Computing and Informatics, Vol. 37, 2018, 781–814, doi: 10.4149/cai_2018_4_781

EQUAL: ENERGY AND QOS AWARE RESOURCE ALLOCATION APPROACH FOR CLOUDS

Ashok Kumar, Rajesh Kumar, Anju Sharma

Department of Computer Science and Engineering
Thapar University
Patiala-147004, India
e-mail: `ashok.khunger@gmail.com, {rakumar, anju.sharma}@thapar.edu`

Abstract. The popularity of cloud computing is increasing by leaps and bounds. To cope with resource demands of increasing number of cloud users, the cloud market players establish large sized data centers. The huge energy consumption by the data centers and liability of fulfilling Quality of Service (QoS) requirements of the end users have made resource allocation a challenging task. In this paper, energy and QoS aware resource allocation approach which employs Antlion optimization for allocation of resources to virtual machines (VMs) is proposed. It can operate in three modes, namely power aware, performance aware, and balanced mode. The proposed approach enhances energy efficiency of the cloud infrastructure by improving the utilization of resources while fulfilling QoS requirements of the end users. The proposed approach is implemented in CloudSim. The simulation results have shown improvement in QoS and energy efficiency of the cloud.

Keywords: Energy efficiency, resource utilization, resource allocation, antlion optimization, quality of service

1 INTRODUCTION

Cloud computing delivers Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [1] on pay per usage basis. These services are provided through shared pool of configurable computing resources such as networks, servers, storage, and applications, which are rapidly provisioned and released on demand. The liability of resource management lies with the service provider. This effortless computing paradigm (from the point of view of cloud users) has resulted in dramatic increase in number of cloud users. To cope with resource demands of increasing number of users, cloud market players such as Amazon, Microsoft, Google, Gogrid, Flexiant, etc., establish large sized data centers. Due to this cloud computing infrastructure grows approximately 36 % every year, and is estimated to touch $ 19.5 billion market by 2016 [2]. Further, data centers consume huge amount of energy. It has increased threefold between 2007 and 2012 [3]. In 2013, data centers in U.S consumed an estimated 91 billion kWh of electricity, which was enough to power entire New York for two years [5]. Moreover, average resource utilization of data centers is approximately 15–20 % [4, 6]. The most part of the energy consumption of a data center is wasted due to under utilization of resources because even an idle resource consumes 50 % of its maximum power utilization [7]. This means low utilization decreases the energy efficiency of the resources.

On the other hand, performance may degrade if the VMs executing the tasks are not allocated resources as per requirements [8], and hence may cause Service Level Agreement (SLA) violations. Consequences of performance degradation can be critical. One direct consequence may be losing users. For example, 100 ms delay results 1 % decrease in sales of Amazon, and Google observed 20 % decrease in traffic with 0.5 seconds delay in search page generation [9].

The huge amount of energy consumption and liability to fulfill QoS requirements demand for efficient allocation of resources. Therefore, a novel Energy and QoS aware resource allocation approach Using AntLion optimization (EQUAL) is proposed. EQUAL allocates proportion of computing capability of a resource to a VM. It can be managed to operate in power, performance, or balanced mode. Further, the VMs encapsulate time constrained users' tasks which are distributed among them in a round robin fashion. The major contributions of the proposed resource allocation approach are:

1. A novel energy and QoS aware resource allocation approach is proposed.
2. Antlion optimization is employed to group VMs on lesser number of physical resources in order to optimize energy consumption.
3. The proposed approach can be tuned to power aware, performance aware, or balanced mode.
4. EQUAL is implemented in CloudSim and tested with VMs/tasks having different processing requirements.
5. Up to 15.04 % reduction in energy consumption is achieved.

The rest of the paper is organized as follows: Related work is presented in Section 2. In Section 3, the proposed resource allocation approach, power model, problem definition, and antlion optimization are presented. Resource provisioning using antlion optimization is elaborated in Section 4. Performance evaluation and comparative analysis is given in Section 5. Section 6 discusses conclusion and future scope for expansion.

2 RELATED WORK

Number of researchers have done significant work on energy efficiency and QoS aware resource allocation. They considered various application domains such as high-performance scientific computing, multi-tier web applications, or workflow applications, and used variety of approaches to obtain better results. In this section, an extensive survey on various resource allocation related approaches and state of the art techniques is conducted.

2.1 Resource Allocation with Traditional Algorithms

This subsection discusses various resource allocation approaches existing in literature that are not based on nature inspired meta heuristics. Quan et al. [10] presented resource allocation framework that improves utilization and hence the energy efficiency of the cloud infrastructure. Lee et al. [11] proposed performance based resource allocation strategy for green cloud. Each physical machine in the data center is assigned a performance value based on its CPU processing speed, number of cores and memory capacity. The physical machines resources are provisioned in the order of their performance value. Quarati et al. [12] proposed resource allocation for hybrid cloud with the objective to maximize broker's revenue and user satisfaction. The requested service is allocated resources on either private or public cloud depending on reserved quota of private cloud resources. Further, the service is run on a physical machine (PM) having maximum availability of free resources. Resource allocation is modeled as bin packing problem in order to improve utilization of resources [13, 14]. The PMs are treated as bins and VMs are assumed the items to be packed in. Bobroff et al. [13] presented an approach which periodically runs an offline bin packing algorithm to calculate VMs to PMs mapping. The approach eliminates hot-spots and minimizes the number of PMs in use. Takeda and Takemura [15] proposed ranking of physical servers for consolidation and VM placement. Servers with higher priorities are considered more reliable than the servers with lower priority value. Higher priorities are assigned to newly installed servers. The VMs are consolidated on more reliable servers to conserve energy. Son et al. [16] introduced workload and location-aware resource allocation scheme (WLARA) with automated SLA negotiation mechanism but they have not considered energy efficiency. Chieu et al. [17] proposed an architecture for dynamic allocation of resources to workloads, based on threshold number of active sessions. The proposed work is capable of maintaining higher resource utilization, thus reducing infrastructure and management costs. Wu et al. [18] advocated SLA based provisioning technique which reduces resource cost and SLA violations. The management of customer requests, mapping them with resources is defined along with the supervision of different types of workloads by considering QoS such as execution time. Raycroft et al. [19] analyzed the effect of global virtual machine allocation policy on energy consumption. Kim et al. [20] proposed energy credit scheduler which allocates resources to a VM based on its energy credit. The resources allocated to VM are preempted when its energy credit vanishes. Xu and Fortes [56] proposed multi-objective VM allocation algorithm. The authors have taken CPU, and memory parameters for VMs and have claimed reduction in power consumption, thermal dissipation costs, and resource wastage. Wu et al. [22] presented a technique to increase the utilization and efficiency of hardware equipment. Dynamic voltage frequency scaling is employed to decrease energy consumption for executing jobs without sacrificing its performance. The authors in [15, 23, 24, 25] offered approaches to conserve energy and maximize resource utilization without affecting the performance of the system. They used energy-conscious consolidation heuristics to improve utilization and conserve energy. Beloglazov et al. [26] proposed power efficient and QoS aware resource allocation heuristics. An algorithm for minimization of number of VM migrations is also proposed. Upper and lower threshold utilization levels are used to detect overloaded and underloaded machines. When the resource utilization of a particular server falls below the lower threshold value, all the VMs running on the machine are shifted to some other machine. If utilization of a machine is above upper threshold, one or more VMs are shifted to other machines to keep the utilization between the threshold values. They proposed algorithms for single core machines. Kusic et al. [27] proposed performance and power efficient resource-management approach based on look-ahead control method for virtualized heterogeneous environments. Prediction is employed for dynamic reallocation of resources. Gao et al. [28] presented a dynamic resource management approach for energy saving and service level agreements fulfillment. CPU speed is considered as the bottleneck of performance. Dynamic voltage/frequency scaling and server consolidation are used for energy saving.

2.2 Resource Allocation Based on Nature-Inspired Metaheuristics

Feller et al. [29] presented a multi-dimensional ant colony optimization based job consolidation algorithm. The algorithm uses resource utilization history to predict future resource demands and dynamically overbooks the resources. The authors tested the proposed algorithm on homogeneous PMs. Gao et al. [30] proposed multi-objective ant colony system algorithm for virtual machine placement that minimizes total resource wastage and power consumption. Ant colony optimization technique for assigning real-time tasks to heterogeneous processors is proposed by Chen et al. [31]. Local search technique is applied to improve energy efficiency of the feasible assignment solution generated by the proposed assignment algorithm. Huang et al. [32] presented genetic algorithm based adaptive sub-optimal resource management scheme to estimate number of VMs required to provide desired level of service. Kansal and Chana [33] suggested a model based on artificial bee colony to improve utilization of resources. The model supports energy efficient allocations of tasks to resources and minimizing execution time of applications. Chimakurthi and Madhu Kumar [34] offered ant colony based adaptive resource allocation framework for hosting applications with throughput and response time as QoS requirements. Further, it supports reduction in power consumption of data center resources. Hu et al. [35] expressed problematic issue of VM placement as a multi-objective optimization problem. An improved ant colony system algorithm is offered for the data centers to reduce total resource wastage and energy consumption. Liu et al. [36] gives ant colony optimization based solution for VM placement on physical servers in order to decrease the number of active physical servers. Portaluri et al. [37] presented genetic algorithm based trade-off solutions between tasks completion time and system power consumption. The system allocates resources to independent tasks on homogeneous single-core resources. Xiong and Xu [38] presented a multiresource energy efficiency VM allocation model based on particle swarm optimization for energy efficiency of cloud data center. Total Euclidean distance is used as a fitness function to keep balance between resource utilization and energy consumption. This algorithm avoids falling into local optimal solution, which is common in traditional heuristic algorithms. Kumar and Raza [39] presented particle swarm optimization based strategy for VM allocation to physical machines in order to reduce total energy consumption and resource wastage. Dashti and Rahmani [40] proposed modified particle swarm optimization solution to guarantee quality of service of users' tasks, and reduce energy efficiency. Response time and deadline are considered as QoS parameters. This approach reallocates virtual machines from the overloaded host, and dynamically consolidates under-loaded hosts for power saving. Kansal and Chana [41] used firefly optimization to enhance energy efficiency of the cloud without sacrificing the performance. The authors used VM migration to enhance energy efficiency. Kumar et al. [42] presented two level ant colony based resource allocation approach to minimize total cost of execution, total execution time and total energy consumption. They used server consolidation and dynamic performance scaling to conserve energy. Kumar et al. [43] proposed power and performance aware resource allocation. They improved performance and energy efficiency of the cloud employing cuckoo optimization.

