planning, logic programming, Markov process, States Machines, etc. AI planning is one of the most promising techniques to solve the problem of the automated web service composition, and was chosen for the mentioned task in this work. We also wanted to determine the existing data interoperability problems among the selected commercial cloud solutions by comparing their associated Semantic Web services to find out which of these problems can be solved using the currently available API operations of commercial vendors. For this purpose, the AI planning methods were used and are described in more detail in Section 9.

A JSHOP2 planner was used for the AI planning process. The JSHOP2 planner was chosen because it is implemented in Java and can be easily incorporated into other parts of the prototype system that was developed using Java technologies, and it was used in the past for similar purposes, i.e. the composition of web services in various contexts. JSHOP2 is a Java version of Simple Hierarchical Ordered Planner (SHOP). It is used to generate sequential plans. It is based on ordered task decomposition where tasks are planned in the same order as later in the execution [39]. The objective of JSHOP2 and other HTN planners is to accomplish a set of tasks where each task can be decomposed, until primitive tasks [40]

are reached. The inputs of JSHOP2 are a planning domain and a planning problem. In JSHOP2, primitive tasks are called operators whose name must begin with an exclamation mark. The body of an operator consists of a precondition (must be satisfied to execute the action), a delete list (a set of properties that will be removed), and an add list (a set of properties that will be added) [39]. Solving a planning problem in JSHOP2 is done in three steps: the domain description file is compiled into Java code, the problem descriptions are converted into Java class, and the second Java class should be executed to initiate the planning process and inspect the planning results. These three steps were incorporated into the authors' prototype.

The problem description file is composed of logical atoms showing the initial state and a task list [39]. The task list and the initial state are created on the fly, when the user executes some interoperability actions using the client web application. Based on the choices of the user, the tasks that need to be completed are generated and saved in JSHOP2 problem description file. For example, if the user selects the data migration between Salesforce's and Google App Engine's PaaS storage, it looks like this:

```
((migrateData SalesForce GoogleAppEngine))
```

Java class was developed that handles this and writes the appropriate content to file using standard Java I/O and file classes and methods. In this case, the task lists are simply methods defined in the domain description file described later. The initial state (a set of logical atoms) is also created programmatically based on SAWSDL files and the PaaS ontology. A SAWSDL parser was developed in Java by using an EasyWSDL open-source library and its extension EasySAWSDL. The class for parsing the OWL ontology was implemented by using the Apache Jena library. Based on these two files, various logical atoms could be generated to represent the initial state. The most important logical atoms regarding PaaS data migration, together with their definition and the description of their creation are systematically listed in Table 3.

The domain description file is defined manually. A method called *migrateData* shows which operators should be called to migrate data from one PaaS storage to another:

```
(:method (migrateData ?from ?to)
  ()
  ((!checkDataTypeMappings ?from)
  (!createDataModelOntology ?from)
  (!createDataElementsFromOntology ?to))
)
```

First, the existence of the needed data type mappings are checked, then data model ontology is created, and finally data is migrated to target PaaS storage. The operator *!checkDataTypeMappings* includes preconditions that check whether all data types from data to be migrated have the appropriate data type mappings defined

Logical Atom (with Example)	Description and Generating Method
hasApiOperation (has-ApiOperation Azure CreateDataOperation)	<ul style="list-style-type: none"> - it claims that a specific PaaS API has a specific API operation - cross-PaaS operation names are specified in the PaaS ontology, and services are annotated using SAWSDL - it is generated based on SAWSDL files - if Java class parsing SAWSDL finds a semantic annotation by means of sawsdl:modelReference on a service operation, it then generates hasApiOperation logical atom in JSHOP2 problem description file
typeInCurrentData (typeInCurrentData salesforcecurrency)	<ul style="list-style-type: none"> - it shows which type is present in storage system of the chosen PaaS offers - present data types in PaaS storage are obtained calling remote APIs of PaaS providers
dataTypeMappingExists (dataTypeMappingExists azuresmallmoney xsdecimal)	<ul style="list-style-type: none"> - it specifies the data type mapping between data types of different PaaS storage systems - the PaaS ontology is parsed to obtain all instances of DataTypeMapper OWL class that represent data type mappings between PaaS storage systems

Table 3. Logical atoms in the initial state

in the JSHOP2 problem file. The operators *!createDataModelOntology* and *!createDataElementsFromOntology* include preconditions to check whether the selected PaaS providers have the appropriate operations to execute the migration of data from source to target PaaS offer.

9 ARCHITECTURE FOR CLOUD DATA MIGRATION

After the domain and problem description files were successfully created, these definitions are forwarded to a component in the prototype which invokes JSHOP2 planner to get a plan if it exists. The domain and problem descriptions are dynamically compiled into Java code, and the resulting Java files are redeployed to Glassfish server. AI planning process can then be started. If JSHOP2 planner finds a plan, this plan is printed on the client web application, and an option to execute the plan (to invoke relevant web services) is given to the user. If the planner finds the appropriate plan, then no interoperability problems were found at this stage. The plan given by JSHOP2 is parsed to retrieve adequate web services from SAWSDL files that need to be executed. Apache CXF framework [27] was used to dynamically invoke the appropriate web services.

If there is no suitable plan returned by the JSHOP2 planner, the client web application displays the error message. In this case, some interoperability problems exist and the cause of the failure needs to be determined. In the existing literature,

there are few approaches to tackle gaps in the planning domains. Our approach is similar to the one proposed by Goebelbecker and Keller [41]. They proposed to change the initial state, when no plan can be found, with the aim to find reasons why some tasks cannot be solved. They named this change “an excuse”; they created a method for finding the candidates for the excuse where they replan with new initial states to find out whether they found the cause why the plan is not found. Our approach differs from the one proposed by Goebelbecker and Keller [41], because it does not need replanning that is an expensive and time-consuming task. This algorithm consists of four main steps:

Find problematic operator or method – The domain description file of JSHOP2 is simple. The data migration action is represented by a method that describes a set of operators that need to be executed. There is only one way to successfully get a plan – all operators defined in a particular JSHOP2 method must be successfully finished. JSHOP2 supports a function to programmatically inspect every step in the planning process. This function was used to get the list of all the steps of the planner. This list of steps was programmatically parsed in Java, and an operator or a method were found where the first BACKTRACKING action occurs. This action occurs when some preconditions of the operator or the method are not satisfied, and then the JSHOP2 planner goes back up in the tree to try to find another path to the solution. In this case, the first BACKTRACKING action in a plan step represents the problematic atom (problematic method or operator where interoperability problem had occurred).

Parse concrete preconditions – The next step is to parse preconditions of the problematic operator or method. JSHOP2 domain file is directly parsed to get all the relevant preconditions. A list of preconditions was created, and in the next step it was determined which of the preconditions is the cause of the problem.

Check whether the preconditions are satisfied in the end state – The end state (the last state after the AI planner fails to get a plan) is parsed to compare which of the preconditions are not satisfied in this state, and one or more preconditions are listed as indicators of interoperability problems.

List interoperability problems – Each logical atom that is used in states and preconditions has some meaning (for example, *hasApiOperation* describes that one PaaS offer has a particular API operation annotated with the cross-PaaS concept from the ontology). Using this meaning, error messages were programmatically created to explain the found interoperability problem in the client web application to a user. For example, if the problematic precondition contains *hasApiOperation*, then there is a missing API operation problem in the concerned PaaS offer.

Let us now examine the proposed architecture for data migration. The user starts data migration using the client web application (Figure 2). The CSV files

are parsed and data, data structures and types are retrieved by calling remote API functions. The data model ontology is created, and data is ready for migration to another PaaS provider. There are also internal web services that can read the data ontology, perform mappings, create data and data structures and move them into target PaaS storage. The mentioned internal web services parse SAWSDL files and the PaaS ontology, and use the techniques described in the previous sections. The AI planning component deals with the AI planning files and executes the required semantic web services. If the user chooses only one data container (table, entity, or Salesforce’s custom object) the migration flow is the same, only data container name is forwarded as a filter to include only the chosen container and disregard the other remaining data during migration.

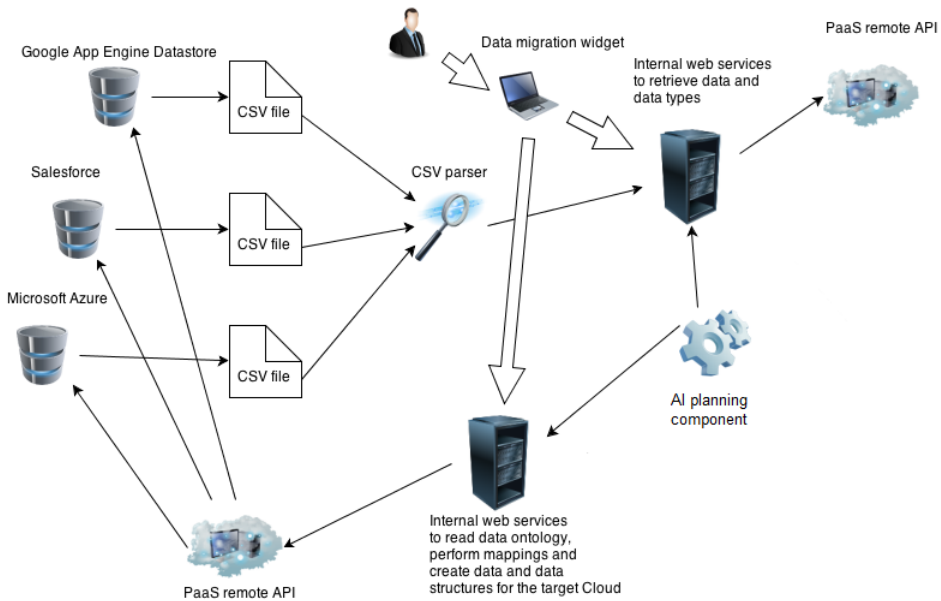


Figure 2. Architecture for data migration between PaaS providers

10 VALIDATION AND ASSESSMENT

10.1 Migration of PaaS Storage Data

The validation of the data migration architecture was done by migrating a more complex set of data and manually checking all of the migrated data elements. For this purpose, sample data of an open-source content management system (CMS) Vosao [42] was chosen. Vosao CMS uses Google App Engine Datastore and initially

consists of 19 entity kinds, 186 entites, and 203 properties (see Table 4 for more details).

Entity Kind	Number of Entities	Number of Properties
ConfigEntity	1	23
ContentEntity	23	9
ContentPermissionEntity	1	10
FieldEntity	3	15
FileChunkEntity	43	8
FileEntity	43	11
FolderEntity	15	8
FolderPermissionEntity	3	8
FormConfigEntity	1	1
FormEntity	1	14
GroupEntity	1	6
LanguageEntity	1	7
PageEntity	26	30
PageTagEntity	13	7
StructureEntity	1	7
StructureTemplateEntity	2	11
TagEntity	5	9
TemplateEntity	2	8
UserEntity	1	11

Table 4. Quantitative information about sample Vosao’s data

Its data were then migrated to the other two chosen PaaS offers (Microsoft Azure and Salesforce) to check whether the developed prototype migration tool works smoothly when there is a real cloud application with more data containers and data objects. The implementation of the migration consists of eight main steps:

Prepare migration – We developed two composite web services to transform the source data into an intermediate OWL file, and vice versa: from the intermediate OWL data ontology to concrete data in target PaaS offer’s storage. The mentioned composite web services invoke remote PaaS APIs and call our developed services that parse CVS, SAWSDL and JSHOP2 files. Next, they were semantically annotated (classes *CreateDataModelOntologyOperation* and *CreateDataElementsFromOntologyOperation* from the PaaS ontology describe this functionality) using SAWSDL. Finally, a JSHOP2 domain description file is defined manually (see Section 7 for more details).

Export CSV files from PaaS provider – Our prototype first needs to identify the names of entities used by Vosao, because most API data operations deal with only one data container, and entity name is a required input parameter for these operations. Google provides the bulk loader tool in Python SDK that provides

functionalities to download CSV data from a specific application's Google App Engine Datastore (an instance of Vosao CMS application deployed on Google App Engine in our example). Vosao's data consist of 19 entities, so we get 19 separate CSV files, where each file represents one entity.

User chooses action, source, and target of the migration – We start the data migration using our developed client web application. We have an option to choose to port all data, or only one chosen data element (e.g., a specific entity of GAE Datastore), and we select source and target PaaS offers. In our case, we selected Vosao instance as source and two other PaaS offers (our test instances of Salesforce and Microsoft Azure) as targets. For now, our web application enables migration to one target PaaS offer at a given time, so we need to repeat this step twice to migrate Vosao data to the two mentioned target platforms.

Generate problem description file and execute AI plan – JSHOP2 problem description file is generated based on the user's choices, SAWSDL files, and the PaaS ontology (see Section 7). For example, when we choose to move Vosao's data from Google App Engine to Salesforce, the generated goal is:

```
((migrateData GoogleAppEngine SalesForce))
```

Next, the AI planner is executed to see whether there are interoperability problems (e.g. missing operations, missing or impossible data type mappings). If there is no suitable plan returned by the JSHOP2 planner, the client web application displays the error message. In this case, some interoperability problems exist and the cause of the failure needs to be determined. The detailed algorithm is presented in Section 8. In our migration use case there were no identified problems, so we could proceed to the next step described below.

Dynamic invocation of web services – The JSHOP2 and SAWSDL files are parsed to execute adequate web services using Apache CXF framework. For instance, the web services annotated with *CreateDataModelOntologyOperation* in SAWSDL representing Google App Engine's operations and *CreateDataElementsFromOntologyOperation* in SAWSDL file representing Salesforce's operation are executed. Two transformations were performed:

- a) *Transformation to unified data model ontology* – On the server side, we implemented web services in Java that parse CSV files, call the appropriate data manipulation remote provider's API to extract details about data (attributes, identifiers, relationship between different data elements), perform data type mappings defined in the PaaS ontology, and use Jena framework to construct the data model ontology according to the rules presented in Section 4. For each Vosao's data store entity, attributes, identifiers, data types, and the number of instances were checked and the conclusion was that the transformation was successful. The data ontology contained all the

entities and data from Vosao's data stored in Google App Engine Datastore.

- b) *Transformation to target data model* – During this transformation, the intermediate OWL data file is transformed to data elements in the target PaaS storage. The verification of migration of Vosao's data to Salesforce and Google App Engine was done manually in Excel. All data containers, their names, the names and the number of their attributes, and the number of records were listed there. Additionally, all the data for randomly chosen entities were also checked. Some errors were initially found and bugs in the prototype were fixed until the migration was properly done. In Salesforce, custom objects must have `_c` postfix, so it is necessary to add these to the names of entities stored in Google App Engine's Datastore. The names of custom fields must also end with `_c` string. In Excel, the number of properties (of entities from GAE Datastore) and custom fields of each custom objects were compared, and the numbers were identical. Salesforce automatically creates an ID standard field for each object, so the `identifier_c` custom field was created to save the Google's identifier. When creating a new object, Salesforce always adds some obligatory standard fields (Name, CreatedBy, LastModifiedBy, and Owner). Then, the data record numbers in Google's and Salesforce's platforms were compared, and identical values were obtained. ApexDataLoader tool was used to get data records from Salesforce. Next, the data migrated from Google App Engine to Microsoft Azure was checked in the similar way. The number of properties (of entities from GAE Datastore) and columns of tables created in Microsoft Azure were compared, and the numbers were identical. Then, the data record numbers in Google's and Azure's platforms were compared, and identical values were obtained. Microsoft SQL Server Management Studio tool was used to inspect the data migrated to the Microsoft Azure instance. Finally, some entities were randomly chosen and all the data and mappings of data types in one and the other platform were checked.