2.3 Motivation

Resource allocation in cloud computing can be accomplished using either traditional deterministic algorithms [11, 12, 13, 14, 15, 18] or metaheuristic algorithms [29, 30, 31, 32, 33, 41, 42, 43]. Deterministic algorithms suffer from local optima entrapment, i.e., they got struck in local solutions and consequently fail to find the true global optimal solution. Moreover, resource allocation using deterministic algorithms is NP-hard. So in the recent years metaheuristic algorithms have been employed for efficient allocation of resources [29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43]. The fundamental characteristic of metaheuristic algorithms is stochastic operators which are used for finding optimal solution in the search space. Stochastic operators help them to escape local solutions. Due to their random behavior, they are able to obtain different solutions in each run. They start with some random solutions, called candidate solutions, of the problem at hand, and then improve the candidate solutions iteratively. Their solution finding process is completely independent from the problem. We get motivation from the way metaheuristic algorithm operates to find optimal solution of the problem. When a metaheuristic algorithm gets trapped in local solution, stochastic operators make random changes in the solution and eventually help in escaping from local optimal solution. In a nutshell, all metaheuristic algorithms follow a general and common framework, in which they improve a set of randomly created solutions iteratively. The algorithms differ in the method of improving the initial random solutions. We preferred antlion optimization over other existing metaheuristic algorithms used [29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43] for resource allocation because it provides very competitive results in terms of improved exploration, local optima avoidance, exploitation, and convergence [49]. Further, our work significantly differs from the other metaheuristic based resource allocation approaches in the area of cloud computing as it can be tuned to operate in power aware, performance aware, or balanced mode.

The detailed working of the proposed resource allocation approach is presented in the next section.

3 ENERGY AND QOS AWARE RESOURCE ALLOCATION

Resource allocation is a process of provisioning resources to VMs. Resources are allocated to VMs with the aim to minimize energy consumption while satisfying QoS requirements. In this work, we used antlion optimization for energy and QoS aware allocation of resources to VMs. The proposed approach can be operated in power, performance, and balanced mode. In power aware mode, a VM is allocated to a resource that causes minimum increase in energy consumption. Whereas in performance mode, a VM is allocated to a resource that has maximum available computational capacity. In balanced mode, power and performance are given equal weightage while allocating resources. The VMs encapsulate users' tasks which are scheduled in Earliest Deadline First (EDF) order. Each task has a deadline, a point of time, by which execution of the task should finish. If deadline of a task is missed then SLA violation is said to have occurred.

In the proposed work, the following assumptions are taken into consideration:

1. Each task is independent of other tasks.
2. A VM can be executed on a server with lesser free available resources than required but at the cost of reduced performance.
3. Resources can be switched to sleep mode to conserve energy.
4. Energy consumption of a resource in sleep mode is negligible.

The major components of energy and QoS aware resource allocation approach are shown in Figure 1. The *bag of tasks* is the collection of time constrained tasks submitted by the end users. The detailed information about each resource like a type of resource and its computational capability is provided by *resource description* component. The *resource allocation* component refers *resource description* component while allocating resources to VMs. The resources are allocated to VMs employing antlion optimization. Once the resources are provisioned to VMs, *resource scheduler* manages the scheduling of VMs on the provisioned resources. Utilization of each

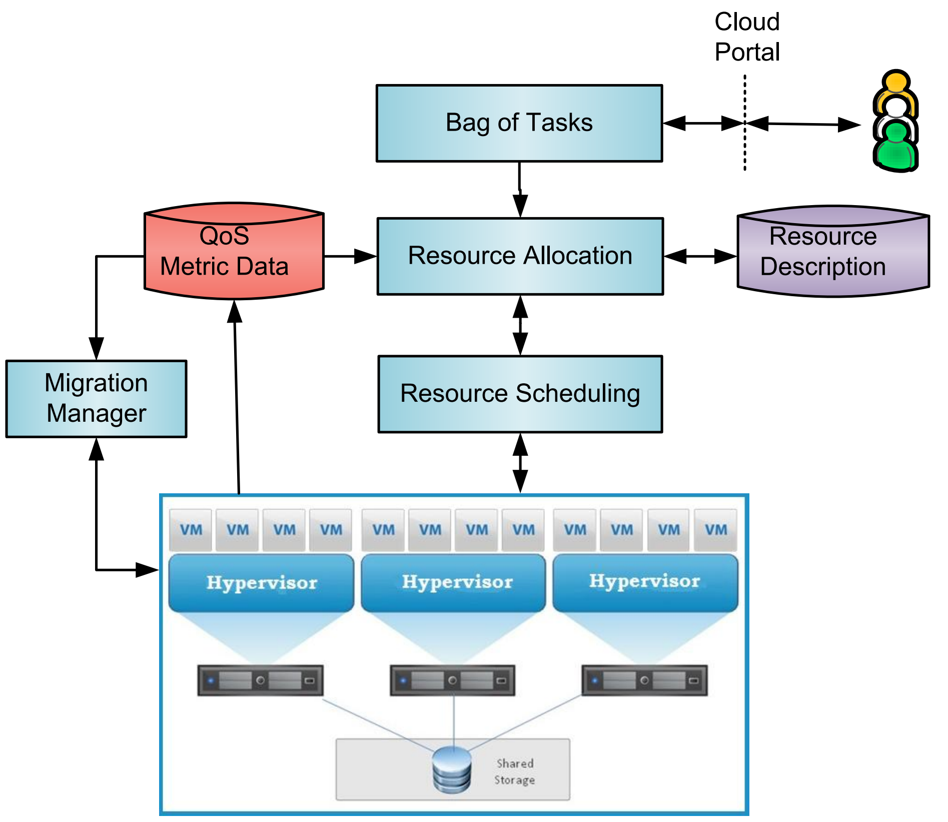


Figure 1. Energy and QoS aware resource allocation

resource (server) is monitored at regular interval of time, and saved in *QoS metric database*. Resource utilization information is used by *migration manager* to perform server consolidation. It is invoked after a fixed interval of time. A VM is selected for migration based on interquartile range (IQR) [44] of the utilization history data. VM migration is performed in two cases. First, when a resource is under-loaded, i.e., the utilization of the resource is below the lower green threshold (LGT) limit. In this case, all the VMs running on the under-loaded resource are shifted to other resources and the resulted idle resource is switched to low power (sleep) mode to conserve energy. Second, when the resource is over-loaded, i.e., the utilization of the resource is above the upper green threshold (UGT). In this case, one of the VMs running over the resource is migrated to other resource to bring the resource utilization below UGT.

3.1 Power Model

Generally, the resources have different run-time power consumption because of their heterogeneous processor architectures, processing speeds, hardware features, etc. Power consumption of a resource is given by Equation (1):

$$P_{total} = P_{dynamic} + P_{static}, \tag{1}$$

static power consumption (SPC), P_{static}, is due to leakage current and is independent of clock frequency and usage scenario. SPC can be reduced by switching idle resources to sleep mode [45]. However, dynamic power consumption (DPC), $P_{dynamic}$, is due to circuit activity, and it depends on resource utilization. DPC of a resource increases linearly with its utilization [26], and is given by Equation (2).

$$P = P_{idle} + (P_{max} - P_{idle})U \tag{2}$$

where P_{idle} is the power consumption when the resource is idle, P_{max} is the power consumption at 100 % utilization, and P is the power consumption of the resource at utilization $U \in [0, 1]$.

3.2 Problem Definition

Cloud computing leverages virtualization such as XEN [46], KVM [47], or VMWare [48] to support execution of multiple VMs on a single physical resource. Each VM has some resource demands such as CPU, number of processing cores, memory, network bandwidth, etc. If a VM is not allocated the required resource capacity then it processes encapsulated tasks at slower speed thereby elongating the tasks' completion time. Consequently, some of the tasks may miss the deadline. When deadline of a task is not observed, it is considered as SLA violation.

Suppose a set $J = \{J_i | 1 \leq i \leq n\}$ of n tasks, and each task J_i is associated with deadline time d_i and processing volume w_i. The processing volume is the amount of processing in millions of instructions (MI) that must be carried out to finish the task. Tasks are distributed among V number of VMs. Further, $S = \{S_j | 1 \leq j \leq m\}$ is a set of m resources. The problem is to allocate resources to VMs in such a way to minimize the number of active resources and their energy consumption while observing QoS requirements (deadline) of the end users' task. We considered only processing requirements while allocating resources because CPU is accounted for major part of energy consumption by a physical machine [45]. The allocation of resources to VMs is NP-hard. Therefore, antlion metaheuristic optimization is employed for allocation of resources to VMs. The proposed approach groups VMs over a small number of resources and thus allows turning off those resources that are not in use. Energy efficiency and QoS are considered while allocating resources to the VMs. The suitability of resource r for VM j is determined from fitness function $f_{j,r}$, given by Equation (3), which helps in fulfilling the following goals:

1. Allocation of a VM to a resource that results in minimum increase in energy consumption of the cloud.
2. Provisioning VMs on reduced number of resources.
3. Performance requirements are taken into consideration while allocating resources.

$$f_{j,r} = \frac{\left[\frac{\Re_r^a}{\Re_j^d}\right]^{\theta}}{\left[\gamma \triangle E_{j,r} + (1-\gamma)\kappa_r \underbrace{(1 - \sum_{i \in S, i \neq j} \Re_{i,r})}_{y}\right]^{1-\theta}}, \quad \forall r \tag{3}$$

where $\Re_r^a$ is available processing power of resource r, $\Re_j^d$ is processing demand of VM j. $\triangle E_{j,r}$ is energy contribution of VM j on resource r, κ_r is energy affinity, $\Re_{i,r}$ is fraction of processing power allocated to VM i on resource r, and $0 \leq \gamma \leq 1$ is a constant. $0 \leq \theta \leq 1$ is trade off between performance and energy. By changing the value of θ, EQUAL can be operated in one of the three modes, namely:

1. power aware,
2. performance aware, or
3. balanced mode.

EQUAL operates in power aware mode when θ is set to 0. In this mode, the increase in energy consumption of the resource is considered while allocating VMs. Thus, a VM is allocated to a resource which results in minimum increase in energy consumption. When θ is set to 1, EQUAL operates in performance aware mode, and thus allocates a VM to a resource which has maximum available computing power at disposal. In balanced mode, when θ is 0.5, EQUAL maintains the balance between power and performance while allocating resources to VMs. EQUAL inclines towards power aware allocation if $\theta < 0.5$, and towards performance aware allocation if $\theta > 0.5$.

In this work, $\frac{\Re_r^a}{\Re_j^d}$ is called performance affinity which is desired to be greater than or equal to 1. When performance affinity value is less than one, VM would not get sufficient resource and would therefore slow down the execution of encapsulated tasks. Energy affinity, κ_r, is the minimum energy consumption of the resource, i.e. energy consumption in idle state. Therefore, EQUAL gives preference to resources having lesser energy consumption in idle state. A resource having low power consumption in the idle state has higher value of fitness function and is therefore given preference over others while resource provisioning. The term y allows to group VMs on lesser number of resources. The value of term y for a resource r decreases as more and more VMs are deployed on it. Consequently, fitness function of the resource r increases and thereby enables grouping of VMs on lesser number of resources.

When θ is set to 1, power consumption of a resource does not contribute in finding suitable resource for the candidate VM. The machine having maximum available resource capacity is given preference over the others and the allocation approach reduces to worst-fit decreasing. In order to consider power affinity while searching for the best resource and to pack VMs on lesser number of machines, denominator of the Equation (3) should not evaluate to 1. This is possible only if θ is assigned value smaller than one. Thus, we used $\theta = 0.95$ to bias EQUAL towards performance aware allocation while taking advantage of its VM packing capability in order to save power consumption. However, when θ is set to 0, EQUAL reduces to best-fit approach which strives to pack VMs on lesser number of resources. The machine which is hosting more VMs and has low energy affinity is given preference over the others. The processing power of the resources is not taken into consideration. In order to consider computing capability of the resource in power aware mode θ should be assigned small positive value other than 0. In this work, we used $\theta = 0.05$ for power aware allocation in order to consider computing power of a resource in addition to its power consumption.

3.3 Application and Infrastructure Model

Cloud computing is suitable platform for deadline constrained scientific applications in areas such as astronomy, bioinformatics, and physics [57]. In this work, we proposed resource allocation approach, EQUAL, that can be used for deadline constrained applications such as Montage, which is used for generation of sky mosaics; Cyber-Shake, used for earthquake risk characterization; LIGO, used for detection of gravitational waves and SIPHT, used in bioinformatics. All these four applications are characterized by Juve et al. [58]. Scientific application (task) consists of thousands of sub-tasks, and can take benefit of large-scale infrastructure of cloud computing. Scientific application has soft deadline which is required to be accomplished. A soft-deadline does not make the computation useless if the task is not completed in time [59]. A computation begets maximum benefit if deadline is achieved. A scientific application may consist of sub-tasks and may have dependencies between them. Each task i has a deadline d_i and processing volume w_i associated with it. Deadline of a task determines the time to accomplish the execution of the task from the moment it is submitted to EQUAL, which manages the execution of tasks, allocates VMs to them, and schedule their execution in the cloud. EQUAL offers a set of four VM types denoted by set $V = \{A0, A1, A2, A3\}$. Each VM type offers different amount of resources. There is no limit on the number of VMs of each type that can be running at any moment for the execution of tasks. The problem addressed in this work is the execution of tasks latest by deadline time at the smaller possible energy cost. The problem is solved by the efficient allocation of resources using antlion optimization.

3.4 Antlion Optimization

Antlion optimization [49] is proposed by Seyedali Mirjalili in 2015. Antlions belong to the myrmeleontidae family. An antlion larvae makes cone-shaped pit and hides itself underneath the pit waiting for prey to be trapped in. The size of the pit is proportional to the level of hunger. When an insect is trapped in the pit, the antlion tries to catch it by intelligently throwing sand towards to edge of the pit to slide the prey into the bottom of the pit. Once the insect is caught, it is pulled under the sand and then consumed.

The following are the reasons for selecting antlion optimization for resource allocation:

1. Random selection of antlions guarantees the exploration of search space.
2. Adaptive shrinking boundaries of antlions' traps guarantee the exploitation of search space.
3. The promising regions of search space are guided by antlions.
4. It is a gradient-free algorithm and considers the problem as a black box.

4 RESOURCE ALLOCATION USING ANTLION OPTIMIZATION

The antlion optimization algorithm, which mimics behavior of antlions and ants, is used for discovering resource for a VM. The objective is to find a resource that fulfills not only the resource requirements of the VM but also causes a minimum increase in energy consumption. Each antlion provides initial guess of the resource, and then a resource better than the initial guess is searched through random walk of an ant around the antlion. When a better resource is found, the location of the antlion is replaced with the location of the corresponding ant.

The location of ant and antlion, each representing a resource, are saved in matrix M^a and M^{al}, respectively. Location of i^{th} ant, W_i^t, and j^{th} antlion, V_j^t, at t^{th} iteration are represented by i^{th} and j^{th} rows of matrices M^a and M^{al}, respectively. In each iteration, location of an ant is updated to reflect its latest position. The fitness value of an ant, which determines goodness of a solution, is also updated in each iteration. The fitness value of an ant/antlion is evaluated from Equation (3). When the fitness value of an ant becomes greater than the fitness value of the antlion, the location and fitness value of the antlion are replaced with the location and fitness value of the corresponding ant. The fitness values of ants and antlions are saved in matrix M^{fa} and M^{fal}, respectively.

Random walk of an ant i in the search space is modeled by Levy Flight (LF) [50], which can be expressed by Equation (4).

$$W_i^t = W_i^{t-1} + \alpha\ L(s, \lambda) \tag{4}$$

where α is the scaling factor for step size s. Levy exponent, λ, is a constant. $L(s, \lambda)$ is Levy distribution with parameters s and λ. W_i^t is the location of an ant i at t^{th} iteration.

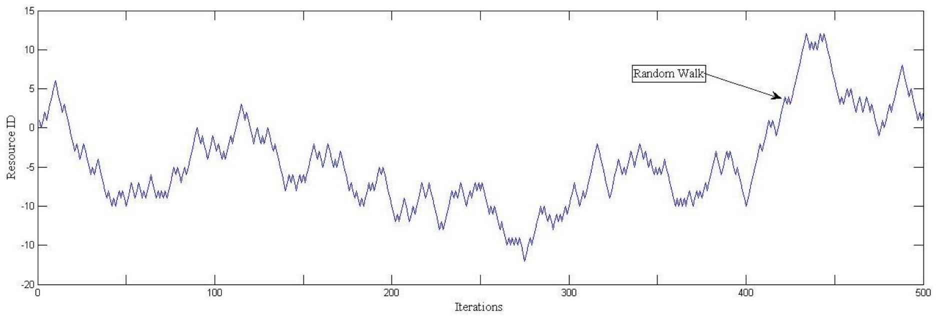


Figure 2. Random walk of an ant

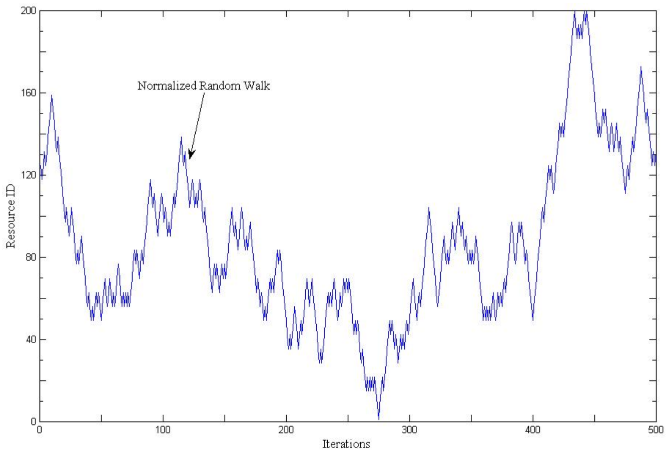


Figure 3. Normalized random walk of an ant

Figure 2 shows random walk for an ant generated from Equation (4). Number of iterations is represented along x-axis, whereas, resource ID (identification) is denoted by y-axis. Since search space, consisting of identification of each resource, has a range of permitted values, so max–min normalization, as shown in Equation (5), is applied to keep the random walk within the desired range.

$$W_i^t = \left\lceil \frac{(W_i^t - a_i)(d_i^t - c_i^t)}{b_i - a_i} + c_i^t \right\rceil \tag{5}$$

where a_i, b_i are the minimum and maximum of random walk of i^{th} ant, and c_i^t, d_i^t are the minimum and maximum of search space at t^{th} iteration. Figure 3 shows normalized random walk of the ant generated using Equation (5). Random walk of an ant (shown in Figure 2) is normalized to range of resource identifications used in EQUAL, i.e. from 1 to 200.

Random walks of ants are affected by positions of antlions. An ant is allowed to move around an antlion which is selected using roulette wheel. The range of search space at t^{th} iteration for random walk of an ant i around the antlion j is mathematically modeled by Equations (6) and (7).

$$c_i^t = V_j^t + c^t, \tag{6}$$

$$d_i^t = V_j^t + d^t \tag{7}$$

where c^t and d^t are the minimum and maximum of the search space at t^{th} iteration, c_i^t and d_i^t are the minimum and maximum for i^{th} ant, and V_j^t is the position of the selected j^{th} antlion at t^{th} iteration. In order to find the best resource, values of c^t and d^t are updated in each iteration using Equations (8) and (9), respectively.

$$c^t = \left\lceil \frac{c^t}{I} \right\rceil, \tag{8}$$

$$d^t = \left\lceil \frac{d^t}{I} \right\rceil, \tag{9}$$

here, $I = 10^w \frac{t}{T}$, where t is the current iteration, T is the maximum number of iterations, and w is a constant that can adjust the level of exploitation and is defined on the basis of the current iteration.
As discussed before, the location of an antlion is replaced with the location of corresponding ant when the fitness of a resource referred by an ant becomes greater than the fitness of a resource referred by the antlion. This situation is represented by Equation (10).

$$V_j^t = W_i^t, \text{if } f(W_i^t) > f(V_j^t) \tag{10}$$

where V_j^t is location of j^{th} antlion at t^{th} iteration, and W_i^t is location of i^{th} ant at t^{th} iteration.

EQUAL maintains record of the best resource (solution). The best solution, called elite, is saved in each iteration. The elite solution has the highest fitness value and effects the random walk of each ant (as shown in Equation (11)).

$$W_i^t = \frac{W_i^t + W_e^t}{2} \tag{11}$$

where W_i^t is random walk of ant i around an anlion and W_e^t is random walk of elite e at t^{th} iteration.

The detailed resource provisioning process is given in Algorithm 1. It employs antlion optimization to find the best resource for a VM. The resource search process is repeated until maximum iterations T has elapsed or the elite solution is same for three consecutive iterations.