Furthermore, the migrated data was taken and put again using the migration tool to a new instance of Google App Engine (`datafromazure.appspot.com` and `datafromsalesforce.appspot.com`) and then this data was compared to the original Vosao's data in its original instance of Google App Engine and its underlying datastore. The number and the names of entities, properties and identifiers were manually checked. When migrating from Salesforce, the only difference in data is the identifier, because Salesforce platform automatically assigns identifiers (ID field of each custom object). The same procedure was repeated to migrate data back from Microsoft Azure to the new instance of Google App Engine. The video of the sample PaaS data migration using our developed tool and ontology is available at <https://www.youtube.com/watch?v=tmwoV6XgIhs>.

10.2 Migration of SaaS Storage Data

The same approach was used to migrate sample data of Zoho CRM to Salesforce. We believe that our approach for migrating cloud storage data is general, and we have used the mentioned sample use case to further analyse, test and validate our approach. Zoho CRM [43] provides REST API to access its data. It is a cloud based customer relationship management (CRM) software. We have created a free instance and filled it with initial data. The sample Zoho's data contains 20 standard objects and a total of 553 columns. Each object has been filled with a couple of rows. Zoho's API for the free version supports method *getRecords* for the following standard objects: *Account*, *Campaign*, *Contact*, *Lead*, *Potential*, and *Task*. We have created new mappings to the data ontology from Zoho CRM (similar to the one described in Section 5 for the other three cloud providers) and a new web service that calls Zoho CRM REST API to get the records. Then we have semantically annotated the mentioned web service and added the needed data type mappings in the ontology. The new cloud provider was also added to the AI planning problem. We started the migration, and it was successfully finished, besides that custom objects were created in Salesforce. Salesforce CRM offer has standard objects similar to Zoho's account, campaign, contact, lead, potential and task objects, and to use data directly in Salesforce's cloud application, the migration to standard objects needs to be performed. To successfully migrate data at software as a service level, the standard objects of one CRM offer (e.g. Zoho CRM) need to be translated into standard objects of another CRM (e.g. Salesforce) cloud offer. For this purpose, the data migration architecture described in Section 9 of this work was changed only in the step where intermediate data ontology is created. An adapter Java class that reads the XML file representing standard objects mappings of different SaaS offers (in our test case two different cloud CRM offers) and changes intermediate data ontology representing cloud storage data was developed. Apache Jena was used to rename or delete names of standard data objects (OWL classes in the ontology) and objects' attributes (data properties in the intermediate ontology). The migration was finished successfully, and with this example we have shown that our approach with minor modifications (the addition of simple intermediate data ontology transformation according to mappings of standard objects and their attributes) can also be used for the data migration at SaaS level. The samples of migrated data and generated intermediate OWL data ontologies are available at <https://github.com/dandrocec/PaaSInterop/tree/master/migratedData>.

11 CONCLUSION

There are many data migration problems among cloud providers. To minimize the possible data migration problems in the cloud domain, users should carefully choose a cloud offer, underlying cloud storage systems, and features. It is best to avoid using vendors' specific features that are not supported in any other cloud

offer. For example, most data types problems can be avoided if the established variants of data types (for example, integer, string, etc.) were used and if the usage of new or innovative data types (e.g., Salesforce's anyType, calculated, or DataCategoryGroupReference data type) that cannot be mapped to data types of different cloud storage is avoided. The more users use advanced and innovative functionalities that are vendor specific, the more difficult it will be for migration and interoperability to occur.

In this paper, we proposed a flexible data migration architecture. Our work aims to develop a configurable method for batch migration of data from one cloud data store to another provider's platform. The chosen approach rests on a standard intermediate representation of data and meta-data (in OWL), along with custom-built wrappers for the data manipulation operations available on the source and target platforms. This architecture answers two research questions: How to leverage automatic data migration between cloud solutions? How to provide cross cloud data type mappings for different cloud storage systems used by various PaaS offers? Automatic data migration between cloud offers was done by using OWL as an intermediate format, PaaS ontology, semantic web services implemented in SAWSDL, and AI planner JSHOP2. The validation of the data migration architecture was done by migrating a more complex set of PaaS and SaaS data (concretely, data of the open-source content management system Vosao and data of Zoho CRM) and manually checking all of the migrated data elements. Data type mappings were implemented by means of the ontology, where instances of data type mapping classes were used for this purpose. Our approach enables one-shot migration of cloud data between different types of cloud storage systems (e.g. NoSQL and relational cloud databases). It uses a flexible approach to avoid point-to-point mappings. The main novelty of the paper is a specific application domain (ontology-based migration of data between different cloud offers) and the implementations of mappings among different types of cloud data storage types (NoSQL, relational database, object database). We also provide a comprehensive description of the design and an implementation of the automated cloud data migration solution using state of the art Semantic Web and AI planning techniques. The identified cross-PaaS concepts of the defined PaaS storage data types and their mappings improve the understanding of PaaS and SaaS models in more detail than any other models and ontologies in the existing literature. These concepts also enable semantic annotations and help solve known interoperability problems.

There are several limitations of this work that need to be considered. The AI planning component of this system does not take into consideration the non-determinism of the domain (as an example, some of the remote API operations could be unavailable at specific time; the output of one web service could differ from the expected one, etc.). For this purpose, a contingent planner could be used for planning under uncertainty. Four prominent commercial cloud offers (Google App Engine, Salesforce, Zoho CRM, and Microsoft Azure) were used in this work, and it would be certainly beneficial to include other providers as well.

Some possible future research topics could arise by solving the mentioned limitations. In addition to the JSHOP2 planner that is used in this approach, our architecture could be upgraded to use some contingent planners to address the non-determinism of the domain. The presented PaaS ontology can be extended including the other cloud providers. The ontology is designed to be easily extended with additional API operations, data types and mappings of data types. Our approach could be integrated with TOSCA and/or other existing multi-cloud orchestration systems, e.g. to migrate web applications together with their data. TOSCA aims to leverage the portability of application layer services between different clouds. Generally, the automatic migration of cloud data is a very complex research and practical problem, and we hope that our paper will be a solid foundation for future research in this field.

Acknowledgement

This work has been fully supported by the Croatian Science Foundation under the project IP-2014-09-3877.

REFERENCES

- [1] SHETH, A. P.—KASHYAP, V.: So Far (Schematically) Yet So Near (Semantically). Proceedings of the IFIP WG 2.6 Database Semantics Conference on Interoperable Database Systems, North-Holland Publishing Co., 1993, pp. 283–312, doi: 10.1016/B978-0-444-89879-1.50022-1.
- [2] PARENT, C.—SPACCAPIETRA, S.: Database Integration: The Key to Data Interoperability. In: Papazoglou, M. P., Spaccapietra, S., Tari, Z. (Eds.): Advances in Object-Oriented Data Modeling, MIT Press, 2000.
- [3] PARK, J.—RAM, S.: Information Systems Interoperability: What Lies Beneath? ACM Transactions on Information Systems, Vol. 22, 2004, No. 4, pp. 595–632.
- [4] ARENAS, M.—BARCELÓ, P.—LIBKIN, L.—MURLAK, F.: Foundations of Data Exchange. Cambridge University Press, Cambridge, New York, 2014.
- [5] ROCHA, L.—VALE, F.—CIRILO, E.—BARBOSA, D.—MOURÃO, F.: A Framework for Migrating Relational Datasets to NOSQL. Procedia Computer Science, Vol. 51, pp. 2593–2602.
- [6] The Apache Software Foundation: Welcome to Apache Libcloud's Documentation! 2013, available at: <https://ci.apache.org/projects/libcloud/docs/#main>.
- [7] Apache: About Deltacloud. 2013, available at: <http://deltacloud.apache.org/about.html>.
- [8] Apache: What Is Apache jClouds? 2013, available at: <http://jclouds.incubator.apache.org/documentation/gettingstarted/what-is-jclouds/>.
- [9] YOUSEFF, L.—BUTRICO, M.—DA SILVA, D.: Toward a Unified Ontology of Cloud Computing. Grid Computing Environments Workshop (GCE'08), IEEE, 2008, pp. 1–10, doi: 10.1109/GCE.2008.4738443.

- [10] RANABAHU, A.—SHETH A.: Semantics Centric Solutions for Application and Data Portability in Cloud Computing. IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom 2010), Indianapolis, 2010, pp. 234–241, doi: 10.1109/CloudCom.2010.48.
- [11] QUINTON, C.—ROMERO, S.—DUCHIEN, L.: SALOON: A Platform for Selecting and Configuring Cloud Environments. *Software: Practice and Experience*, Vol. 46, 2016, No. 1, pp. 55–78.
- [12] TSAI, W.-T.—SUN, X.—BALASOORIYA, J.: Service-Oriented Cloud Computing Architecture. 2010 Seventh International Conference on Information Technology: New Generations (ITNG), IEEE, 2010, pp. 684–689, doi: 10.1109/ITNG.2010.214.
- [13] BASTIÃO SILVA, L. A.—COSTA, C.—OLIVEIRA, J. L.: A Common API for Delivering Services over Multi-Vendor Cloud Resources. *Journal of Systems and Software*, Vol. 86, 2013, No. 9, pp. 2309–2317, doi: 10.1016/j.jss.2013.04.037.
- [14] GOGOUVITIS, S. V.—KOUSIOURIS, G.—VAFIADIS, G.—KOLODNER, E. K.—KYRIAZIS, D.: OPTIMIS and VISION Cloud: How to Manage Data in Clouds. In: Hutchison, D., Kanade, T., Kittler, J. et al. (Eds.): Euro-Par 2011: Parallel Processing Workshops. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7155, 2012, pp. 35–44.
- [15] SCAVUZZO, M.—DI NITTO, E.—CERI, S.: Interoperable Data Migration Between NoSQL Columnar Databases. 2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations (EDOCW), Ulm, 2014, pp. 154–162, doi: 10.1109/EDOCW.2014.32.
- [16] SHIRAZI, M. N.—KUAN, H. C.—DOLATABADI, H.: Design Patterns to Enable Data Portability Between Clouds' Databases. 12th International Conference on Computational Science and Its Applications (ICCSA '12), Salvador, Brazil, IEEE, 2012, pp. 117–120, doi: 10.1109/ICCSA.2012.29.
- [17] ALI, H.—MOAWAD, R.—HOSNI, A. A. F.: A Cloud Interoperability Broker (CIB) for Data Migration in SaaS. IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA '16), Chengdu, China, IEEE, 2016, pp. 250–256, doi: 10.1109/ICCCBDA.2016.7529566.
- [18] BANSEL, A.—GONZÁLEZ-VELÉZ, H.—CHIS, A. E.: Cloud-Based NoSQL Data Migration. 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP '16), Heraklion, Greece, IEEE, 2016, pp. 224–231, doi: 10.1109/PDP.2016.111.
- [19] DI MARTINO, B.: Applications Portability and Services Interoperability among Multiple Clouds. *IEEE Cloud Computing*, Vol. 1, 2014, No. 1, pp. 74–77, doi: 10.1109/MCC.2014.1.
- [20] NATIS, Y.—PEZZINI, M.—DRIVER, M.—SMITH, D. M.—IJIMA, K.—ALTMAN, R.: Magic Quadrant for Enterprise Application Platform as a Service (Jan. 2014). Available at: <http://www.gartner.com/technology/reprints.do?id=1-1P502BX&ct=140108&st=sb>.
- [21] Salesforce: Database.com Workbook (Jun 2013). Available at: http://www.salesforce.com/us/developer/docs/workbook_database/workbook_database.pdf.

- [22] Google: Google App Engine – Storing Data (Jun 2013). Available at: <https://developers.google.com/appengine/docs/java/datastore/>.
- [23] Google: Google Cloud SQL (May 2013). Available at: <https://developers.google.com/cloud-sql/>.
- [24] FRANKS, L.: Data Storage Offerings on the Windows Azure Platform (Oct. 2010). Available at: <http://social.technet.microsoft.com/wiki/contents/articles/1674.data-storage-offerings-on-the-windows-azure-platform.aspx>.
- [25] AUER, S.—FEIGENBAUM, L.—MIRANKER, D.—FOGAROLLI, A.—SEQUEDA, J.: Use Cases and Requirements for Mapping Relational Databases to RDF. W3C working draft (Jun 2010). Available at: <http://www.w3.org/TR/rdb2rdf-ucr/>.
- [26] ASTROVA, I.—KORDA, N.—KALJA, A.: Rule-Based Transformation of SQL Relational Databases to OWL Ontologies. Proceedings of the 2nd International Conference on Metadata and Semantics Research, 2007. Available at: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.110.8189>.
- [27] Apache: Apache CXF Dynamic Clients. 2013, available at: <http://cxf.apache.org/docs/dynamic-clients.html>.
- [28] NOY, N. F.—MCGUINNESS, D. L.: Ontology Development 101: A Guide to Creating Your First Ontology. 2001, available at: <http://www-ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness.pdf>.
- [29] SOFTEAM SINTEF Tecnalia: REMICS Deliverable d4.1 PIM4cloud. Project deliverable, REMICS Consortium (Mar. 2012). Available at: http://www.remics.eu/system/files/REMICS_D4.1_V2.0_LowResolution.pdf.
- [30] NORTON, B.—KERRIGAN, M.—MOCAN, A.—CARENINI, A.—CIMPIAN, E.—HAINES, M.—SCICLUNA, J.—ZAREMBA, M.: Reference Ontology for Semantic Service Oriented Architectures. Public Review Draft 01, OASIS (Nov. 2008). Available at: <http://docs.oasis-open.org/semantic-ex/ro-soa/v1.0/pr01/see-rosoa-v1.0-pr01.pdf>.
- [31] ANDROČEC, D.—VRČEK, N.: Ontologies for Platform as Service APIs Interoperability. *Cybernetics and Information Technologies*, Vol. 16, 2016, No. 4, pp. 29–44, doi: 10.1515/cait-2016-0065.
- [32] Google: Entities, Properties, and Keys. 2013, available at: <https://cloud.google.com/appengine/docs/standard/java/datastore/entities>.
- [33] Microsoft: Data Types (Windows Azure SQL Database). 2013, available at: <http://msdn.microsoft.com/en-us/library/windowsazure/ee336233.aspx>.
- [34] Salesforce: SOAP API Developer’s Guide Version 28.0. 2013, available at: https://developer.salesforce.com/docs/atlas.enus.api.meta/api/sforce_api_quickstart_intro.htm.
- [35] BECHHOFFER, S.—VAN HARMELEN, F.—HENDLER, J.—HORROCKS, I.—MCGUINNESS, D. L.—PATEL-SCHNEIDER, P. F.—STEIN, L. A.: OWL Web Ontology Language Reference. 2004, available at: <http://www.w3.org/TR/owl-ref/>.
- [36] W3C: XML Schema Part 2: Datatypes Second Edition. 2004, available at: <http://www.w3.org/TR/xmlschema-2/>.