Algorithm 1 Pseudo code for energy and QoS aware resource allocation using Antlion Optimization

```
Input: Set $V$ of VMs; Set $A$ of ants; Set $L$ of antlions; Set $S$ of resources
Output: VMs-Resources map ($M_{VR}$)
for each VM $v \in V$ do
  Initialize ants' position matrix $M^a$ randomly.
  Initialize antlions' position matrix $M^{al}$ randomly.
  Evaluate suitability of resource referred by each ant $i \in A$ ($f_{v,i}$) and antlion
  $j \in L$ ($f_{v,j}$) for VM $v$ from fitness function (Equation (3)) and store the values
  at $i^{\text{th}}$ and $j^{\text{th}}$ row of matrix $M^{fa}$ and $M^{fal}$, respectively.
  Find an antlion (say $e$) for which value of fitness function (Equation (3)) is
  maximum (say $f_{v,e}$) and call it elite solution.
  set iteration counter $t \leftarrow 1$
  while ($t \leq$ T) and ( until $e$ is same for three consecutive iterations) do
    for each ant $i \in A$ do
      Select an antlion $j$ using Roulette wheel.
      Calculate $c^t$ and $d^t$ using Equations (8) and (9).
      Evaluate $c_i^t$ and $d_i^t$ using Equations (6) and (7) to select range of random
      walk for ant $i$
      Generate random walk for ant $i$ using Equation (4)
      Normalize random walk for ant $i$ using Equation (5)
      Update random walk on ant $i$ using Equation (11).
      Update position vector ($M_i^a$) and fitness value ($M_i^{fa}$) of ant $i$.
      if ($f_{v,i} > f_{v,j}$) then
        set $M_j^{fal} = M_i^{fa}$
        set $M_j^{al} = M_i^a$
      end if
    end for
    Find new elite solution among antlions and assign it to $e$.
    set $t \leftarrow t + 1$
  end while
  Allocate VM $v$ to resource referred by elite solution $e$, and add VM-resource
  pair to map $M_{VR}$
end for
return $M_{VR}$
```

5 PERFORMANCE EVALUATION AND COMPARATIVE ANALYSIS

Type	Processing	PEs	RAM	Storage	BW
1	2 933	4	8	500	10
2	3 067	4	8	500	10
3	2 933	12	12	500	10
3	3 067	12	16	500	10

Processing, processing speed in millions of instructions per second; PEs, number of processing elements; RAM, random access memory in GB; Storage, permanent storage capacity in GB; BW, network bandwidth in gigabits per second.

Table 1. Specification of resources

VM Type	CPU	PEs	RAM	BW
A0	500	1	768	1 000
A1	1 000	1	1 792	1 000
A2	1 500	2	3 584	1 000
A3	2 000	4	7 168	1 000

CPU, processing speed in millions of instructions per second; PEs, number of cores; RAM, random access memory in megabytes; BW, network bandwidth in megabits per second.

Table 2. Specification of virtual machines

A number of cloud simulation tools such as CloudSim [51], CloudAnalyst [52], GreenCloud [53], NetworkCloudSim [54], etc., are available to implement and evaluate a resource allocation approach on large scale, repeatable, and controlled cloud environment. But the proposed approach is implemented in CloudSim because it supports modeling of various cloud entities such as datacenters, servers, virtual machines, and tasks with ease. The proposed resource allocation approach can be implemented easily by extending VM allocation policy of CloudSim.

For performance analysis, EQUAL is compared with Artificial Bee Colony (ABC) [33], Genetic Algorithm (GA) [56], and non-QoS aware resource allocation (NQRA) which is designed by combining round robin and earliest deadline first scheduling approach that allocates resources using best effort approach.

5.1 Experimental Setup

The simulation testbed consists of a datacenter containing 200 resources. The specification of four types of resources used in simulation is as per Table 1. We created equal number of resources of each type in a simulation run. The datacenter models instances of general purpose compute-basic tier of Microsoft Azure [55], and the parameters relevant for the experiments are shown in Table 2. The tasks having diverse CPU and memory requirements are used, and the number of tasks are varied from 200 to 1 000. Further, the tasks are modeled as Cloudlets and their processing requirements are represented in MI. Simulation is repeated forty to fifty times with different number of resources, VMs, and tasks. The different parameters used during simulation are shown in Table 3.

Number of Resources	50–200	
Number of Tasks (Cloudlets)	200–1 000	Varied in every simulation run
Size of tasks	10 000 + (5–30 %) MI	in millions of instructions (MI)
Simulation Span	86 400 s	Simulation time period
Idle Time	10 min.	Time to switch PM to sleep mode
UGT	0.85	Upper Green Threshold limit
LGT	0.20	Lower Green Threshold limit
HT	0.95	Hot-spot Threshold
CT	0.15	Cold-spot Threshold

Table 3. Simulation parameters

5.2 Simulation Results

Case 1: EQUAL in Balanced Mode

EQUAL switches to balanced mode when 0.5 is assigned to θ. In this mode, energy and performance are given equal weightage while allocating resources to VMs. In order to group VMs over minimum number of resources 0.05 is assigned to γ.

Figure 4 shows the comparison of number of resources used by EQUAL in balanced mode (EQUAL-B), NQRA, ABC, and GA for different number of VMs. The number of used resources increases with increase in number of VMs to be deployed. But EQUAL-B uses lesser number of resources than NQRA, ABC, and GA for given number of VMs. It has been observed that EQUAL-B uses approximately 8.68 %, 4.47 %, and 6.84 % lesser number of resources than NQRA, ABC, and GA, respectively.

Figure 5 depicts the comparison of total energy consumption of EQUAL-B, NQRA, ABC, and GA. It is observed that EQUAL-B consumes lesser energy than NQRA, ABC, and GA for given number of VMs. In EQUAL-B, energy consumption of 107.67 kWh is measured for 200 VMs, and it increases to 735.65 kWh for 1 000 VMs. It is observed from simulation results that EQUAL-B consumes 10.8 %, 5.44 %, and 7.69 % lesser amount of energy than NQRA, ABC, and GA, respectively.

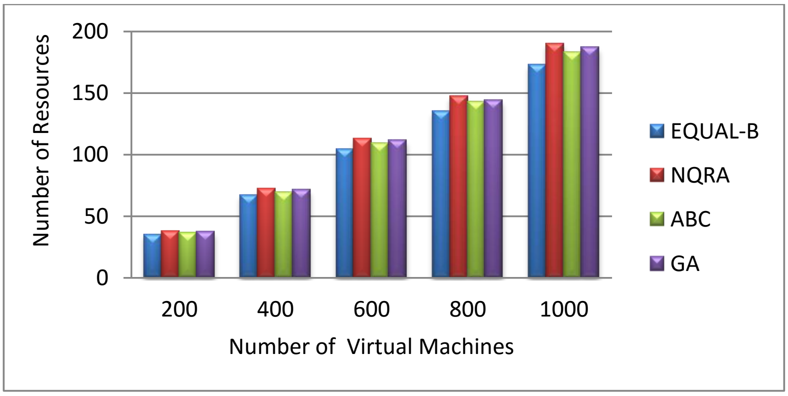


Figure 4. Comparison of the number of resources required

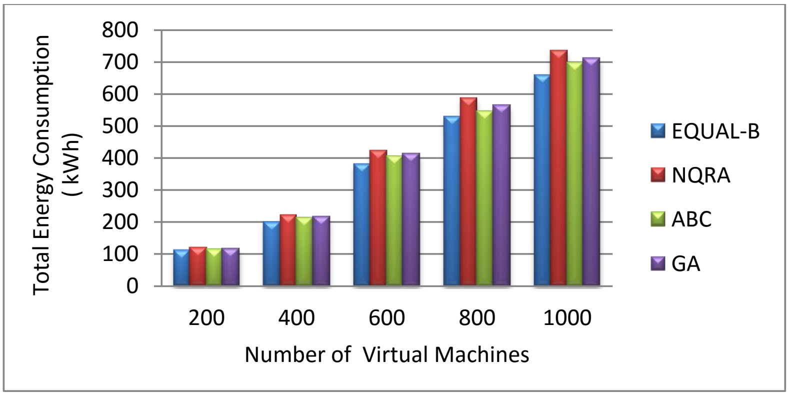


Figure 5. Comparison of total energy consumption

Figure 6 narrates comparison of average number of VM migrations performed in EQUAL-B, NQRA, ABC, and GA for different number of VMs. A VM is selected for migration using IQR. Resources becoming idle because of consolidation are switched to low power (sleep) mode to conserve energy. The number of VM migrations increases with the increase in number of VMs. In EQUAL-B 16.75 %, 10.97 %, and 16.65 % lesser number of VM migrations is observed than in NQRA, ABC, and GA, respectively.

Figure 7 describes comparison of number of hot-spots created in EQUAL-B, NQRA, ABC, and GA. The number of VMs is varied from 200 VMs to 1 000 VMs with increment of 200 VMs. A resource is considered as a hot-spot if its utilization is above HT. A hot-spot adversely affects the reliability and performance

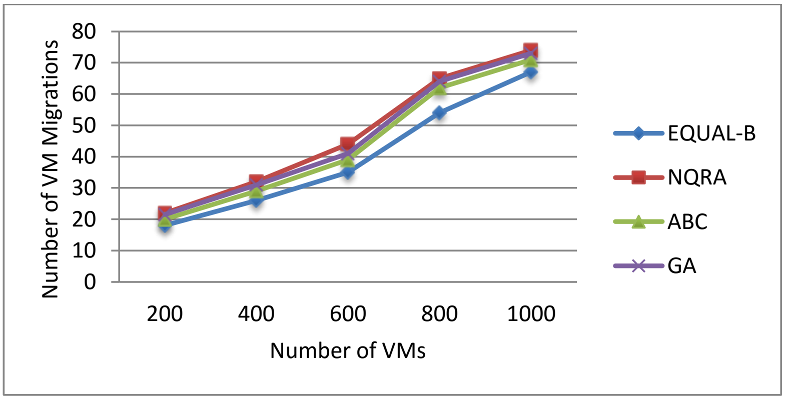


Figure 6. Comparison of the number of VM migrations

of the resource. Moreover, a hot-spot also demands better cooling arrangements. EQUAL-B improves reliability and energy efficiency of the resource as it creates 8.33 %, 18.20 %, and 14.22 % lesser number of hot-spots than NQRA, ABC, and GA, respectively.

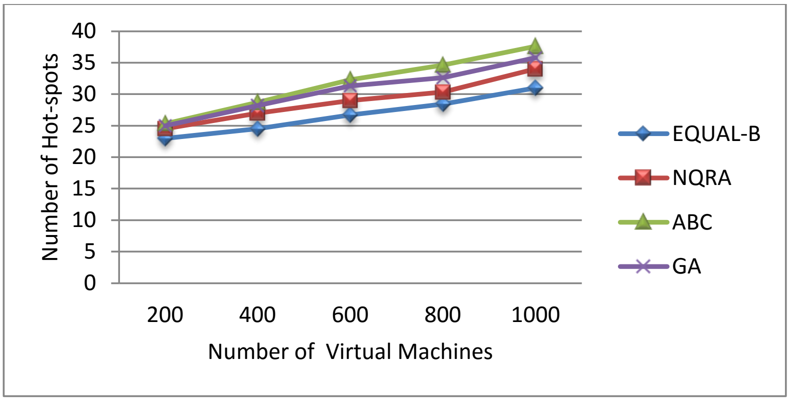


Figure 7. Comparison of the number of hot-spots

Figure 8 outlines the comparison of number of cold-spots observed in EQUAL-B, NQRA, ABC, and GA as the number of VMs is varied from 200 to 1 000 VMs. A resource is considered as a cold-spot if its utilization is below CT. The number of cold-spots portrays the extent of resource wastage. EQUAL-B creates 18, whereas NQRA, ABC, and GA create 25, 19.41, and 21.6 average number of cold-spots when tested with 1 000 VMs. The percentage of cold-spots generated in EQUAL-B, NQRA, ABC, and GA is 9.77 %, 11.52 %, 10.52 %, and

11.25 %, respectively. This shows that EQUAL-B manages the resources most efficiently.

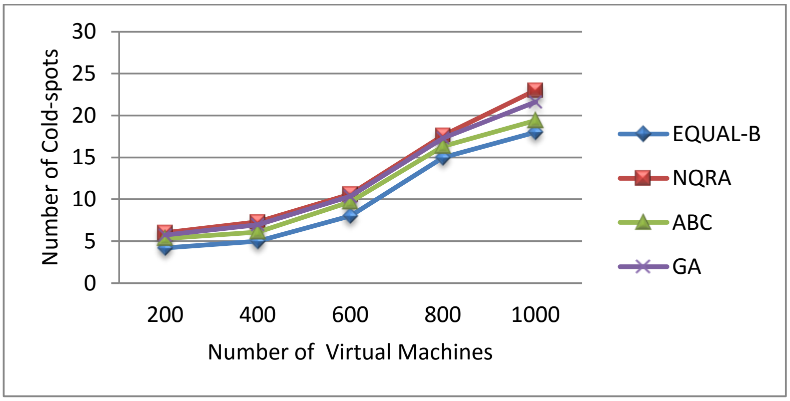


Figure 8. Comparison of the number of cold-spots

Figure 9 outlines number of deadlines missed in EQUAL-B, NQRA, ABC, and GA. A time constrained task executing in a VM is said to miss the deadline if it is not accomplished in stipulated time. The number of tasks is varied from 200 to 1 000. In each run of the simulation 200 VMs are used. The tasks are distributed equally among the VMs. It is observed that the number of deadlines missed increases with the increase in the number of tasks, but the rate of increase of missed deadlines is least in EQUAL. In EQUAL-B, 28.57 %, 11.76 %, and 25.00 % less deadline misses are observed than NQRA, ABC and GA, when number of tasks is 200. However, for 1 000 tasks, 21.58 %, 9.57 %, and 17.33 % less tasks miss their deadline in EQUAL-B as compared to NQRA, ABC, and GA.

Figure 10 shows comparison of allocation overhead that is total time taken by an algorithm to find the most suitable resource for each VMs. Allocation overhead of EQUAL is more than that of NQRA. However it is lesser than ABC and GA. In case of EQUAL-B, allocation overhead is about 26 s for 200 VMs and it increases to 112 s for 1 000 VMs. However, allocation overhead of NQRA, ABC and GA for 200 VMs is 20 s, 28.5 s and 30 s, and for 1 000 VMs it is 92 s, 122 s and 128 s, respectively. The results indicate that EQUAL-B has a higher convergence rate than ABC and GA.

Case 2: EQUAL in Energy-Aware Mode

The variable θ is assigned value 0.05 to operate EQUAL in energy aware mode (EQUAL-E). In this mode of operation, energy consumption of resources is considered while allocating resources to VMs. A VM is allocated to a resource

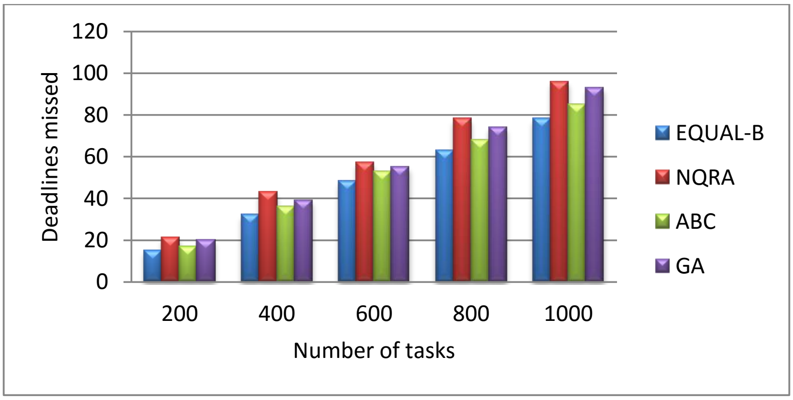


Figure 9. Comparison of number of deadlines missed

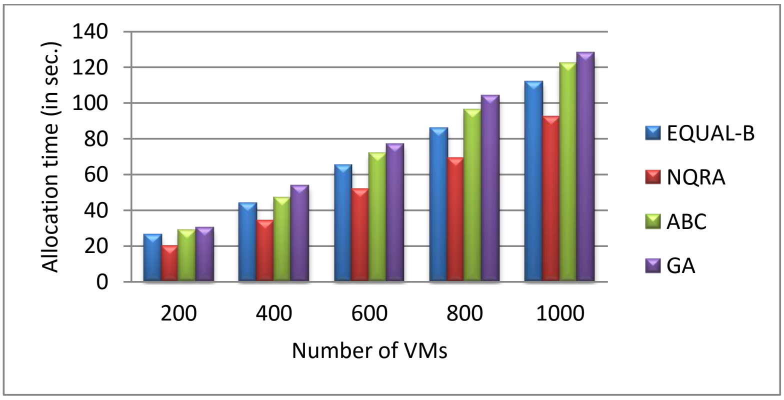


Figure 10. Comparison of total allocation time

that results in minimum increase in energy consumption. Further, the control variable γ is given value 0.05 to pack VMs on minimum number of resources.

Figure 11 depicts the comparison of number of resources used by EQUAL-E, NQRA, ABC, and GA for different number of VMs. The results show that EQUAL-E uses lesser number of resources than NQRA, ABC, and GA for a given number of VMs. It is observed that EQUAL-E uses 14.47 %, 10.05 %, and 12.54 % lesser number of resources than EQRA, ABC, and GA, respectively.

Figure 12 outlines the comparison of energy consumption of EQUAL-E, NQRA, ABC, and GA. EQUAL-E saves energy by packing VMs on lesser number of

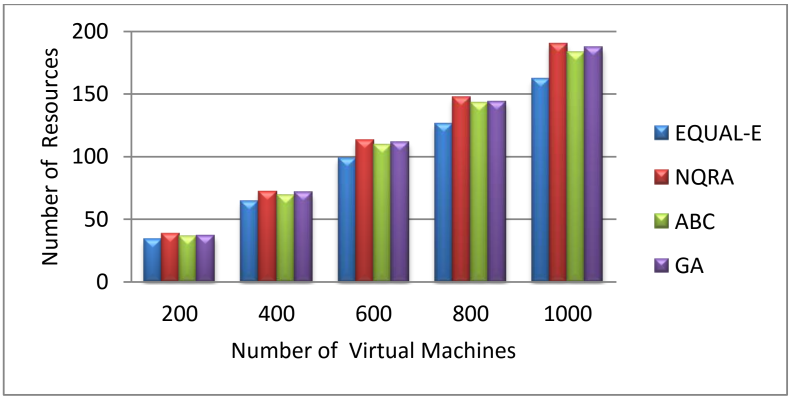


Figure 11. Comparison of number of resources required

resources. As compared to NQRA, ABC, and GA, average energy savings of 15.04 %, 11.91 %, and 14.30 % are observed in EQUAL-E.

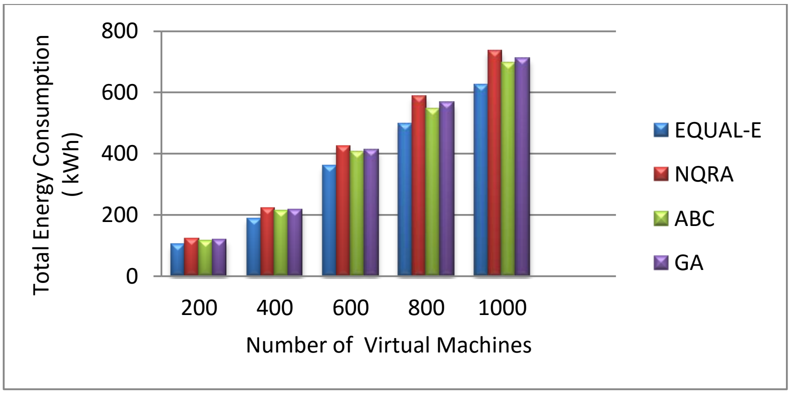


Figure 12. Comparison of total energy consumption

Figure 13 sketches comparison of average number of VM migrations performed in EQUAL-E, NQRA, ABC, and GA. VMs are migrated from either under-loaded or over-loaded resources. A resource is considered under-loaded if its utilization is below LGT, and over-loaded if its utilization is above UGT. It was observed that the number of VM migrations increases when the number of VMs increased from 200 to 1 000. In EQUAL-E, 9.37 % 3.45 %, and 6.05 % lesser number of VM migrations were observed than in NQRA, ABC, and GA, respectively.

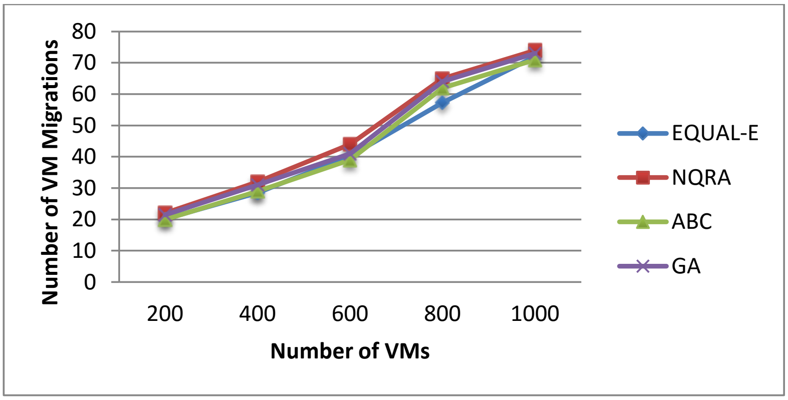


Figure 13. Comparison of the number of VM migrations

Figure 14 depicts the comparison of the number of hot-spots created in EQUAL-E, NQRA, ABC, and GA as the number of VMs are changed from 200 to 1 000 VMs. As compared to EQUAL-B, EQUAL-E packs VMs on lesser number of resources. As a result, the number of hot-spots increases and the gap of percentage number of hot-spots between EQUAL-E and NQRA, ABC, and GA was reduced to 3.86 %, 13.33 %, and 9.52 %, respectively.