- [37] SHETH, A. P.—GOMADAM, K.—RANABAHU, A.: Semantics Enhanced Services: METEOR-S, SAWSDL and SA-REST. *IEEE Data Engineering Bulletin*, Vol. 31, 2008, No. 3, pp. 8–12.
- [38] SOA4All Consortium: SOWER. 2010, available at: <http://technologies.kmi.open.ac.uk/soa4all-studio/provisioning-platform/sower/>.
- [39] ILGHAMI, O.: Documentation for JSHOP2. 2006, available at: <http://sourceforge.net/projects/shop/files/JSHOP2/>.
- [40] ILGHAMI, O.—NAU, D. S.: A General Approach to Synthesize Problem-Specific Planners. Technical report CS-TR-4597 and UMIACS-TR-2004-40, 2003. Available at: <http://www.cs.umd.edu/~nau/papers/ilghami2003general.pdf>.
- [41] GOEBELBECKER, M.—KELLER, T.—EYERICH, P.—BRENNER, M.—NEBEL, B.: Coming Up with Good Excuses: What to Do When No Plan Can Be Found. *Proceedings of the 20th International Conference on Automated Planing and Scheduling (ICAPS 2010)*, 2010, pp. 81–88.
- [42] HUSTED, T.: Vosao – Google App Engine CMS. 2013, available at: <https://code.google.com/p/vosao/>.
- [43] Zoho: Zoho CRM – An Overview. 2017, <https://www.zoho.eu/crm/help/overview.html>.



Darko ANDROČEĆ is Senior Teaching Assistant at the University of Zagreb, Faculty of Organization and Informatics Varaždin, where he was awarded his Ph.D. degree in 2015. He is a member of the Department for Information System Development. Before joining the faculty, he was a computer security incident handler at CARNet (Croatian Academic and Research Network) and Java developer of banking information systems. He currently teaches computer labs in the following courses: Information Systems Development, E-Business, and Business Processes in Organizations. His main research areas are cloud computing, interoperability, semantic web, and internet of things.



Neven VRČEK is Full Professor at the Faculty of Organization and Informatics, University of Zagreb. He is the main lecturer at several courses: Software Engineering, Software Analysis and Design, E-Commerce, ERP Systems and Customer Relationship Management. He graduated at the Faculty of Electrical Engineering, University of Zagreb. He defended his master theses as well as his Ph.D. dissertation at the same faculty. His fields of interests are: strategic planning of information systems development, e-commerce and IT applications in business sector, business performance measurement, and digital signal processing. He was a leader or a team member of several scientific projects financed by the EU institutions, Croatian Ministry of Science, Education and Sports, and Croatian Science Foundation and of more than 30 commercial projects. He was appointed as Dean of the Faculty of Organization and Informatics, University of Zagreb in October 2015.

PETRI NETS AT MODELLING AND CONTROL OF DISCRETE-EVENT SYSTEMS CONTAINING NONDETERMINISM – PART 1

František ČAPKOVIČ

*Institute of Informatics
Slovak Academy of Sciences
Dúbravská cesta 9, 845 07 Bratislava, Slovakia
e-mail: Frantisek.Capkovic@savba.sk*

Abstract. Discrete-Event Systems are discrete in nature, driven by discrete events. Petri Nets are one of the mostly used tools for their modelling and control synthesis. Place/Transitions Petri Nets, Timed Petri Nets, Controlled Petri Nets are suitable when a modelled object is deterministic. When the system model contains uncontrollable/unobservable transitions and unobservable/unmeasurable places or other failures, such kinds of Petri Nets are insufficient for the purpose. In such a case Labelled Petri Nets and/or Interpreted Petri Nets have to be used. Particularities and mutual differences of individual kinds of Petri Nets are pointed out and their applicability to modelling and control of Discrete-Event Systems are described and tested.

Keywords: Analysing, control synthesis, controlled Petri nets, discrete-event systems, interpreted Petri nets, modelling, labelled Petri nets, place/transition Petri nets, timed Petri nets, uncertainty, unobservable/uncontrollable transitions, unmeasurable/unobservable places

Mathematics Subject Classification 2010: 93-C65, 93-C30

1 INTRODUCTION AND PRELIMINARIES

Discrete-Event Systems (DES) are frequently modelled by Petri Nets (PN). As to their structure, PN are [9] bipartite directed graphs with two kinds of nodes –

places $p_i \in P, i = 1, \dots, n$ (i.e. $|P| = n$) and transitions $t_j \in T, j = 1, \dots, m$ (i.e. $|T| = m$) – and two kinds of arcs – the arcs directed from places to transitions $F \subseteq P \times T$ and the arcs directed from transitions to places $G \subseteq T \times P$. Here, $P \cap T = \emptyset$ and $P \cup T \neq \emptyset$, where \emptyset symbolizes an empty set. Hence, $B = F \subseteq P \times T \cup G \subseteq T \times P$ represents the PN structure. Thus, the net structure is [29, 30] $N = \langle P, T, B \rangle$. The preset of a transition t (i.e. the set of its input places) is defined as ${}^{(p)}t = \{p | (p, t) \in B\}$, while the postset of t (i.e. the set of its output places) is defined as $t^{(p)} = \{p | (t, p) \in B\}$. On the other hand, the preset of a place p (i.e. the set of its input transitions) is defined as ${}^{(t)}p = \{t | (t, p) \in B\}$ while the postset of p (i.e. the set of its output transitions) is defined as $p^{(t)} = \{t | (p, t) \in B\}$. If for $p \in P, t \in T, \{(p, t) \in B\} \Rightarrow (t, p) \notin B$, i.e., if no self-loops occur in PN, then the net is said to be *pure*. A transition is said to be the source transition if ${}^{(p)}t = \emptyset$ and the sink transition if $t^{(p)} = \emptyset$.

Places model particular operations or activities of DES, states of which are expressed by marking – i.e. by the number of tokens $n_t \in \mathbb{Z}_{\geq 0}$ put into them. It means that marking m is a vector $m : P \rightarrow \mathbb{Z}_{\geq 0}$ where $\mathbb{Z}_{\geq 0}$ represents positive integers including 0. PN transitions model the discrete events in DES. A transition can be disabled (when it cannot be fired) or enabled (when it can be fired). Of course, the enabled transition might be, but need not to be, fired. An event modelling a failure is fired spontaneously. A set of transitions $\mathcal{T} \subseteq T$ is enabled by means of the marking m if $\forall p \in P, m(p) \geq |p^{(t)} \cap \mathcal{T}|$. It means that $m(p)$ is greater than the number of transitions in \mathcal{T} for which p is the input place or equal to this number. The occurrence of a discrete event is modelled by means of firing the corresponding enabled transition.

Theoretically, more than one transition can be fired at any instant [29, 62]. Thus two possibilities offer:

1. to fire more than one transition at any instant – so called concurrency assumption (in short the C assumption);
2. to fire only one of the transitions at any instant – so called no concurrency assumption (in short the NC assumption).

In the former case, if a set of transitions $\mathcal{T} \subseteq T$ is enabled at marking m , then \mathcal{T} may fire and the new marking m' is obtained as $m'(p) = m(p) + |{}^{(t)}p \cap \mathcal{T}| - |p^{(t)} \cap \mathcal{T}|$. It means that firing the set of transitions $\mathcal{T} \subseteq T$ causes that one token will be removed from each place $p \in {}^{(p)}t$ and one token will be added to each place $p \in t^{(p)}$. In literature about PN, the case C is used very rarely.

The latter case NC is usual in the total most of PN literature. Here, it is assumed that only a single transition is fired at any instant. Under such assumption, \mathcal{T} is a singleton set (a set having exactly one element). Here, in this paper, the NC assumption will be applied.

A firing sequence from an initial marking m_0 is a sequence of transition sets $\mathcal{U} = \{\tau_1 \tau_2 \dots \tau_k\}$ such that $m_0[\tau_1 > m_1[\tau_2 > \dots m_{k-1}[\tau_k > m_k$. The set may also be empty. The notation $m_0[\mathcal{U}$ denotes that the sequence \mathcal{U} can be fired at

m_0 and the notation $m_0[\mathcal{U} > m_k$ denotes that the firing of \mathcal{U} yields m_k . Under the NC assumption, each τ_i is a singleton set, and \mathcal{U} is a sequence of transitions. To denote that by firing of \mathcal{U} the state m_k can be reached from m_0 it can be written that $m_0[\mathcal{U} > m_k$. This is connected with the reachability of PN markings (states). In general, marking m is reachable in a net system $\langle N, m_0 \rangle$ if there exists a firing sequence \mathcal{U} such that $m_0[\mathcal{U} > m$. When the net system $\langle N, m_0 \rangle$ is given, the set of reachable markings is $R(N, m_0)$. Markings reachable from a given initial marking can be expressed by means of the reachability tree (RT) and/or the reachability graph (RG). RG arises from RT by connecting all RT nodes with the same name into one node. The incidence matrix of RG is the same as that of RT. A certain RT appertains to each initial state of PN unambiguously. Unfortunately, the opposite relation does not exist. In general, it is practically impossible to unambiguously obtain PN from a given RT.

In some PN the number of states (i.e. the reachability set) is also infinite. Consequently, RT (RG) is also infinite. To compute a substitutional finite graph, so called coverability graph (CG), a different algorithm has to be used. Each arc corresponds to a transition, but each node corresponds either to a single reachable marking or it represents an infinite set of reachable markings – in such case loops are originating in corresponding CG nodes.

PN-based models can be created either intuitively, based on a creator's empirical experience and knowledge acquired from the external observation of the behaviour of real systems (a usual approach) or by means of a systematic approach – see e.g. [6, 60, 15, 16, 17, 18, 3, 24]. However, during the model creation PN properties should be considered. The basic properties (as safeness, liveness, boundedness, reachability, reversibility, deadlock-freeness, conservativeness, etc.) are defined e.g. in basic literature [49, 47, 11] and in many other papers.

A net N is well-formed if there exists a marking m_0 of N such that $\langle N, m_0 \rangle$ is a live and bounded system.

More details about PN can be found especially in the fundamental literature [49, 47, 11] but also in many other newer sources which are specialized in extending the fundamental knowledge towards different application areas (e.g. supervision, control, etc.).

1.1 Place/Transitions Petri Nets

The later naming of above mentioned PN is Place/Transition PN (P/T PN) – see e.g. [11]. Because linear algebra can be used at PN-based modelling DES, it is useful to apply the better arranged (from the mathematical point of view) vector notation [7, 8] at modelling DES by means of P/T PN. Because from the system point of view markings correspond with state vectors of places and the transitions correspond with control variables, the marking evolution (*dynamics*) of P/T PN can

be expressed as the following restricted linear discrete state equation:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{B} \cdot \mathbf{u}_k, \quad k = 0, 1, \dots, N, \tag{1}$$

$$\mathbf{F} \cdot \mathbf{u}_k \leq \mathbf{x}_k. \tag{2}$$

Here, $\mathbf{x}_k = (\sigma_{p_1}^k, \dots, \sigma_{p_n}^k)^T$ is the state vector of the places in the step k with $\sigma_{p_i}^k \in \{0, 1, \dots, \infty\}$, $i = 1, \dots, n$; $\mathbf{u}_k = (\gamma_{t_1}^k, \dots, \gamma_{t_m}^k)^T$ is the state vector of the transitions in the step k (named as the control vector) with $\gamma_{t_j}^k \in \{0, 1\}$, $j = 1, \dots, m$, where 0 denotes the disabled transition and 1 denotes the enabled one; $\mathbf{B} = \mathbf{G}^T - \mathbf{F}$ is the structural matrix. Here, the matrices \mathbf{B} , \mathbf{G} , \mathbf{F} correspond, respectively, to the sets B , G , F . $\mathbf{F} \in \mathbb{Z}_{\geq 0}^{n \times m}$, $\mathbf{G} \in \mathbb{Z}_{\geq 0}^{m \times n}$ and $\mathbf{B} \in \mathbb{Z}^{n \times m}$, where \mathbb{Z} represents all integers including zero.

For illustration, an example of P/T PN is displayed in Figure 1. Its incidence matrices are

$$\mathbf{F} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \quad \mathbf{G}^T = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{3}$$

The initial state $\mathbf{x}_0 = (0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 1)^T$ displayed in Figure 1 yields RT given in Figure 2.

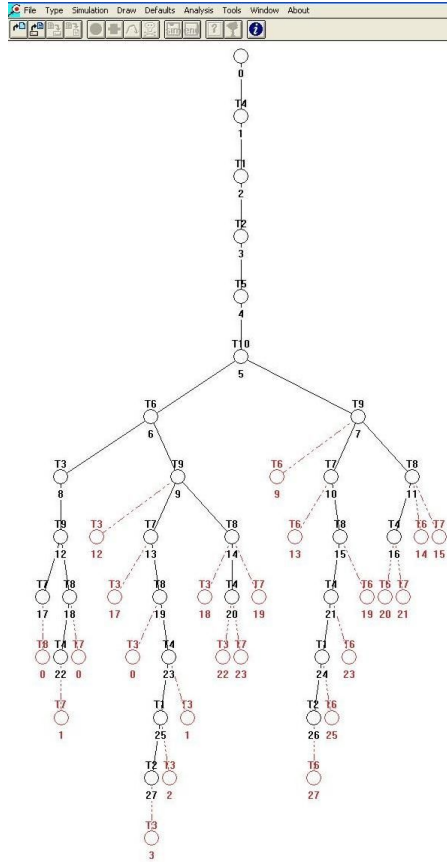


Figure 2. The RT of the P/T PN

where $\mathbf{v} = \mathbf{u}_0 + \mathbf{u}_1 + \dots + \mathbf{u}_{k-1}$ is named as the Parikh's vector. It yields information on how many times the particular transitions are fired during the evolution of P/T PN from the initial state \mathbf{x}_0 to a prescribed terminal state \mathbf{x}_k .

There exist some special kinds of P/T PN. The mostly used are the following:

- P/T PN where every place has exactly one incoming arc, and exactly one outgoing arc are named as *marked graphs* (MG). Mathematically expressed $\forall p \in P : |p^{(t)}| = |{}^{(t)}p| = 1$. No conflicts in MG occur, but a concurrency can be expressed there. In a graph interpretation, MG is a graph where each place represents an arc and each transition represents a node.
- P/T PN where every transition has exactly one incoming arc, and exactly one outgoing arc and all markings have exactly one token then are named as *state machines* (SM). Mathematically expressed $\forall t \in T : |t^{(p)}| = |{}^{(p)}t| = 1$. Concur-

rency cannot be expressed there, but conflicts appear caused by more outgoing transitions from a place (i.e. a kind of uncertainty or nondeterminism).