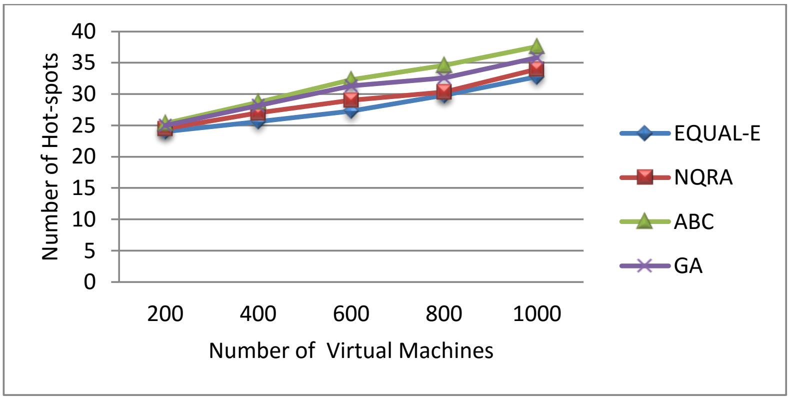


Figure 14. Comparison of the number of hot-spots

Figure 15 shows the comparison of the number of cold-spots observed in EQUAL-E, NQRA, ABC, and GA. In EQUAL-E, fewer number of cold-spots are observed than in EQUAL-B. On the average, approximately 16 cold-spots are observed in EQUAL-E against 1 000 VMs compared to 23, 19.41, and 21.6 average number of cold-spots in NQRA, ABC, and GA, respectively, for the same number of VMs.

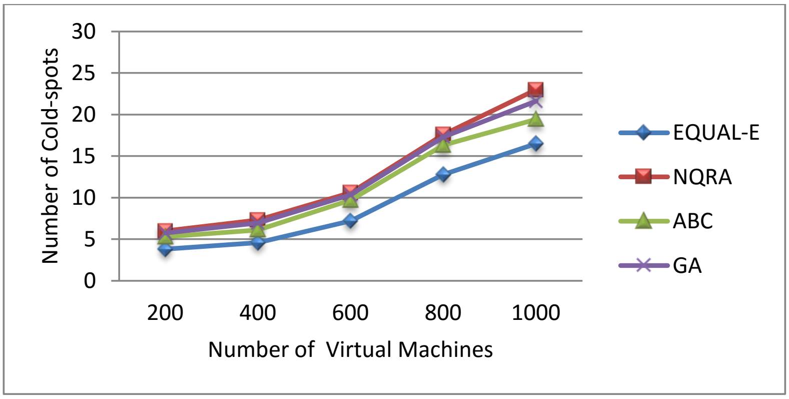


Figure 15. Comparison of the number of cold-spots

Figure 16 narrates comparison of the number of tasks that missed their deadline in EQUAL-E, NQRA, ABC, and GA. The number of tasks is changed from 200 to 1 000. In each simulation run 200 VMs are used. Further, the tasks are distributed equally among the VMs. In EQUAL-E, VMs are mapped on lesser number of resources than in EQUAL-B. As a result, tasks encapsulated in the VMs do not get sufficient resources causing increase in number of deadline miss. Due to this, percentage deadlines missed gap between EQUAL-E and the other three approaches, i.e., NQRA, ABC, and GA reduce to 13.25 %, 6.28 %, and 7.31 %, respectively.

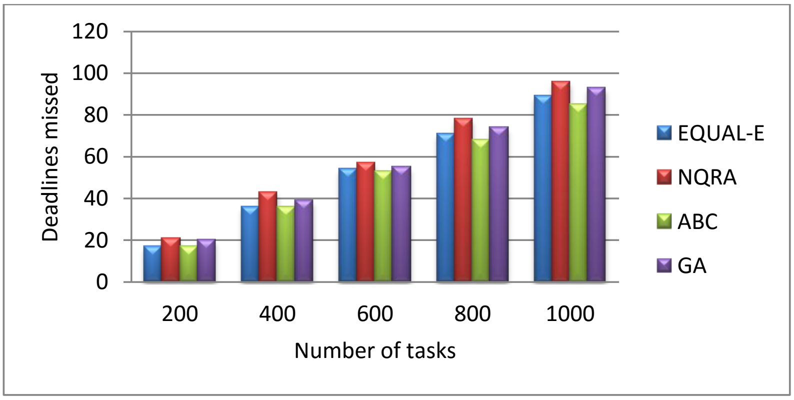


Figure 16. Comparison of the number of missed deadlines

Case 3: EQUAL in Performance-Aware Mode

The variable θ is assigned value 0.95 to tune EQUAL to performance-aware mode (EQUAL-P). In this mode of operation, available computational capacity of each resource is considered while discovering suitable resource for a VM. Since the VMs are required to be allocated on a minimum number of resources, so value 0.05 is assigned to control parameter γ.

Figure 17 shows the comparison of the number of resources used by EQUAL-P, NQRA, ABC, and GA. In EQUAL-P, a resource with higher performance affinity value is given preference over the others. In EQUAL-P more number of resources are used than in EQUAL-B and EQUAL-E for the given number of VMs. It is observed that EQUAL-E uses 8.20 %, 4.98 %, and 7.23 % lesser number of resources than NQRA, ABC, and GA, respectively.

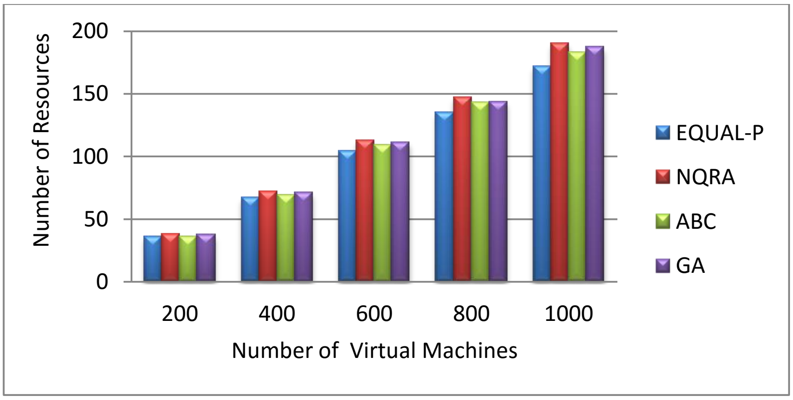


Figure 17. Comparison of the number of resources required

Figure 18 depicts the comparison of energy consumption of EQUAL-P, NQRA, ABC, and GA. Energy consumption in EQUAL-P for given number of VMs is higher than energy consumption in EQUAL-B and EQUAL-E because it uses larger number of resources. However, energy consumption in EQUAL-P is lower than that of ABC and GA. Energy consumption of EQUAL-P is measured 8.77 %, 4.73 % and 6.94 % lower for 200 VMs, and 9.75 %, 5.04 % and 6.98 % lower for 1 000 VMs than that of NQRA, ABC and GA, respectively.

Figure 19 represents a comparison of the average number of VM migrations performed in EQUAL-P, NQRA, ABC, and GA. In EQUAL-P, in average 16.8 and 64 migrations are observed for 200 VMs and 1 000 VMs, respectively. In EQUAL-P, the number of migration is lesser than the number of migrations in the balanced and energy aware mode.

Figure 20 depicts the comparison of the hot-spots created in EQUAL-P, NQRA, ABC, and GA. In EQUAL-P, fewer hot-spots than in EQUAL-P and EQUAL-B

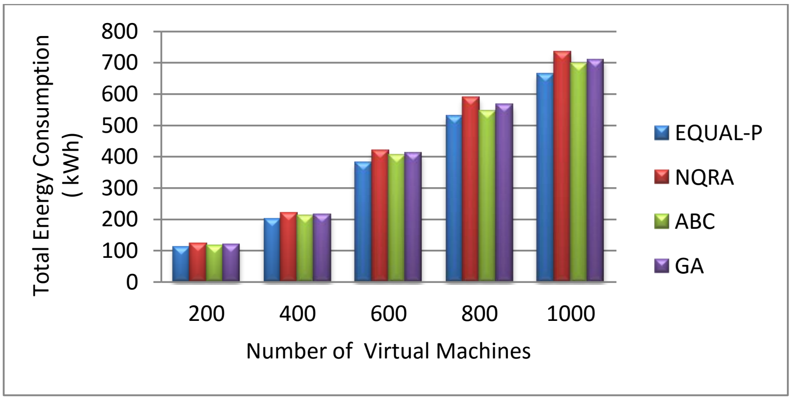


Figure 18. Comparison of the total energy consumption

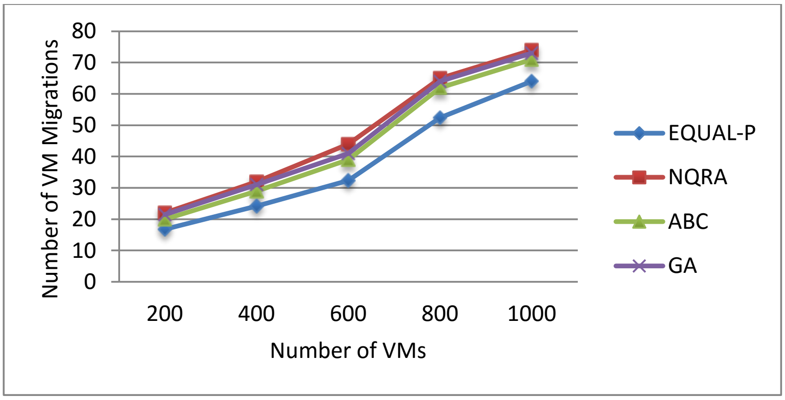


Figure 19. Comparison of the number of VM migrations

are observed. It is observed that EQUAL-P creates 11.6 %, 21.31 %, and 17.21 % lesser number of hot-spots than NQRA, ABC, and GA, respectively.

Figure 21 shows the comparison of number of cold-spots observed in EQUAL-P, NQRA, ABC, and GA. A resource is considered as a cold-spot if its utilization is below CT. A large proportion of the resource capacity goes wasted if it is a cold-spot. Therefore, the larger is the number of cold-spots the greater is the resource wastage. It is observed that EQUAL-P generates 13.68 %, 6.12 %, and 11.66 % lesser number of cold-spots than NQRA, ABC and GA. Therefore, it utilizes the resource more efficiently.

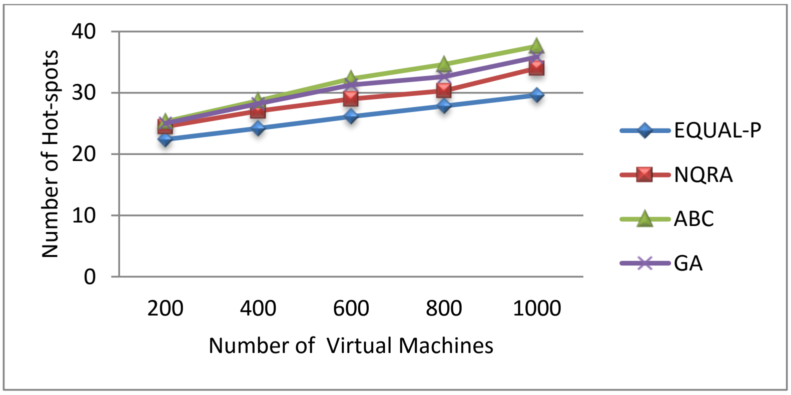


Figure 20. Comparison of the number of hot-spots

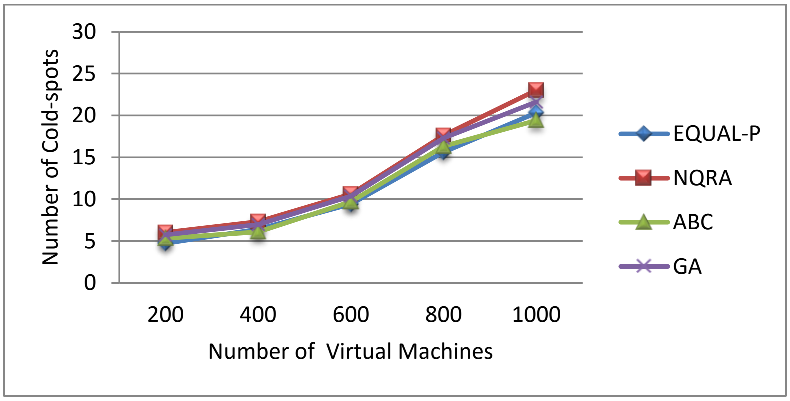


Figure 21. Comparison of the number of cold-spots

Figure 22 outlines the comparison of the average number of deadlines missed by EQUAL-P, NQRA, ABC, and GA. In each simulation run 200 VMs are used. Number of task is varied from 200 to 1 000 and tasks are distributed equally among the VMs. In EQUAL-P, fewer tasks miss their deadlines than in EQUAL-B and EQUAL-E. This is due to the fact that EQUAL-P uses more resources to map a given number of VMs. In EQUAL-P, on the average, 12 tasks miss their deadlines when the total number of tasks is 200, whereas 63 tasks miss the deadlines when the total number of tasks increased to 1 000. However, for NQRA, ABC and GA the number of tasks that missed their deadline is 21, 17 and 20 for total 200 tasks, and 96, 85 and 93 for 1 000 tasks, respectively.

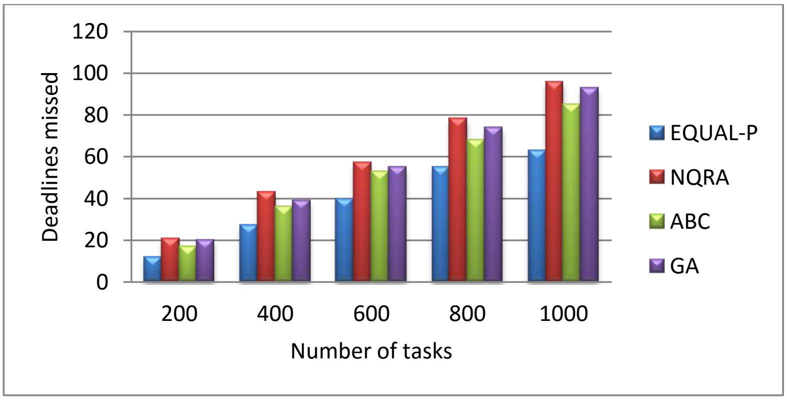


Figure 22. Comparison of the number of missed deadlines

6 CONCLUSION AND FUTURE WORK

In this work, the energy and QoS aware resource allocation approach (EQUAL) is proposed. Antlion optimization is used for allocation of resources to VMs which encapsulate heterogeneous time constrained tasks. EQUAL can be governed to operate in one of the three modes namely power aware, performance aware and balanced mode. The proposed approach was implemented in CloudSim, and tested with VMs/tasks having diverse resource requirements. The experimental results have proved that the proposed approach reduces the energy consumption up to 15 %, and also improves the quality of service in terms of reduction in the percentage of tasks that missed their deadlines. In future, the proposed approach can be further extended for tasks having mixed characteristics such as CPU intensive, memory intensive, input/output intensive, etc.

Appendix A SYMBOLIC NOTATIONS USED IN EQUAL

Table A1: List of Symbols

Symbol	Definition
P_{total}	Total power consumption of a physical machine
$P_{dynamic}$	Dynamic power consumption of a physical machine
P_{static}	Static power consumption of a physical machine
P_{idle}	Power consumption of a physical machine when idle
P_{max}	Power consumption of PM at 100 % utilization
U	Utilization of a PM
P	Power consumption of PM at U % utilization

Continued on next page

Table A1 – continued from previous page

Symbol	**Definition**
e	Elite solution representing the best resource
n	Number of tasks
J_i	i^{th} task
d_i	Deadline of task J_i
w_i	Processing volume of task J_i
V	Number of virtual machines
m	Number of resources
S	Set of resources
A	Set of ants
L	Set of antlions
S_j	j^{th} resource
$\Re_r^a$	Available processing power of resource r
$\Re_j^d$	Processing demand of VM j
$\triangle E_{j,r}$	Energy contribution of VM j on resource r
κ_r	Energy affinity
$\Re_{i,r}$	Fraction of processing power allocated to VM i on resource r
γ	Constant that controls energy contribution and VMs consolidation
θ	Trade off between performance and energy
$f_{j,r}$	Fitness of VM j on resource r
M^a	Matrix to store location of ants
M^{al}	Matrix to store location of antlions
M^{fa}	Matrix to store fitness values of ants
M^{fal}	Matrix to store fitness values of antlions
W_i^t	Location of i^{th} ant at t^{th} iteration
V_j^t	Location of j^{th} antlion at t^{th} iteration
α	Scaling factor for step size s
λ	Levy exponent
$L(s, \lambda)$	Levy Distribution with parameters s and λ
a_i	Minimum of random walk of i^{th} ant
b_i	Maximum of random walk of i^{th} ant
c_i^t	Minimum of search space at t^{th} iteration
d_i^t	Maximum of search space at t^{th} iteration
M_{VR}	VM-Resource map
UGT	Upper Green Threshold limit
LGT	Lower Green Threshold limit
HT	Hot-spot Threshold
CT	Cold-spot Threshold

REFERENCES

[1] Armbrust, M.—Fox, A.—Griffith, R.—Joseph, A. D.—Katz, R. H.—Konwinski, A.—Lee, G.—Patterson, D. A.—Rabkin, A.—Stoica, I.—Zaharia, M.: Above the Clouds: A Berkeley View of Cloud Computing. Technical Report No. UCB/EECS-2009-28, EECS Department, University of California, Berkeley, 2009.

[2] Columbus, L.: Predicting Enterprise Cloud Computing Growth. `http://www.forbes.com/sites/louiscolumbus/2013/09/04/predicting-enterprise-cloud-computing-growth/`, online; accessed 24-Jan-2016.

[3] Fargo, F.—Tunc, C.—Al-Nashif, Y.—Akoglu, A.—Hariri, S.: Autonomic Workload and Resources Management of Cloud Computing Services. International Conference on Cloud and Autonomic Computing (ICCAC), 2014, pp. 101–110, doi: 10.1109/ICCAC.2014.36.

[4] Vogels, W.: Beyond Server Consolidation. Queue, Vol. 6, 2008, No. 1, pp. 20–26, doi: 10.1145/1348583.1348590, doi: 10.1145/1348583.1348590.

[5] America's Data Centers Consuming and Wasting Growing Amounts of Energy. `http://www.nrdc.org/energy/data-center-efficiency-assessment.asp`, online; accessed 24-Jan-2016.

[6] Kaplan, J. M.—Forrest, W.—Kindler, N.: Revolutionizing Data Center Energy Efficiency. Technical report, McKinsy and Company, 2008.

[7] Vrbsky, S. V.—Galloway, M.—Carr, R.—Nori, R.—Grubic, D.: Decreasing Power Consumption with Energy Efficient Data Aware Strategies. Future Generation Computer Systems, Vol. 29, 2013, No. 5, pp. 1152–1163, doi: 10.1016/j.future.2012.12.016.

[8] Dahiphale, D.—Karve, R.—Vasilakos, A. V.—Liu, H.—Yu, Z.—Chhajer, A.—Wang, J.—Wang, C.: An Advanced MapReduce: Cloud MapReduce, Enhancements and Applications. IEEE Transactions on Network and Service Management, Vol. 11, 2014, No. 1, pp. 101–115, doi: 10.1109/TNSM.2014.031714.130407.

[9] Hamilton, J.: The Cost of Latency. `http://perspectives.mvdirona.com/2009/10/the-cost-of-latency/`.

[10] Liang, Q.—Liang, J.—Zou, F.: The Resource Configuration Method with Lower Energy Consumption Based on Prediction in Cloud Data Center. Journal of Networks, Vol. 9, 2014, No. 7, pp. 1692–1700.

[11] Lee, H. M.—Jeong, Y.-S.—Jang, H. J.: Performance Analysis Based Resource Allocation for Green Cloud Computing. The Journal of Supercomputing, Vol. 69, 2014, No. 3, pp. 1013–1026.

[12] Quarati, A.—Clematis, A.—Galizia, A.—D'Agostino, D.: Hybrid Clouds Brokering: Business Opportunities, QoS and Energy-Saving Issues. Simulation Modelling Practice and Theory, Vol. 39, 2013, pp. 121–134, doi: 10.1016/j.simpat.2013.01.004.

[13] Bobroff, N.—Kochut, A.—Beaty, K.: Dynamic Placement of Virtual Machines for Managing SLA Violations. 10th IFIP/IEEE International Symposium on Integrated Network Management, 2007, pp. 119–128, doi: 10.1109/INM.2007.374776.