- P/T PN where every arc from a place to a transition is either the only arc from that place, or the only arc to that transition, are named as a *free-choice PN* – see Figure 3.

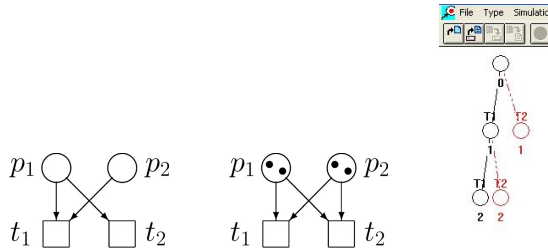


Figure 3. A fragment of the free-choice PN (left), a simple free-choice PN (in the centre) and its RT (right)

A net $N = \langle P, T, B \rangle$ is free-choice [12] if $(p, t) \in B$ implies ${}^{(p)}t \times p^{(t)} \subseteq B$ for every place p and every transition t . A net system $\langle N, M_0 \rangle$ is free-choice if its underlying net N is free-choice. In [12] also the fundamental property of a free-choice PN was proved. If a marking of N enables some transition of $p^{(t)}$, then it enables every transition of $p^{(t)}$.

1.2 Timed Petri Nets

Timed Petri Nets (TPN) [64, 59, 7, 9] are based on P/T PN structure. They rise by introducing time into P/T PN transitions through their duration function $D : \mathcal{T} \rightarrow \mathbb{Q}_0^+$ (\mathbb{Q}_0^+ symbolizes non-negative rational numbers). The timing can be deterministic (time delays) or nondeterministic (expressed by a probability distribution – e.g. exponential, discrete uniform, etc.). More details can be found in [59, 7, 9]. As to the DES control, at P/T PN a supervisor can be synthesized. By means of TPN the performance evaluation of the supervised system can be tested.

1.3 Closing Remark

The P/T PN and TPN are very useful tools for modelling, analysing and control of DES. Such models of DES represent an initial point (a base) for some procedures of DES control synthesis. However, in the course of time other kinds of PN arising from P/T PN were developed, that are more suitable for specific kinds of DES, especially designated for control – Controlled Petri Nets, and for control of DES containing nondeterminism in the form of unobservable/uncontrollable transitions and/or unmeasurable places – Labelled Petri Nets and Interpreted Petri Nets.

Because of the extent the problem (a broad issue), publication is divided into two separate parts – Part 1 and Part 2. The particular parts will be published as individual papers in successive steps. This paper represents the individual Part 1. It is devoted to presentation of the mentioned kinds of PN and marking the possibilities for applicability at modelling DES (in the first place) and also to point out that the PN can be used for control purposes. In the planned consecutive Part 2 more detailed case studies on applicability at control of different kinds of DES, will be introduced, including the error recovery approach at a fault occurrence. The Part 2 will be published as the separate paper.

2 SOME OF OTHER KINDS OF PN RELATED TO DES CONTROL

P/T PN is a good mathematical tool for modelling and analysing DES. However, creation of the model is not sufficient for the DES control. After creating the model of the system to be controlled, it is necessary to synthesize the procedure of control. The best way is when it is possible to utilize the model for the control synthesis not only in deterministic cases when all transitions are controllable and all places are measurable but also in cases where uncontrollable transitions and non-measurable places or even faults occur in the model. Then, two sets of transitions arise. That is to say, the set T_c of controllable transitions and the set T_u of uncontrollable transitions. $T = T_c \cup T_u$. Analogically, also two sets of places arise – the set P_m of the measurable places and the set P_{nm} of unmeasurable places, $P = P_m \cup P_{nm}$. Because of the existence T_c and T_u , there exist structural matrices \mathbf{B}^c and \mathbf{B}^u . There are also Parikh's vectors \mathbf{v}_k^c and \mathbf{v}_k^u . When t_i, t_j, t_k, \dots is a firing transition sequence, corresponding Parikh's vector $\mathbf{v}_k = \mathbf{v}_k^c + \mathbf{v}_k^u$ where a \mathbf{v}_k^c component $v_k^c = v(i)$ if t_i is measurable, otherwise $v_k^c = 0$, and a \mathbf{v}_k^u component $v_k^u = v(i)$ if t_i is unmeasurable, otherwise $v_k^u = 0$. Because of the existence of the P_m and P_{nm} the output vector \mathbf{y}_k has to be introduced. Namely, only measurable states are observable from outside. Thus,

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{B}^c \cdot \mathbf{u}_k^c + \mathbf{B}^u \cdot \mathbf{u}_k^u; \quad k = 0, 1, \dots, N, \quad (6)$$

$$\mathbf{y}_k = \mathbf{C} \cdot \mathbf{x}_k. \quad (7)$$

To deal with such a problem, there are several approaches how to do it. Some other kinds of PN exist which are more suitable for DES control. Moreover, they utilize the P/T PN model presented above. In order to present how they can be used in this direction, let us introduce some of them: Controlled PN, Labelled PN and Interpreted PN.

For the completeness' sake, it is necessary to bring to mind the above mentioned TPN suitable to analyse performance evaluation – see the author's papers [7, 8] where TPN were applied. Moreover, we must not forget Hybrid PN (HPN), more precisely First Order HPN (FOHPN), suitable for modelling hybrid systems containing a continuous part and a discrete-event part, that were also applied in [7, 8].

For these reasons given, these two kinds of PN are not mentioned in details here, in this paper.

2.1 Controlled Petri Nets

The P/T PN described above can be influenced from outside by means of external interferences and/or controller in order to affect the movement of tokens. In such a case we are speaking about Controlled Petri Nets (CtPN) – see e.g. [36, 32, 28, 29, 30, 31]. While P/T PN can be formally represented by the triple $N = \langle P, T, B \rangle$, CtPN can be expressed as the triple

$$PN_c = \langle N, P_c, B_c \rangle \tag{8}$$

with P_c being a finite set of control places where $P_c \cap P = \emptyset$, $P_c \cap T = \emptyset$, and $B_c \subseteq P_c \times T$ being the set of directed arcs from control places to P/T PN transitions. In CtPN the places $p \in P$ are named as state places. The set of control places entering a transition $t \in T$ can be expressed as ${}^{(p_c)}t = \{p_c | (p_c, t) \in B_c\}$. On the other hand for a control place $p_c \in P_c$ the set of its output transitions can be denoted as $p_c^{(t)} = \{t | (p_c, t) \in B_c\}$. Denote the set of all controlled transitions by T_c .

A control for a CtPN is understood to be a function $u : P_c \rightarrow \{0, 1\}$. It assigns a binary value to each control place. All such controls are denoted by \mathcal{U} . They influence a set of transitions $\mathcal{T} \in T$. When for all $t \in \mathcal{T}$, $u(p_c) = 1$ for all $p_c \in {}^{(p_c)}t$ the set \mathcal{T} is said to be control enabled. An example of CtPN is displayed in Figure 4. In the vector/matrix expression, CtPN can be described as follows:

$$\begin{pmatrix} \mathbf{x}_{k+1} \\ \mathbf{x}_{k+1}^c \end{pmatrix} = \begin{pmatrix} \mathbf{x}_k \\ \mathbf{x}_k^c \end{pmatrix} + \begin{pmatrix} \mathbf{B} \\ \mathbf{B}_c \end{pmatrix} \cdot \mathbf{u}_k, \quad k = 0, 1, \dots, N, \tag{9}$$

$$\begin{pmatrix} \mathbf{F} \\ \mathbf{F}_c \end{pmatrix} \cdot \mathbf{u}_k \leq \begin{pmatrix} \mathbf{x}_k \\ \mathbf{x}_k^c \end{pmatrix}, \tag{10}$$

$$\mathbf{x}_{k+1}^{ct} = \mathbf{x}_k^{ct} + \mathbf{B}_{ct} \cdot \mathbf{u}_k, \quad k = 0, 1, \dots, N, \tag{11}$$

$$\mathbf{F}_{ct} \cdot \mathbf{u}_k \leq \mathbf{x}_k^{ct} \tag{12}$$

where $\mathbf{x}_k^{ct} = (\mathbf{x}_k^T (\mathbf{x}_k^c)^T)^T$, $\mathbf{B}_{ct} = (\mathbf{B}^T \mathbf{B}_c^T)^T$, $\mathbf{F}_{ct} = (\mathbf{F}^T \mathbf{F}_c^T)^T$, $\mathbf{G}_{ct}^T = (\mathbf{G} \mathbf{G}_c)^T$, $\mathbf{B} = \mathbf{G}^T - \mathbf{F}$ and $\mathbf{B}_c = \mathbf{G}_c^T - \mathbf{F}_c$. Because \mathbf{F} and \mathbf{G}^T are the same as in (3), only the incidence matrices \mathbf{F}_c and \mathbf{G}_c^T are introduced as follows:

$$\mathbf{F}_c = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}; \quad \mathbf{G}_c^T = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{13}$$

However, because the control places have no input transitions, $\mathbf{G}_c^T = \mathbf{0}$. It means that the firing of transitions does not contribute to the marking development of the control places.

However, a suitable setting of the control place markings from outside, e.g. by means of a suitable controller, makes possible to control the model of the system. In such case the control places permanently represent an output of the controller.

Over time many approaches how to synthesize controller and/or supervisor for DES control were developed. There exist different kinds of the supervisor synthesis – see e.g. [61, 44, 45, 46, 50, 19, 66, 33, 34] and many others included also in [9, 7, 27].

2.1.1 An Example

Consider e.g. the group of five simple autonomous agents $GrA = \{A1, A2, A3, A4, A5\}$ with the same structure expressed by a working cycle $\{p_i, t_i, p_{i+1}, t_{i+1}\}$ – see Figure 5 – where particular places and transitions mean: p_i – the agent is idle; p_{i+1} – the agent is working; t_i – the agent is starting the work; t_{i+1} – the agent is ending the work.

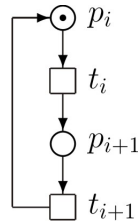


Figure 5. An example of the P/T PN

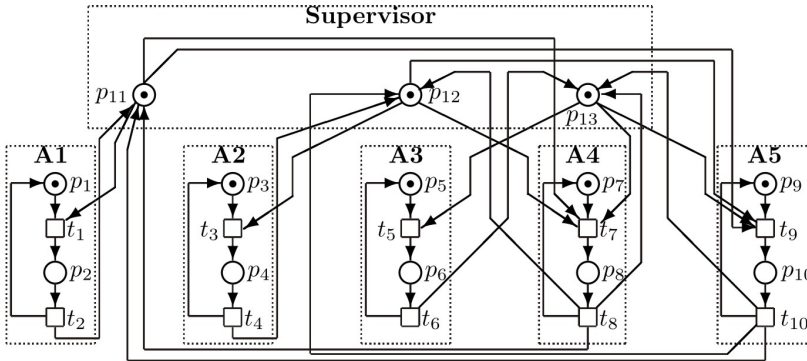


Figure 6. An example of the CtPN

Let us solve the situation when it is necessary to ensure that only one agent from the subgroup $Sgr1 = \{A1, A4, A5\}$ (a representative), only one agent from the subgroup $Sgr2 = \{A2, A4, A5\}$, and only one agent from the subgroup $Sgr3 = \{A3, A4, A5\}$ can simultaneously cooperate with other agents from GrA . In other words, the agents inside the designated subgroups must not work simultaneously.

Even, the agents $A4$ and $A5$ can work only individually (any cooperation with other agents is excluded). However, the agents $A1, A2, A3$ can work simultaneously. The conditions prescribing the cooperation of agents are:

$$\sigma_{p_2} + \sigma_{p_8} + \sigma_{p_{10}} \leq 1, \tag{15}$$

$$\sigma_{p_4} + \sigma_{p_8} + \sigma_{p_{10}} \leq 1, \tag{16}$$

$$\sigma_{p_6} + \sigma_{p_8} + \sigma_{p_{10}} \leq 1. \tag{17}$$

After applying the method of the supervisor synthesis (based on P-invariants of PN) presented in [7, 9] the PN model of the cooperating agents is displayed in Figure 6. As it can be seen, the supervisor (controller in DES relations) created by p_{11}, p_{12}, p_{13} coordinates the activities of the five agents. In other words, it becomes the additional sixth agent enabling the prescribed cooperation of the five agents.

2.1.2 Petri Nets Controllability and Observability

In PN the terms controllability and observability are nearly related. As to observability, the so called silent transitions [10, 13] are invisible in an environment and may cause problems with observability of discrete events. They represent the discrete events that cause a change in the state of the DES, however they are not observable from outside – i.e., they become unobservable. Therefore, it is particularly important to obtain a powerful approach to PN-based DES control which relies only on information about the observable transitions and forbids firing the unobservable ones. When a controller is not allowed to influence some transitions, such transitions become uncontrollable. Both such groups of transitions bring uncertainty into the DES control. Therefore, it is necessary to deal with the problem how to control DES in such uncertainty conditions.

More precisely said, a transition is named to be *unobservable* when its firing cannot be directly detected or measured. A transition is named to be *uncontrollable* when its firing cannot be inhibited by an external action. All unobservable transitions are implicitly uncontrollable [43, 44].

In [24, 47] controllability in the framework of Petri nets is mentioned. Unfortunately, the necessary and sufficient conditions being able to decide about controllability and observability are known only in continuous and/or discrete-time linear systems. In continuous linear time invariant systems on the controllability of a system with the state equation $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$, $\mathbf{x} \in \mathbb{R}^n$ (where \mathbb{R} represents real numbers), $\mathbf{u} \in \mathbb{R}^m$ decides – see e.g. [38, 20, 48] – the *rank* of the controllability matrix $\mathbf{CM} = [\mathbf{B}|\mathbf{AB}|\mathbf{A}^2\mathbf{B}|\dots|\mathbf{A}^{n-1}\mathbf{B}]$ with the dimensionality ($n \times n.m$). The system is controllable when $rank(\mathbf{CM}) = n$. This is valid not only for continuous linear systems but also for discrete-time modification of linear time invariant systems. A system is called controllable (or reachable) if all states are reachable (i.e. the reachable set $R \in \mathbb{R}^n$). The uncontrollable system in which uncontrollable part is stable is named the stabilizable system.

When the output equation of the continuous linear time invariant systems is $\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$, $\mathbf{y} \in \mathbb{R}^p$, the observability is decided on the *rank* of the observability matrix $\mathbf{OM} = [\mathbf{C}^T | \mathbf{A}^T \mathbf{C}^T | (\mathbf{A}^T)^2 \mathbf{C}^T | \dots | (\mathbf{A}^T)^{n-1} \mathbf{C}^T]$. The system is observable when $\text{rank}(\mathbf{OM}) = n$. In other words, a system is observable if the initial state can be obtained (observed) from the knowledge of the input and the output. Unobservable system in which the unobservable subsystem is stable is named as the detectable system.

However, in PN modelling the completely different kind of systems, i.e. DES, the *rank* of such controllability matrix $\mathbf{M}_c = (\mathbf{B} | \mathbf{B} | \dots | \mathbf{B})$ of the system (1) always coincides with the *rank* of the PN incidence matrix \mathbf{B} , because in (1) $\mathbf{A} = \mathbf{I}$ (i.e. the identity matrix). Therefore, $\text{rank}(\mathbf{B}) = n$ is only a necessary condition for controllability if the control input is restricted to $\mathbf{u}_k \in \{0, 1\}^m$ and to $\mathbf{x}_k + \mathbf{B}\mathbf{u}_k \geq \mathbf{0}$. Moreover, the PN state equation (1) itself only provides a necessary but not sufficient condition for reachability.

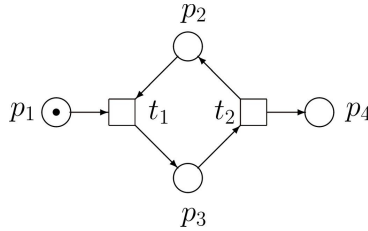


Figure 7. An example on the controllability of P/T PN

For example, consider the net system in Figure 7 with the structure

$$\mathbf{F} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}; \quad \mathbf{G}^T = \begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}; \quad \mathbf{B} = \mathbf{G}^T - \mathbf{F} = \begin{pmatrix} -1 & 0 \\ -1 & 1 \\ 1 & -1 \\ 0 & 1 \end{pmatrix} \quad (18)$$

and initial state $\mathbf{x}_0 = (1, 0, 0, 0)^T$. Although the $\text{rank}(\mathbf{B}) = 2$, even for $\mathbf{u}_0 = (1, 1)^T$ when $\mathbf{x}_0 + \mathbf{B}\mathbf{u}_0 \geq \mathbf{0}$ the state $\mathbf{x}_1 = (0, 0, 0, 1)^T$ is not reachable. In fact, no transition is enabled at the initial state. Namely, neither the sequence $t_1 t_2$ nor the sequence $t_2 t_1$ can be fired because they are not enabled. It means practically that no RT exists (more precisely, RT consists of only one node \mathbf{x}_0 and has no arcs).

Still more complicated situation than that in case of the controllability is the situation with the observability of PN. Because $\mathbf{A} = \mathbf{I}$ the *rank* of the observability matrix $\mathbf{M}_o = (\mathbf{C}^T | \mathbf{C}^T | \dots | \mathbf{C}^T)$ is reduced on the *rank* of the matrix \mathbf{C}^T , which principally cannot have the $\text{rank}(\mathbf{C}^T) = n$. Really, the number of measurable places $p < n$, where n is the total number of places.

Only very few published works appertain to observability in PN – see e.g. [52, 25, 22, 54, 65]. In [65] CtPN model is used for forbidden state avoidance under

partial event observation with the assumption that the initial marking is known. The observability properties depend not only on the net structure N , but also on the initial marking m_0 , that in [22] is assumed to be unknown. The structural observability and marking observability are distinguished.

The structural observability requires the study of the system properties for all possible initial markings. If a place $p \in P$ is observable in $\langle N, M \rangle$ then it is also observable in $\langle N, M' \rangle \forall M' \geq M$ with $M'(p) = M(p)$.

The marking observability means that there exists at least one word that is complete, while strong marking observability means that all words can be completed in a finite number of steps into a complete word. Here, the term *word* means a word of events – i.e. a sequence of transition firings. It has the direct relationship with RG and/or CG. Observability in general [54] allows, through an observer, the computation of state variable values that cannot be directly measured. Observers are used to estimate the system state.

Finally, it has to be said that for control of PN with unobservable/uncontrollable transitions and non-measurable places CtPN are not too suitable. Consequently, it is necessary to apply next kinds of PN mentioned below, i.e. Labelled PN and Interpreted PN.

2.2 Labelled Petri Nets

Labelled Petri nets (LbPN) are the standard Petri nets with a function attaching a label to each event. As a fundamental operation in modular design the parallel composition LbPN is used. Often, models of subsystems are combined into a model of the whole system. PN languages (at least their simple forms) are applied in LbPN.

In terms of a separate set of event labels, PN languages were defined e.g. in [14, 35, 37, 40, 47, 49, 63], etc. Events can be assigned to some transitions, even to all of the transitions. Then, the firing of a transition represents an event labelled by the corresponding label. The set of all sequences of admissible event is named as PN language.

In general, the formal definition of LbPN is the following:

$$LbPN = \langle N, \mathcal{L}, l, m_0, F_m \rangle. \tag{19}$$

Here N is the PN structure; $\mathcal{L} = L \cup \varepsilon$ is an alphabet representing a finite set of events, where L represents observable events and ε represents unobservable events; $l : T \rightarrow \mathcal{L}$ is a labeling function assigning an event to each transition; $m_0 \in \mathcal{M}$ is an initial marking, with \mathcal{M} being a set of markings; $F_m \in \mathcal{M}$ is a finite set of final markings.

At the simplest understanding of LbPN – in the sense of the λ -free PN language [49] – no transition is labelled with the empty string λ and two or more transitions may have the same label. Otherwise, especially in case of so called silent events [23, 4] being unobservable and causing a change of DES markings (states),

transitions may be labelled with the empty string ε . In case of indistinguishable events, that may yield two or more new states from a given state, two or more transitions are labeled with the same symbol and enabled at a given state.

2.2.1 State Estimation

In linear time invariant systems (continuous and/or discrete-time) in case of presence of the state disturbance or noise and sensor noise or error which are described statistically, or assumed to be small, when $\mathbf{u}(t)$ and $\mathbf{y}(t)$ are observed on an interval $(0, t - 1)$ the problem of the state estimation is the following. To estimate $\mathbf{x}(s)$ from $\mathbf{u}(t)$ and $\mathbf{y}(t)$ three possibilities exist:

1. to estimate the initial state ($s = 0$);
2. the current state ($s = t - 1$);
3. to estimate (i.e. predict) next state ($s = t$).

The algorithm or a system yielding the estimate $\hat{\mathbf{x}}(s)$ is called an observer or state estimator.

In PN the situation is different, likewise as in case of the controllability and observability. The state estimation depends on the so called PN sensors.

2.2.2 Sensors in PN

The model of PN with outputs consists not only from the state equation but also from the output equation $\mathbf{y}_k = \mathbf{C} \cdot \mathbf{x}_k$. PN with outputs [2, 56] are PN with the so called *place sensors* which count the tokens contained in some places – named as measured or observable places (in modelled DES e.g. vision sensors, touch sensors, etc.), and *transition sensors* which detect the firing of some of the transitions – named as measured or observable transitions (in real DES e.g. motion sensors, speed sensors, etc.). However, not all places and not all transitions may have a sensor. As it was premised above, in modelled DES not all state variables and not all control variables can be measured or observed. More precisely, a place sensor configuration V is a vector $(v_1, v_2, \dots, v_n)^T$, where $v_i = 0$ if no place sensor exists on place p_i and $v_i = 1$ otherwise. $|V| = \sum_{i=1}^n v_i \leq n$ denotes the total number of place sensors in the place sensor configuration V . On the other hand, analogically, $|L| \leq m$ is the total number of transition sensors in use and could be zero if no transition sensor is used.

As it was already mentioned, LbPN may have nondeterministic transitions – i.e. transitions that share the same label or unobservable transitions associated with the null label. Other faults are modelled also as unobservable transitions.

The set of transitions $T = T_d \cup T_n$, where T_d is a set of deterministic transitions, while T_n is the set of nondeterministic transitions. For deterministic PN with the so called λ -free labeling function it is defined [23] that if two transitions are labelled with the same symbol they cannot simultaneously be enabled at \mathcal{M} . Therefore, for

structurally deterministic function two different transitions cannot be labelled with the same symbol. Thus, a transition t is nondeterministic if its label is associated also to other transitions, otherwise a transition t is said to be deterministic. The association between sensors and transitions can be captured by a labeling function $\mathcal{L} : T \rightarrow L \cup \{\varepsilon\}$ that assigns to each transition $t \in T$ either a symbol from a given alphabet L or the empty string ε . The set of transitions whose label is ε is denoted as $T_u = \{t \in T | \mathcal{L}(t) = \varepsilon\}$. Transitions $t \in T_u$ are called unobservable or silent. T_o denotes the set of transitions labelled with a symbol in L . Transitions $t \in T_o$ are called observable, because when they fire, their label can be observed. In general, the same label $l \in L$ can be associated to more than one transition. In particular, two transitions $t_1, t_2 \in T_o$ are called non-distinguishable if they share the same label, i.e., $L(t_1) = L(t_2)$. The set of transitions sharing the same label l are denoted as T_l .

In LbPN whose initial marking is not known exactly [5], marking estimation is possible. It is sufficient to know only that the marking belongs to a given convex set. The silent transitions (i.e. labelled with the empty word) and indistinguishable transitions (i.e. sharing the same label with other transitions) are allowed on that way.

For the partially observed LbPN with a place sensor configuration V , a labeled function \mathcal{L} and an *observation sequence*, the set of *consistent firing sequences* and *consistent markings* can be found [55]. The firing of t will generate token changes in place sensors and/or an observable transition label.

2.2.3 Illustrative Example 1

Consider a partially observed P/T PN given in Figure 8 (left), where t_4, t_5 are unobservable.

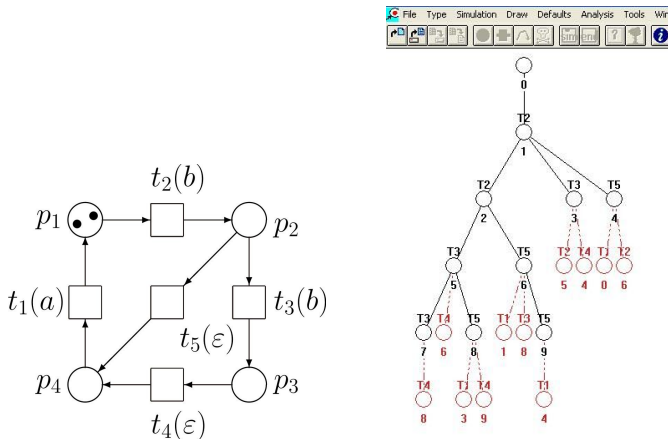


Figure 8. An example of partially observed LbPN (left) and the RT corresponding to the fully observable net (right)

Consequently, p_4 is unobservable too because both of its input transitions are unobservable. Thus, $P_o = \{p_1, p_2, p_3\}$ is the set of observable places while $P_{uo} = \{p_4\}$ is the set (singleton) of unobservable ones. Consider that only p_2 is equipped by a sensor. Next, $T_o = \{t_1, t_2, t_3\}$ is the set of observable transitions while $T_u = \{t_4, t_5\}$ is the set unobservable ones. The labeling function \mathcal{L} is given as $\mathcal{L}_1 = a, \mathcal{L}_2 = \mathcal{L}_3 = b, \mathcal{L}_4 = \mathcal{L}_5 = \varepsilon$.

An observable transition $t \in T_o$ can have a sensor that indicates when a transition within a given subset of transitions has fired. Any unobservable transition $t \in T_u = T - T_o$ cannot have such a sensor associated with it. The association between sensors and transitions is captured by a labeling function $\mathcal{L} : T \rightarrow L \cup \{\varepsilon\}$ that assigns a label to each transition. Unlike the fixed labeling function in LbPN, the labeling function \mathcal{L} in a partially observed PN can be configured subject to the constraint that unobservable transitions must be assigned the null label – $\mathcal{L}(t) = \varepsilon$ for all $t \in T_u$.

In case of fully observable net, its RT has the form as it is displayed in Figure 8 (right) with the nodes represented by the columns of the matrix

$$\mathbf{X}_r = \begin{pmatrix} 2 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 2 \end{pmatrix}. \tag{20}$$

LbPN are suitable for testing the observability of places and transitions. For example let the transition t_5 be the only fault transition which needs to be detected. Firing the sequence of transitions $S = t_2t_5$ the system trajectory is $M0[t_2 > M1[t_5 > M4$, where $M0 = (2000)^T, M1 = (1100)^T$ and $M4 = (1001)^T$, and the corresponding observation from place and transition sensors is $M0 \rightarrow b \rightarrow M1 \rightarrow \varepsilon \rightarrow M4 \rightarrow a \rightarrow M0$. Hence, we can deduce that the fault transition t_5 had to be occurred. Namely, only the firing of t_5 can decrease the number of tokens in p_2 by 1 and at the same time not generate any label – b in t_3 . The observation is driven by token changes in observable places with sensors and/or observed labels. When the observed label is ε , it means that there is no observation output from transition sensors. Because of the unobservable transitions and the unobservable place, the actual marking of p_4 and actual firing of t_4, t_5 cannot be anticipated (they are unpredictable).

2.2.4 Illustrative Example 2

To illustrate LbPN deeper, consider the example of such a net given in Figure 9.

There are 13 transitions in the net. Consider that the observable transitions create the set $T_o = \{t_1, t_2, t_6, t_7, t_8, t_9, t_{12}\}$ while the unobservable ones create the set $T_u = \{t_3, t_4, t_5, t_{10}, t_{11}, t_{13}\}$, i.e. $T_u = \{\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4, \varepsilon_5, \varepsilon_6\}$. Let the labeling functions of the observable transitions be $\mathcal{L}(t_1) = a, \mathcal{L}(t_2) = \mathcal{L}(t_6) = b, \mathcal{L}(t_7) = \mathcal{L}(t_8) = c, \mathcal{L}(t_9) = \mathcal{L}(t_{12}) = d$. RT of the fully observable net is given in Figure 10.

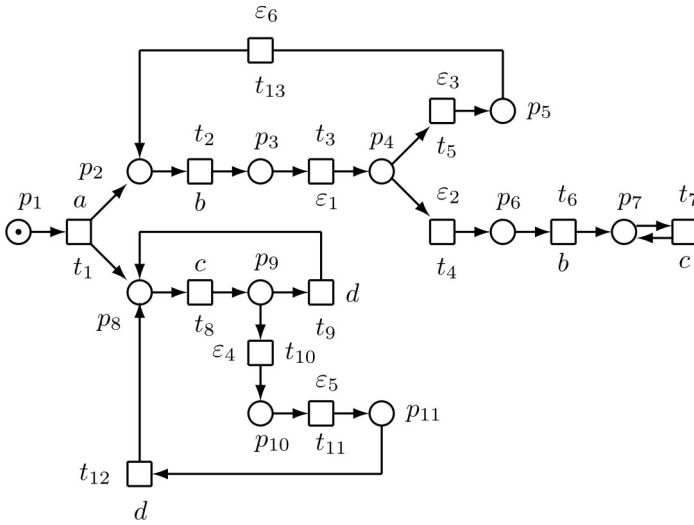


Figure 9. An example of LbPN

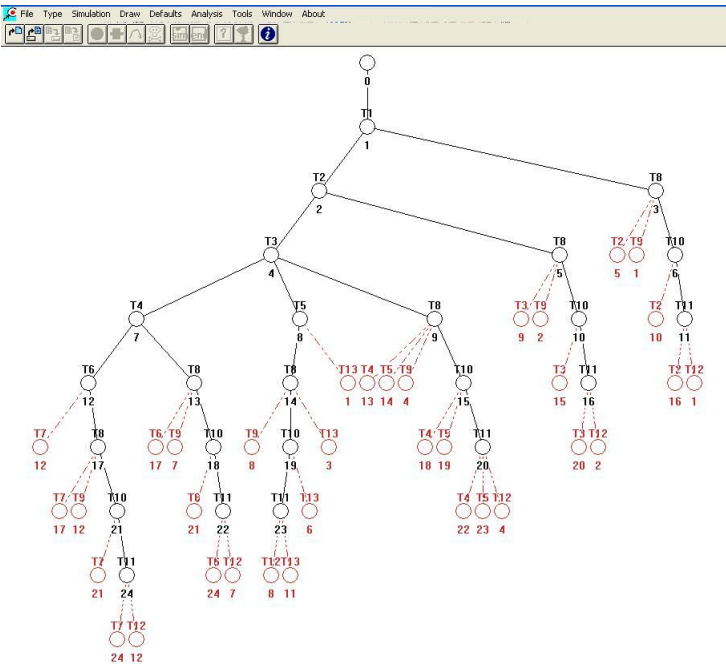


Figure 10. The RT of the corresponding P/T PN

It has 25 nodes being columns of the matrix where the first column represents the initial marking M_0 .

$$\mathbf{X}_r = \begin{pmatrix} 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}. \quad (21)$$

When we consider the input word $w = ab$, we obtain the set of consistent firing sequences in the form $S(w) = \{t_1t_2, t_1t_2\varepsilon_1, t_1t_2\varepsilon_1\varepsilon_2, t_1t_2\varepsilon_1\varepsilon_2\varepsilon_6, t_1t_2\varepsilon_1\varepsilon_3\}$. In RT they cause the set of consistent markings M_1, M_2, M_4, M_8 and M_7 represented by the columns 2, 3, 5, 9, 8 of the reachability matrix \mathbf{X}_r , respectively. When the word is considered to be $w = acd$, then $S(w) = \{t_1t_8t_9, t_1t_8\varepsilon_4\varepsilon_5t_{12}\}$, the set of markings is represented by the column 2. Both cases of the word w are displayed in Figure 11 – the former word on the left branch of the cropped RT, while the latter one on the right branch.

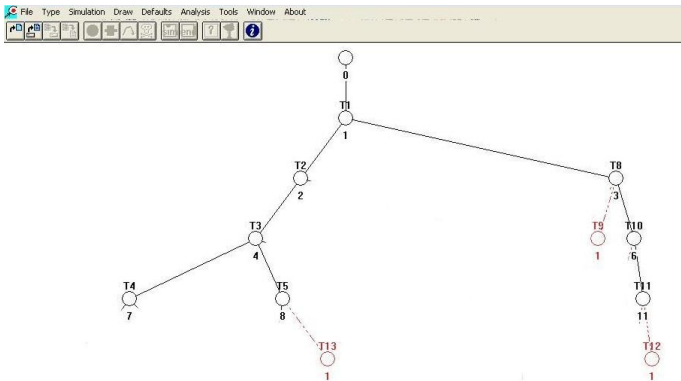


Figure 11. The cropped RT expressing the reachable markings corresponding to the both words – on the left side to the word $w = ab$ and on the right side to the word $w = acd$

2.3 Interpreted Petri Nets

IPN can be formally described [1, 39, 54, 57, 58, 8] by the quadruplet as follows:

$$IPN = \langle N, \Sigma, \Phi, \lambda, \Psi, \varphi \rangle \quad (22)$$

where N is the PN structure; $\Sigma = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ is an input alphabet with α_r , $i = 1, 2, \dots, r$ being input symbols; $\Phi = \{\delta_1, \delta_2, \dots, \delta_s\}$ is the output alphabet with δ_i , $i = 1, \dots, s$, being the output symbols; $\lambda = T \rightarrow \Sigma \cup \{\varepsilon\}$ is the labeling function of transitions (assigning either an input symbol $\alpha_i \in \Sigma$ or the internal event ε to each transition) with the constraint: $\forall t_j, t_k \in T$, $j \neq k$, if $\forall p_i F(p_i, t_j) = F(p_i, t_k) \neq 0$ and both $\lambda(t_j) \neq \varepsilon$, $\lambda(t_k) \neq \varepsilon$ then $\lambda(t_j) \neq \lambda(t_k)$ – i.e. each transition is assigned a unique label; $\Psi : P \rightarrow \Phi \cup \{\varepsilon\}$ labels the places (either an output symbol $\delta_i \in \Phi$ or the null output signal ε is assigned to each PN place by this function); $\varphi : R(IPN, M_0) \rightarrow \mathbb{Z}_{\geq 0}^{q \times n}$ is an output function that associates an output vector to each marking $R(IPN, M_0)$. Here $q \in \mathbb{Z}_{>0}$ is a positive integer representing the number of available output signals and $n = |P|$ – i.e. the number of all places in the set P . It means that φ is $(q \times n)$ matrix.

IPN are to model the DES behavior that includes partially observable both events and states [53]. The net system where each transition is assigned a unique label is named as free-labeled PN. Identifying of such nets is possible. The approaches consist in observing the marking of a subset of places and (when some additional information on the dependency between transitions is given) allow to reconstruct the part of the net structure related to unobservable places. Such approaches can be found e.g. in [41, 42] and the approach based on integer programming in [26].

3 IPN IN MODELLING AND CONTROL OF DES

The control specifications create [21] a set of forbidden markings M_F which correspond to undesirable states. Namely, they either jeopardize the system safety or they give birth to deadlock situations. Therefore, it is necessary to determine a convenient set of places that, after adding to the PN model of the plant, will prevent the whole system from reaching these states. When the PN model has uncontrollable transitions it is necessary to prevent the system from reaching the forbidden markings, containing all dangerous markings from which a forbidden one may be reached by firing a sequence of uncontrollable transitions from T_u .

Let M_D be the set of dangerous markings, i.e. $M_D = \{M \in R(N, M_0) \mid \exists M' \in M_F \wedge \tau \in T_u^*, M[\tau > M']\}$, where T_u^* is a sequence of uncontrollable transitions. Of course, $M_F \subseteq M_D$. The controlled system has to be safe (without forbidden markings) and live (without deadlocks). Then, the set M_L of legal (i.e. admissible) markings is the maximal set of reachable markings such that

1. $M_L \cap M_D = \emptyset$;

2. it is possible to reach the marked marking M_0 from any legal marking without leaving the set M_L ;
3. any transition t from a legal marking to a nonlegal marking is a controllable transition.

Of course the full RG is cropped into R_c being the RG containing all legal markings. Thus, $M_L \subseteq R(N, M_0) \setminus M_D$.

It was shown in [50] that M_L exists and it is called the *maximally permissive behavior*. M_L is such that, whatever the marking in M_L , the system cannot be uncontrollably led outside M_L .

IPN are closely related to LbPN. Even, it may be said that IPN are a modification of LbPN. They are [51] an extension of PN that allow to represent *the output signals generated when a marking is reached*, and *the input signals associated with transitions that are controllable*. These signals are useful to infer the initial marking of the net. Such a kind of PN is also suitable to model and control DES the PN model of which contains uncontrollable transitions and unmeasurable places.

Let us introduce two illustrative examples concerning

1. the principle of creating the IPN model of DES at the existence of uncontrollable transitions and unmeasurable places;
2. the principle of its control.

3.1 Example – Creation of IPN Model

To show how to create the IPN model let us consider the simple PN-model given in Figure 12.

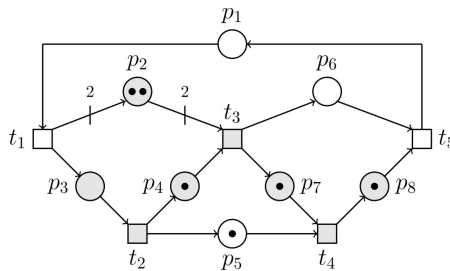


Figure 12. An example of the PN-based model

The full RT (of the net without taking unmeasurable places and uncontrollable transitions into account) is displayed in Figure 13 left. The particular RT nodes

create the columns of the following matrix:

$$\mathbf{X}_r = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 & 2 & 2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 2 & 2 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 2 & 0 & 1 & 2 & 1 & 2 & 1 & 2 & 3 \\ 1 & 1 & 2 & 2 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}. \quad (23)$$

Suppose that the measured places are $P_m = \{p_1, p_5, p_6\}$ and the unmeasured ones (filled by the gray color) are $P_{um} = P \setminus P_m = \{p_2, p_3, p_4, p_7, p_8\}$. Consider that the controllable transitions are $T_c = \{t_1, t_5\}$ while the uncontrollable ones (filled by the gray color) are $T_u = T \setminus T_c = \{t_2, t_3, t_4\}$. Consider for such IPN the input alphabet $\Sigma = \{a, b\}$ and the output alphabet $\Phi = \{\delta_1, \delta_2, \delta_3\}$. Consequently, $\lambda(t_k)_{k=1, \dots, 5} = \{a, \varepsilon, \varepsilon, \varepsilon, b\}$, $\Psi(p_i)_{i=1, \dots, 8} = \{\delta_1, \varepsilon, \varepsilon, \varepsilon, \delta_2, \delta_3, \varepsilon, \varepsilon\}$. The output equation is as follows:

$$\mathbf{y}_k = \varphi \cdot \mathbf{x}_k \quad \text{where} \quad \varphi = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}. \quad (24)$$

It is necessary to say that the RT of such IPN is not connected in spite of the fact that t_1, t_5 are controllable, because we do not know what is up with the marking development inside of the *foggy area* (containing uncontrollable transitions and states containing unmeasurable places – see the right side of Figure 13).

Because at the beginning p_1 is passive ($\sigma_{p_1} = 0$) and p_5 is active ($\sigma_{p_5} = 1$), the initial state is $\mathbf{x}_0 = (0, \varepsilon, \varepsilon, \varepsilon, 1, 0, \varepsilon, \varepsilon)^T$. Neither t_1 nor t_5 are enabled (regardless of the activity or passivity of unmeasurable places p_2 and p_8 , respectively) because measurable places p_1, p_6 are passive (without tokens).

On the other hand for the initial state $\mathbf{x}_0 = (0, 2, 0, 1, 1, 0, 1, 1)^T$ the output of the PN (without taking the influence of uncontrollable transitions into account) is $\mathbf{y}_0 = (0, 1, 0)^T$.

3.2 Example – Principle of IPN Model Control

To explain how the IPN model is controlled, consider a segment shown in Figure 14 left. While the upper *line* containing p_4 and t_3 represents the fragment of the model of the control system PN_{cs} (containing the control specifications), the lower line represents the fragment of the IPN model of the controlled plant PN_{pl} . The RG of the model is given in Figure 14 right, where $\mathbf{x}_0 = (1, 0, 0, 1)^T$, $\mathbf{x}_1 = (0, 1, 0, 1)^T$, $\mathbf{x}_2 = (0, 0, 1, 1)^T$ and $\mathbf{x}_3 = (0, 0, 1, 0)^T$.

Here, the controllable transition t_1 is enabled because it is required to reach the stated output while p_4 represents the state of a sensor. The self-loop between them represents the relation between the place of the control specification and the

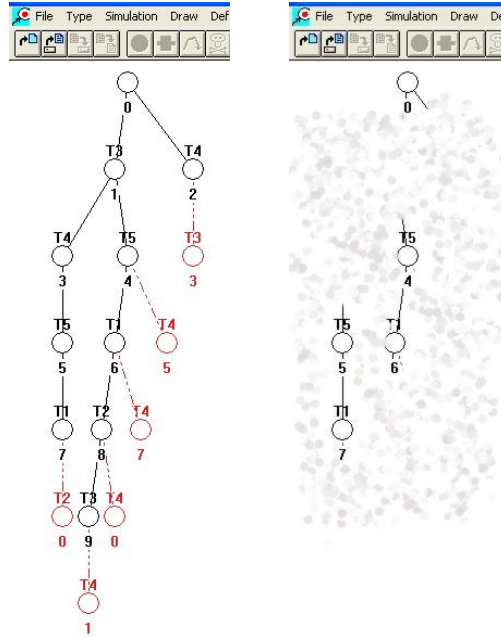


Figure 13. The RT corresponding to the PN-based model (left) and to the IPN model (right)

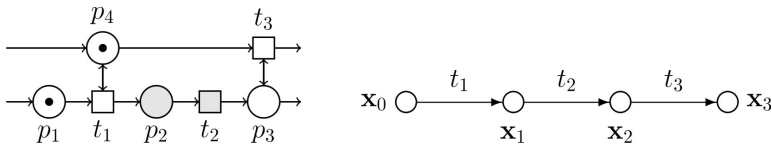


Figure 14. The controlled segment of the IPN-based model (left) and its RG (right)

plant controllable discrete event. The transition t_3 represents enabling the event expressing the situation when the plant and control specification have the same output and p_3 represents the state of the sensor. The self-loop between them expresses the relation between the plant measured place and the control specification. Thus, t_2 is bypassed and it can fire (as an internal event) or not. Note that the fragment of RG accordant with the segment of the controlled plant is straight-lined. Such fragments occur also in more complicated structures. Of course, RG of more complicated structures of the plant models will not be straight-lined.

3.3 Illustrative Example on Robotized Cell

Consider the robotized cell schematically drawn in Figure 15.

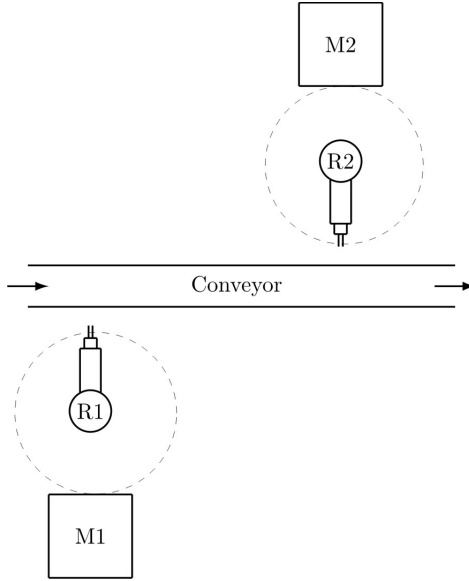


Figure 15. The schema of the plant

There are two robots R1, R2 serving, respectively, machines M1, M2. R1 inserts into M1 a raw material fed by the Conveyor from an input buffer. M1 machines the raw material. After finishing operations the intermediate product is unloaded from M1 by R1 and put on the Conveyor. By means of the Conveyor the semi-product proceeds towards M2. R2 takes it up and inserts it into M2. After finishing operations the final product is unloaded from M2 by R2 and put on the Conveyor which conveys it to an output buffer. The PN model of the plant operation is given in Figure 16 left. Its RT is given in Figure 16 right. It is very simple, however only in case when all transitions are controllable and all places (states) are measurable. The nodes of the RT are given by rows of the following reachability matrix representing the state vectors $\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_8$:

$$\mathbf{X}_r^T = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}. \quad (25)$$

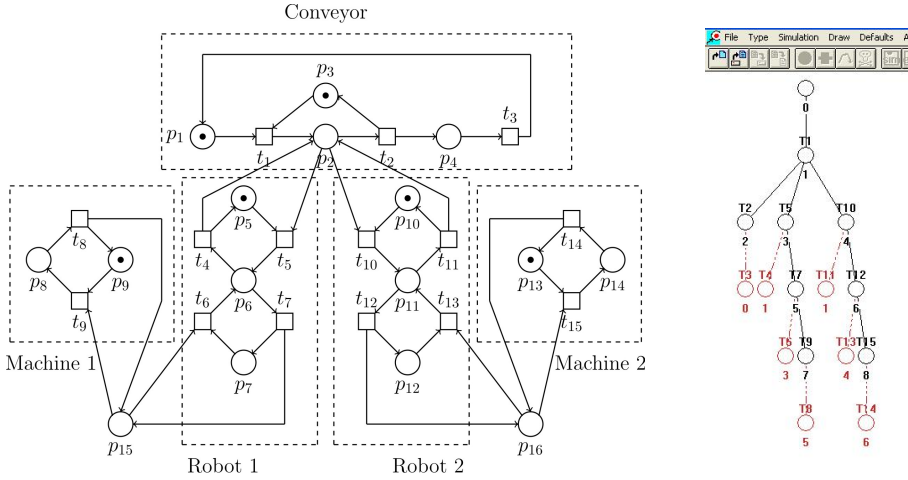


Figure 16. The PN model of the plant (left) and its RT (right)

In opposite case, when there are the uncontrollable transitions and unmeasurable places in the model, the situation is dramatically changed. The IPN model of the plant – see Figure 17 left – has the same structure, but transitions and places filled in gray color complicate the situation. They represent uncontrollable transitions $T_u = \{t_3, t_7, t_8, t_{12}, t_{14}\}$ and unmeasurable places $P_{um} = \{p_4, p_6, p_8, p_{11}, p_{14}\}$. Of course, controllable transitions create the subset $T_c = T \setminus T_u$ and measurable places create the subset $P_m = P \setminus P_{um}$.

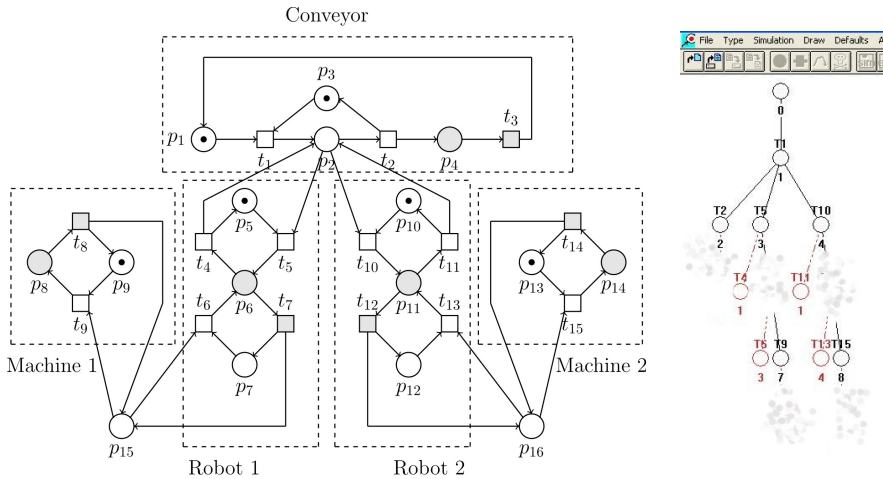


Figure 17. The IPN model of the plant (left) and its RT (left)

In such a case also the previous RT is not clear because of the uncontrollable transitions $t \in T_u$ – see Figure 17 right. Then,

$$\mathbf{X}'_r = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \varepsilon & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & \varepsilon & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \varepsilon & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \varepsilon & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \varepsilon \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}. \tag{26}$$

Because of $p \in P_{um}$ the output vector of such a system is the following:

$$\mathbf{y}_k = \varphi \cdot \mathbf{x}_k \tag{27}$$

where

$$\varphi = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Also in such a case the plant can be controlled by the controller. The structure of the plant with the incorporated controller is in Figure 18.

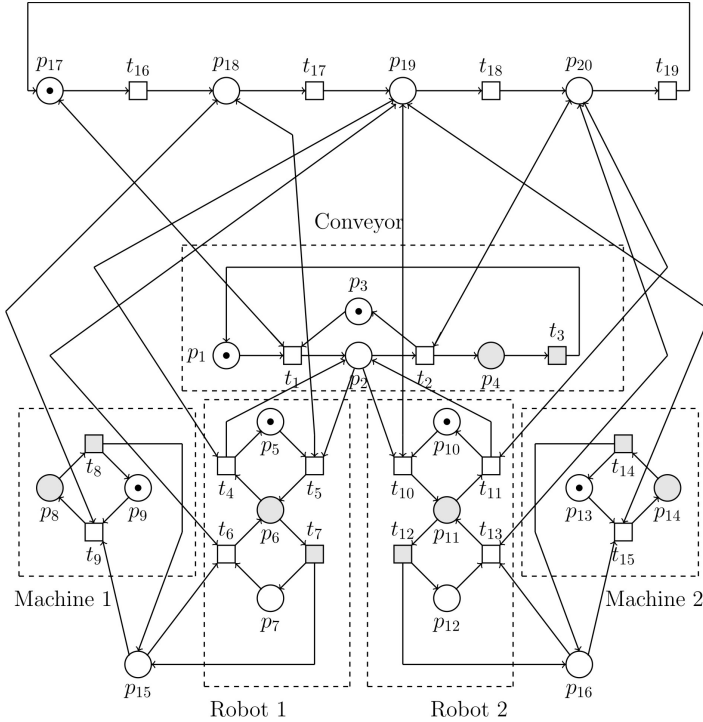


Figure 18. The controlled IPN model of the plant

occur, especially LbPN and IPN are suitable. In both areas of PN models illustrative examples were introduced.

While in the former PN area it is possible to find feasible paths (control trajectories) without big problems, in the latter one this cannot be said. Namely, uncertainty demands completely different approach not only to the model creation in order to express appropriately the uncontrollable/unobservable transitions and unobservable (unmeasurable) places but also to the control synthesis. Though the model is created, it still does not mean that the control synthesis will be simple. Consecutive problems yet occur concerning the controllability and observability of the model which are fundamental over the control synthesis. Moreover, for the state estimation PN sensors are very important as well as the points in PN where they have to be located. Only after the correct dislocation of sensors the successful control can be expected.

For modelling and simulation of DES by means of PN the simulation tools are used. In principle, there are two kinds of simulators:

1. graphical, and
2. computational.

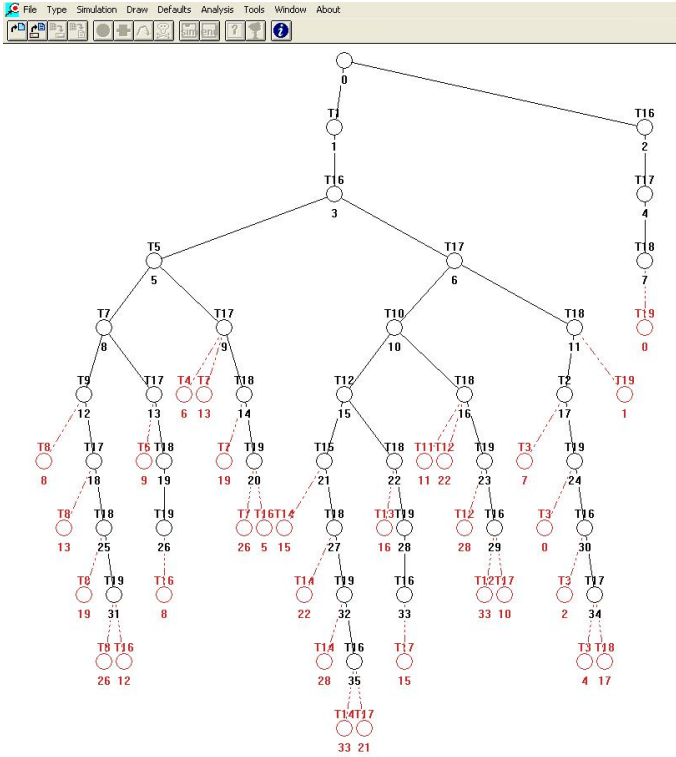


Figure 19. The controlled IPN model of the plant

In case of graphical simulators the PN is drawn by means of clicking on icons (place, transition, arc, token). Then, the marking evolution can be monitored step-by-step by means of clicking on enabled transitions (being distinguished from disabled transitions in color). Some of such simulators are able to draw RT and/or test the basic properties of PN. The graphical simulators are either free or commercial. The list of some available simulators is on the web site <http://www.informatik.uni-hamburg.de/TGI/PetriNets/index.php>. The graphical simulators are applicable for not very large PN models. The computational simulators are built on different bases (C++, Java, etc.). However, for common users the simulators utilizing Matlab tool are most friendly and applicable. Although Matlab contains own PN graphical tool, from computational point of view the simulator HYPENS (suitable for computational simulation of P/T PN, TPN, HPN, FOHPN) [59] used in Matlab is more user friendly. It works with matrix/vector based model for which the Matlab tool is an ideal environment. The computational simulators are suitable for reasonable larger PN models in comparison with the graphical ones. Moreover, Matlab itself makes possible to utilize prearranged computational procedures for

the supervisor synthesis as well as to test the behaviour and properties of the final model of the supervised system.

The presented paper represents only Part 1 of the whole paper. It is planned to publish also the separate Part 2 of this paper where also several deeper case studies (including the error recovery approach at fault occurrence, e.g. when a part drops from the robot gripper during an assembly process) will be presented in order to better document the applicability of the approaches on models of DES in a real environment (practice).

Acknowledgement

The author thanks for the partial support of the VEGA Agency (under Grant No. 2/0029/17).

REFERENCES

- [1] AGUIRRE-SALAS, L. I.—SANTOYO-SANCHEZ, A.: Observability Analysis of Interpreted Petri Nets Under Partial State Observations Using Estimations Reachability Graph. Proceedings of the 2008 IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2008), Hamburg, Germany, 2008, pp. 129–136, doi: 10.1109/ETFA.2008.4638383.
- [2] ARICHI, F.—KEBABI, H.—CHERKI, B.—DJEMAI, M.: Failure Components Detection in Discrete Event Systems Modeled by Petri Net. Proceedings of IEEE 3rd International Conference on Systems and Control, October 2013, Algiers, Algeria, 2013, pp. 224–229, doi: 10.1109/ICoSC.2013.6750863.
- [3] CABASINO, M. P.—GIUA, A.—SEATZU, C.: Identification of Deterministic Petri Nets. Proceedings of the 8th International Workshop on Discrete Event Systems (WODES'06), Ann Arbor, Mich., USA, July 2006, pp. 325–331, doi: 10.1109/WODES.2006.382527.
- [4] CABASINO, M. P.—GIUA, A.—SEATZU, C.: Diagnosis Using Labeled Petri Nets with Silent or Undistinguishable Fault Events. IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems, Vol. 43, 2013, No. 2, pp. 345–355.
- [5] CABASINO, M. P.—HADJICOSTIS, C. N.—SEATZU, C.: State Feedback Control of Labeled Petri Nets with Uncertainty in the Initial Marking. Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA 2014), Barcelona, Spain, September 2014, pp. 1–7, doi: 10.1109/ETFA.2014.7005154.
- [6] CHAO, D. Y.: Knitting Technique with TP-PT Generations for Petri Net Synthesis. Journal of Information Science and Engineering, Vol. 22, 2006, No. 4, pp. 909–923.
- [7] ČAPKOVIČ, F.: Timed and Hybrid Petri Nets at Solving Problems of Computational Intelligence. Computing and Informatics, Vol. 34, 2015, No. 4, pp. 746–778.
- [8] ČAPKOVIČ, F.: Interpreted Petri Nets in DES Control Synthesis. In: Nguyen, N. T., Trawiński, B., Fujita, H., Hong, T. P. (Eds.): Intelligent Information and Database

- Systems (ACIIDS 2016). Springer, Berlin, Heidelberg, Lecture Notes in Artificial Intelligence, Vol. 9621, 2016, pp. 377–387, doi: 10.1007/978-3-662-49381-6_36.
- [9] ČAPKOVIČ, F.: Petri Nets in Discrete-Event and Hybrid Systems Modelling, Analysing, Performance Evaluation and Control. In: Szewczyk, R., Zieliński, C. Kaliczyńska, M. (Eds.): Automation 2017 (ICA 2017). Innovations in Automation, Robotics and Measurement Techniques. Springer, Cham, Advances in Intelligent Systems and Computing, Vol. 550, 2017, pp. 3–21.
- [10] CORONA, D.—GIUA, A.—SEATZU, C.: Marking Estimation of Petri Nets with Silent Transitions. Proceedings of the 43rd IEEE International Conference on Decision and Control (CDC), Atlantis, The Bahamas, December 2004, pp. 966–971, doi: 10.1109/CDC.2004.1428810.
- [11] DESEL, J.—REISIG, W.: Place/Transition Petri Nets. In: Reisig, W., Rozenberg, G. (Eds.): Lectures on Petri Nets I: Basic Models. Advances in Petri Nets (ACPN 1996). Springer, Heidelberg, Lecture Notes in Computer Science, Vol. 1491, 1998, pp. 122–173.
- [12] DESEL, J.—ESPARZA, J.: Free Choice Petri Nets. Cambridge University Press, 1995.
- [13] ELHOG-BENZINA, D.—HADDAD, S.—HENNICKER, R.: Refinement and Asynchronous Composition of Modal Petri Nets. In: Jensen, K., Donatelli, S., Kleijn, J. (Eds.): Transactions on Petri Nets and Other Models of Concurrency V. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 6900, 2012, pp. 96–120.
- [14] ESPARZA, J.—NIELSEN, M.: Decidability Issues for Petri Nets. BRICS Report Series RS-94-8, May, 1994, 25 pp.
- [15] ESTRADA-VARGAS, A. P.—LÓPEZ-MELLADO, E.—LESAGE, J.-J.: A Black-Box Identification Method for Automated Discrete-Event Systems. IEEE Transactions on Automation Science and Engineering, IEEE, 2015, Vol. 14, No. 3, pp. 1321–1336, doi: 10.1109/TASE.2015.2445332.
- [16] ESTRADA-VARGAS, A. P.—LÓPEZ-MELLADO, E.—LESAGE, J.-J.: Identification of Partially Observable Discrete Event Manufacturing Systems. Proceedings of the 18th IEEE Conference on Emerging Technologies and Factory Automation (ETFA 2013), 2013, pp. 1–7, doi: 10.1109/ETFA.2013.6648052.
- [17] ESTRADA-VARGAS, A. P.—LESAGE, J.-J.—LÓPEZ-MELLADO, E.: A Stepwise Method for Identification of Controlled Discrete Manufacturing Systems. International Journal of Computer Integrated Manufacturing, Vol. 28, 2015, No. 2, pp. 187–199, doi: 10.1080/0951192X.2013.874591.
- [18] ESTRADA-VARGAS, A. P.—LÓPEZ-MELLADO, E.—LESAGE, J.-J.: A Comparative Analysis of Recent Identification Approaches for Discrete-Event Systems. Mathematical Problems in Engineering, Hindawi, Vol. 2010, 2010, Article ID 453254, 21 pp.
- [19] FENG, L.—WONHAM, W. M.: Supervisory Control Architecture for Discrete-Event Systems. IEEE Transactions on Automatic Control, Vol. 53, 2008, No. 6, pp. 1449–1461.
- [20] FRIEDLAND, B.: Control System Design. McGraw Hill, 1986.
- [21] GHAFFARI, A.—REZG, N.—XIE, X.: Design of a Live and Maximally Permissive Petri Net Controller Using the Theory of Regions. IEEE Transactions on Robotics and Automation, Vol. 19, 2003, No. 1, pp. 137–141, doi: 10.1109/TRA.2002.807555.

- [22] GIUA, A.—SEATZU, C.: Observability of Place/Transition Nets. *IEEE Transactions on Automatic Control*, Vol. 47, 2002, No. 9, pp. 1424–1437.
- [23] GIUA, A.—CORONA, D.—SEATZU, C.: State Estimation of λ -free Labeled Petri Nets with Contact-Free Non-Deterministic Transitions. *Discrete Event Dynamic Systems: Theory and Applications*, Vol. 15, No. 1, 2005, pp. 85–108, doi: 10.1007/s10626-005-5239-4.
- [24] GIUA, A.—SEATZU, C.: A System Theory View of Petri Nets. In: Bonivento, C., Marconi, L., Rossi, C., Isidori, A. (Eds.): *Advances in Control Theory and Applications*. Springer, Berlin, Heidelberg, Lecture Notes in Control and Information Sciences, Vol. 353, 2007, pp. 99–127.
- [25] GIUA, A.—SEATZU, C.: Observability Properties of Petri Nets. *Proceedings 39th IEEE Conference on Decision and Control*, Sydney, Australia, Vol. 3, 2000, pp. 2676–2681, doi: 10.1109/CDC.2000.914209.
- [26] GIUA, A.—SEATZU, C.: Identification of Free-Labeled Petri Nets via Integer Programming. *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, December 12–15, 2005, pp. 7639–7644.
- [27] HIRAISHI, K.—ČAPKOVIČ, F.: Construction of Rule Base for the Control of Discrete Event Dynamic Systems. Research Report IS-RR-2002-001, ISSN 0918-7553, Japan Advanced Institute of Science and Technology (JAIST), Kanazawa, Japan, 2002, pp. 1–8.
- [28] HOLLOWAY, L.E.—KHARE, A.S.—GONG, Y.: Computing Bounds for Forbidden State Reachability Functions for Controlled Petri Nets. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, Vol. 34, No. 2, March 2004, pp. 219–228, doi: 10.1109/TSMCA.2003.822279.
- [29] HOLLOWAY, L.E.—KROGH, B.H.—GIUA, A.: A Survey of Petri Net Methods for Controlled Discrete Event Systems. *Discrete Event Dynamic Systems: Theory and Applications*, Vol. 7, 1997, No. 2, pp. 151–190.
- [30] HOLLOWAY, L.E.—KROGH, B.H.: Controlled Petri Nets: A Tutorial Survey. In: Cohen, G., Quadrat, J. P. (Eds.): *Proceedings of 11th International Conference on Analysis and Optimization of Systems Discrete Event Systems*. Springer, Berlin, Heidelberg, Lecture Notes in Control and Information Sciences, Vol. 199, 1994, pp. 158–168, doi: 10.1007/BFb0033544.
- [31] HOLLOWAY, L. E.—GUAN, X.—ZHANG, L.: A Generalization of State Avoidance Policies for Controlled Petri Nets. *IEEE Transactions on Automatic Control*, Vol. 41, 1996, No. 6, pp. 804–816, doi: 10.1109/9.506233.
- [32] ICHIKAWA, A.—HIRAISHI, K.: Analysis and Control of Discrete Event Systems Represented by Petri Nets. In: Varaiya, P., Kurzhanski, A. B. (Eds.): *Discrete Event Systems: Models and Applications*. Springer, Berlin, Heidelberg, Lecture Notes in Control and Information Sciences, Vol. 103, 1988, pp. 115–134.
- [33] IORDACHE, M.—ANTSACLIS, P.: *Supervisory Control of Concurrent Systems: A Petri Net Structural Approach*. Birkhäuser, 2006.

- [34] IORDACHE, M.—ANTSAKLIS, P.: Supervision Based on Place Invariants: A Survey. *Discrete Event Dynamic Systems*, Vol. 16, 2006, No. 4, pp. 451–492, doi: 10.1007/s10626-006-0021-9.
- [35] JANTZEN M.: Language Theory of Petri Nets. In: Brauer, W., Reisig, W., Rozenberg, G. (Eds.): *Petri Nets: Central Models and Their Properties (ACPN 1986)*. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 254, 1987, pp. 397–412, doi: 10.1007/978-3-540-47919-2_15.
- [36] KROGH, B. H.: Controlled Petri Nets and Maximally Permissive Feedback Logic. *Proceedings 25th Annual Allerton Conference on Communication Control, and Computing*, University of Illinois, Urbana, 1987, pp. 317–326.
- [37] KUNIMUCHI, Y.: Algebraic Properties of Petri Net Languages and Codes. Ph.D. Thesis, University of Debrecen, Faculty of Informatics, Debrecen, Hungary, 2009, 89 pp.
- [38] KWAKERNAAK, H.—SIVAN, R.: *Linear Optimal Control Systems*. John Wiley & Sons Inc. (Wiley-Interscience), New York-Chichester-Brisbane-Toronto, 1972.
- [39] LUTZ-LEY, A.—LÓPEZ MELLADO, E.: Synthesis of Fault Recovery Sequences in a Class of Controlled Discrete Event Systems Modelled with Petri Nets. *Procedia Technology*, Vol. 7, 2013, pp. 257–264, doi: 10.1016/j.protcy.2013.04.032.
- [40] MANDRIOLI, D.: A Note on Petri Net Languages. *Information and Control*, Vol. 34, 1977, No. 2, pp. 169–171, doi: 10.1016/S0019-9958(77)80013-2.
- [41] MEDA-CAMPAÑA, M. E.—LÓPEZ-MELLADO, E.: Incremental Synthesis of Petri Net Models for Identification of Discrete Event Systems. *Proceedings 41st IEEE Conference on Decision and Control*, Las Vegas, Nevada, USA, December 2002, pp. 805–810.
- [42] MEDA-CAMPAÑA, M. E.—LÓPEZ-MELLADO, E.: Required Event Sequences for Identification of Discrete Event Systems. *Proceedings 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, USA, December 2003, pp. 3778–3783.
- [43] MOODY, J. O.—ANTSAKLIS, P. J.: Petri Net Supervisors for DES with Uncontrollable and Unobservable Transitions. *IEEE Transactions on Automatic Control*, Vol. 45, No. 3, March 2000, pp. 462–476, doi: 10.1109/9.847725.
- [44] MOODY, J. O.—ANTSAKLIS, P. J.: Uncontrollable and Unobservable Transitions. In: Moody, J. O., Antsaklis, P. J. (Eds.): *Supervisory Control of Discrete Event Systems Using Petri Nets*. Chapter 4. Springer, Boston, The International Series on Discrete Event Dynamic Systems, Vol. 8, 1998, pp. 33–50.
- [45] YAMALIDOU, K.—MOODY, J.—LEMMON, M. D.—ANTSAKLIS, P. J.: Feedback Control of Petri Nets Based on Place Invariants. *Proceedings of the 33rd IEEE Conference on Decision and Control*, Lake Buena Vista, FL, December 14–16, 1994, pp. 3104–3109.
- [46] YAMALIDOU, K.—MOODY, J.—LEMMON, M. D.—ANTSAKLIS, P. J.: Feedback Control of Petri Nets Based on Place Invariants. *Automatica*, Vol. 32, No. 1, 1996, pp. 15–28, doi: 10.1016/0005-1098(95)00103-4.
- [47] MURATA, T.: Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE*, Vol. 77, 1989, No. 4, pp. 541–580, doi: 10.1109/5.24143.
- [48] OGATA, K.: *Modern Control Engineering*. 3rd Edition, Prentice Hall, 1999.

- [49] PETERSON, J. L.: *Petri Net Theory and the Modeling of Systems*. Prentice-Hall, Englewood Cliffs, NJ, 1981.
- [50] RAMADGE, J. G.—WONHAM, W. M.: The Control of Discrete Event Systems. *Proceedings of the IEEE*, Vol. 77, 1989, No. 1, pp. 81–98, doi: 10.1109/5.21072.
- [51] RAMÍREZ—TREVIÑO, A.—RIVERA-RANGEL, I.—LÓPEZ-MELLADO, E.: Observer Design for Discrete Event Systems Modeled by Interpreted Petri Nets. *IEEE International Conference on Robotics and Automation (ICRA)*, 2000.
- [52] RAMÍREZ—TREVIÑO, A.—RIVERA-RANGEL, I.—LÓPEZ-MELLADO, E.: Observability of Discrete Event Systems Modeled by Interpreted Petri Nets. *IEEE Transactions on Robotics and Automation*, Vol. 19, No. 4, August 2003, pp. 557–565.
- [53] RAMÍREZ—TREVIÑO, A.—RUIZ-BELTRÁN, E.—RIVERA-RANGEL, I.—LÓPEZ-MELLADO, E.: Online Fault Diagnosis of Discrete Event Systems. A Petri Net-Based Approach. *IEEE Transactions on Automation Science and Engineering*, Vol. 4, 2007, No. 1, pp. 31–39.
- [54] RIVERA-RANGEL, I.—RAMÍREZ-TREVIÑO, A.—AGUIRRE-SALAS, L. I.—RUIZ-LÉON, J.: Geometrical Characterization of Observability in Interpreted Petri Nets. *Kybernetika*, Vol. 41, 2005, No. 5, pp. 553–574.
- [55] RU, Y.—HADJICOSTIS, C. N.: State Estimation in Discrete Event Systems Modeled by Labeled Petri Nets. *Proceedings of the 45th IEEE Conference on Decision and Control*, San Diego, CA, USA, 2006, pp. 6022–6027, doi: 10.1109/CDC.2006.377126.
- [56] RU, Y.—HADJICOSTIS, C. N.: Sensor Selection for Structural Observability in Discrete Event Systems Modeled by Petri Nets. *IEEE Transactions on Automatic Control*, Vol. 55, 2010, No. 8, pp. 1751–1764.
- [57] SANTOYO-SANCHEZ, A.—PÉREZ-MARTÍNEZ, M. A.—DE JESÚS-VELÁSQUEZ, C.—AGUIRRE-SALAS, L. I.—ALVAREZ-UREÑA, M. A.: Modeling Methodology for NPC’s Using Interpreted Petri Nets and Feedback Control. *Proceedings of the 2010 7th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE 2010)*, Tuxtla Gutiérrez, Chiapas, Mexico, September 8–10, 2010, IEEE Press, 2010, pp. 369–374.
- [58] SANTOYO-SANCHEZ, A.—RAMÍREZ-TREVIÑO, A.—DE JESÚS-VELÁSQUEZ, C.—AGUIRRE-SALAS, L. I.: Step State-Feedback Supervisory Control of Discrete Event Systems Using Interpreted Petri Nets. *Proceedings of the 2008 IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2008)*, Hamburg, Germany, 2008, pp. 926–933.
- [59] SESSEGO, F.—GIUA, A.—SEATZU, C.: HYPENS: A Matlab Tool for Timed Discrete, Continuous and Hybrid Petri Nets. In: van Hee, K. M., Valk, R. (Eds.): *Applications and Theory of Petri Nets (PETRI NETS 2008)*. Springer, Berlin, Heidelberg, *Lecture Notes in Computer Science*, Vol. 5062, 2008, pp. 419–428.
- [60] TAPIA-FLORES, T.—LÓPEZ-MELLADO, E.—ESTRADA-VARGAS, A. P.—LESAGE, J.-J.: Petri Net Discovery of Discrete Event Processes by Computing T-Invariants. *Proceedings of 19th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA ’14)*, Barcelona, Spain, 2014, Art. No. 345, 8 pp.

- [61] THISTLE, J. G.: Supervisory Control of Discrete Event Systems. *Mathematical and Computer Modelling*, Vol. 23, 1996, No. 11-12, pp. 25–53, doi: 10.1016/0895-7177(96)00063-5.
- [62] VAN GLABBEEK, R. J.: The Individual and Collective Token Interpretations of Petri Nets. In: Abadi, M., de Alfaro, L. (Eds.): *Concurrency Theory (CONCUR 2005)*. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 3653, 2005, pp. 323–337, doi: 10.1007/11539452_26.
- [63] VALK, R.—VIDAL-NAQUET, G.: Petri Nets and Regular Languages. *Journal of Computer and System Sciences*, Vol. 23, 1981, No. 3, pp. 299–325.
- [64] WANG, J.: *Timed Petri Nets*. Kluwer Academic Publishers, Boston, MA, USA, 1998, doi: 10.1007/978-1-4615-5537-7.
- [65] ZHANG, L.—HOLLOWAY, L. E.: Forbidden State Avoidance in Controlled Petri Nets under Partial Observation. *Proceedings of 33rd Allerton Conference*, Monticello, Illinois, 1995, pp. 146–155.
- [66] ZHAO, M.—UZAM, M.—HOU, Y. F.: Near-Optimal Supervisory Control of Flexible Manufacturing Systems Using Divide-and-Conquer Iterative Method. *Advances in Mechanical Engineering*, Vol. 8, 2016, No. 3, pp. 1–17.



František ČAPKOVIČ received his Master degree in 1972 from the Faculty of Electrical Engineering of the Slovak Technical University, Bratislava, Slovakia. Since 1972 he has been working at the Slovak Academy of Sciences (SAS), Bratislava, namely in 1972–1991 at the Institute of Technical Cybernetics, in 1991–2001 at the Institute of Control Theory and Robotics and in 2001 till now at the Institute of Informatics. In 1980 he received his Ph.D. from SAS. In 1998 he was appointed the Associate Professor. He works in the area of modelling, analysing and intelligent control of Discrete-Event Systems (DES) and Hybrid

Systems. He is the author of more than 240 publications.