[14] Wang, M.—Meng, X.—Zhang, L.: Consolidating Virtual Machines with Dynamic Bandwidth Demand in Data Centers. Proceedings of INFOCOM, 2011, IEEE, 2011, pp. 71–75, doi: 10.1109/INFCOM.2011.5935254.

[15] Takeda, S.—Takemura, T.: A Rank-Based VM Consolidation Method for Power Saving in Datacenters. IPSJ Transactions on Advanced Computing Systems, Vol. 3, 2010, No. 2, pp. 138–146, doi: 10.2197/ipsjtrans.3.88.

[16] Son, S.—Jung, G.—Jun, S. C.: An SLA-Based Cloud Computing That Facilitates Resource Allocation in the Distributed Data Centers of a Cloud Provider. The Journal of Supercomputing, Vol. 64, 2013, No. 2, pp. 606–637.

[17] Chieu, T. C.—Mohindra, A.—Karve, A. A.—Segal, A.: Dynamic Scaling of Web Applications in a Virtualized Cloud Computing Environment. IEEE International Conference on e-Business Engineering (ICEBE '09), 2009, pp. 281–286.

[18] Wu, L.—Garg, S. K.—Buyya, R.: SLA-Based Resource Allocation for Software as a Service Provider (SaaS) in Cloud Computing Environments. 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), 2011, pp. 195–204.

[19] Raycroft, P.—Jansen, R.—Jarus, M.—Brenner, P. R.: Performance Bounded Energy Efficient Virtual Machine Allocation in the Global Cloud. Sustainable Computing: Informatics and Systems, Vol. 4, 2014, No. 1, pp. 1–9.

[20] Kim, N.—Cho, J.—Seo, E.: Energy-Credit Scheduler: An Energy-Aware Virtual Machine Scheduler for Cloud Systems. Future Generation Computer Systems, Vol. 32, 2014, pp. 128–137, doi: 10.1016/j.future.2012.05.019.

[21] Xu, J.—Fortes, J. A. B.: Multi-Objective Virtual Machine Placement in Virtualized Data Center Environments. IEEE/ACM International Conference on Green Computing and Communications (GreenCom), 2010, pp. 179–188, doi: 10.1109/GreenCom-CPSCom.2010.137.

[22] Wu, C.-M.—Chang, R.-S.—Chan, H.-Y.: A Green Energy-Efficient Scheduling Algorithm Using the DVFS Technique for Cloud Datacenters. Future Generation Computer Systems, Vol. 37, 2014, pp. 141–147, doi: 10.1016/j.future.2013.06.009.

[23] Beloglazov, A.—Buyya, R.: Managing Overloaded Hosts for Dynamic Consolidation of Virtual Machines in Cloud Data Centers under Quality of Service Constraints. IEEE Transactions on Parallel and Distributed Systems, Vol. 24, 2013, No. 7, pp. 1366–1379, doi: 10.1109/TPDS.2012.240.

[24] Nathuji, R.—Schwan, K.: VirtualPower: Coordinated Power Management in Virtualized Enterprise Systems. ACM SIGOPS Operating Systems Review, Vol. 41, 2007, No. 6, pp. 265–278, doi: 10.1145/1294261.1294287.

[25] Hsu, C.-H.—Chen, S.-C.—Lee, C.-C.—Chang, H.-Y.—Lai, K.-C.—Li, K.-C.—Rong, C.: Energy-Aware Task Consolidation Technique for Cloud Computing. IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), 2011, pp. 115–121, doi: 10.1109/CloudCom.2011.25.

[26] Beloglazov, A.—Abawajy, J.—Buyya, R.: Energy-Aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing. Future Generation Computer Systems, Vol. 28, 2012, No. 5, pp. 755–768, doi: 10.1016/j.future.2011.04.017.

[27] Kusic, D.—Kephart, J. O.—Hanson, J. E.—Kandasamy, N.—Jiang, G.: Power and Performance Management of Virtualized Computing Environments via Lookahead Control. International Conference on Autonomic Computing (ICAC '08), 2008, pp. 3–12, doi: 10.1109/ICAC.2008.31.

[28] Gao, Y.—Guan, H.—Qi, Z.—Song, T.—Huan, F.—Liu, L.: Service Level Agreement Based Energy-Efficient Resource Management in Cloud Data Centers. Computers and Electrical Engineering, Vol. 40, 2014, No. 5, pp. 1621–1633.

[29] Feller, E.—Rilling, L.—Morin, C.: Energy-Aware Ant Colony Based Workload Placement in Clouds. IEEE/ACM 12th International Conference on Grid Computing (GRID), 2011, pp. 26–33, doi: 10.1109/Grid.2011.13.

[30] Gao, Y.—Guan, H.—Qi, Z.—Hou, Y.—Liu, L.: A Multi-Objective Ant Colony System Algorithm for Virtual Machine Placement in Cloud Computing. Journal of Computer and System Sciences, Vol. 79, 2013, No. 8, pp. 1230–1242.

[31] Chen, H.—Cheng, A. M. K.—Kuo, Y.-W.: Assigning Real-Time Tasks to Heterogeneous Processors by Applying Ant Colony Optimization. Journal of Parallel and Distributed Computing, Vol. 71, 2011, No. 1, pp. 132–142.

[32] Huang, C.-J.—Guan, C.-T.—Chen, H.-M.—Wang, Y.-W.—Chang, S.-C.—Li, C.-Y.—Weng, C.-H.: An Adaptive Resource Management Scheme in Cloud Computing. Engineering Applications of Artificial Intelligence, Vol. 26, 2013, No. 1, pp. 382–389.

[33] Kansal, N. J.—Chana, I.: Artificial Bee Colony Based Energy-Aware Resource Utilization Technique for Cloud Computing. Concurrency and Computation: Practice and Experience, Vol. 27, 2015, No. 5, pp. 1207–1225.

[34] Chimakurthi, L.—Kumar, S. D. M.: Power Efficient Resource Allocation for Clouds Using Ant Colony Framework. CoRR abs/1102.2608, `http://arxiv.org/abs/1102.2608`.

[35] Hu, W.—Zheng, J.—Hua, X.—Yang, Y.: A Computing Capability Allocation Algorithm for Cloud Computing Environment. Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013), 2013, pp. 2258–2262, doi: 10.2991/iccsee.2013.566.

[36] Liu, X.-F.—Zhan, Z.-H.—Du, K.-J.—Chen, W.-N.: Energy Aware Virtual Machine Placement Scheduling in Cloud Computing Based on Ant Colony Optimization Approach. Proceedings of the 2014 Conference on Genetic and Evolutionary Computation (GECCO '14), 2014, pp. 41–48, doi: 10.1145/2576768.2598265.

[37] Portaluri, G.—Giordano, S.—Kliazovich, D.—Dorronsoro, B.: A Power Efficient Genetic Algorithm for Resource Allocation in Cloud Computing Data Centers. IEEE 3rd International Conference on Cloud Networking (CloudNet), 2014, pp. 58–63, doi: 10.1109/CloudNet.2014.6968969.

[38] Xiong, A.-P.—Xu, C.-X.: Energy Efficient Multiresource Allocation of Virtual Machine Based on PSO in Cloud Data Center. Mathematical Problems in Engineering, 2014, pp. 1–8, `http://www.hindawi.com/journals/mpe/2014/816518/`.

[39] Kumar, D.—Raza, Z.: A PSO Based VM Resource Scheduling Model for Cloud Computing. 2015 IEEE International Conference on Computational Intelligence and Communication Technology, 2015, pp. 213–219, `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7078697`.

[40] Dashti, S. E.—Rahmani, A. M.: Dynamic VMs Placement for Energy Efficiency by PSO in Cloud Computing. Journal of Experimental and Theoretical Artificial Intelligence, Vol. 28, 2016, No. 1-2, pp. 97–112, doi: 10.1080/0952813X.2015.1020519.

[41] Kansal, N. J.—Chana, I.: Energy-Aware Virtual Machine Migration for Cloud Computing – A Firefly Optimization Approach. Journal of Grid Computing, Vol. 14, 2016, No. 2, pp. 327–345, doi: 10.1007/s10723-016-9364-0.

[42] Kumar, A.—Kumar, R.—Sharma, A.: Energy Aware Resource Allocation for Clouds Using Two Level Ant Colony Optimization. Computing and Informatics, Vol. 37, 2018, No. 1, pp. 76–108.

[43] Kumar, R.—Kumar, A.—Sharma, A.: A Bio-Inspired Approach for Power and Performance Aware Resource Allocation in Clouds. MATEC Web of Conferences, Vol. 57, 2016, Art. No. 02008, 6 pp.

[44] Beloglazov, A.—Buyya, R.: Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers. Concurrency and Computation: Practice and Experience, Vol. 24, 2012, No. 13, pp. 1397–1420, doi: 10.1002/cpe.1867, doi: 10.1002/cpe.1867.

[45] Beloglazov, A.: Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing. Ph.D. thesis, University of Melbourne, 2013.

[46] Barham, P.—Dragovic, B.—Fraser, K.—Hand, S.—Harris, T.—Ho, A.—Neugebauer, R.—Pratt, I.—Warfield, A.: Xen and the Art of Virtualization. ACM SIGOPS Operating Systems Review – SOSP '03, Vol. 37, 2003, No. 5, pp. 164–177, doi: 10.1145/945445.945462.

[47] Kivity, A.—Kamay, Y.—Laor, D.—Lublin, U.—Liguori, A.: KVM: The Linux Virtual Machine Monitor. Proceedings of the Linux Symposium, Ottawa, Ontario, Canada, Vol. 1, 2007, pp. 225–230, `http://linux-security.cn/ebooks/ols2007/OLS2007-Proceedings-V1.pdf`.

[48] Walters, B.: VMware Virtual Platform. Linux Journal, 1999, No. 63es., Art. No. 6, `http://dl.acm.org/citation.cfm?id=327906.327912`.

[49] Mirjalili, S.: The Ant Lion Optimizer. Advances in Engineering Software, Vol. 83, 2015, pp. 80–98, doi: 10.1016/j.advengsoft.2015.01.010.

[50] Yang, X.-S.—Deb, S.: Cuckoo Search via Lévy Flights. World Congress on Nature and Biologically Inspired Computing (NaBIC), 2009, pp. 210–214, doi: 10.1109/NABIC.2009.5393690.

[51] Calheiros, R. N.—Ranjan, R.—Beloglazov, A.—De Rose, C. A. F.—Buyya, R.: CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing

Environments and Evaluation of Resource Provisioning Algorithms. Software: Practice and Experience (SPE), Vol. 41, 2011, No. 1, pp. 23–50.

[52] Wickremasinghe, B.—Calheiros, R. N.—Buyya, R.: CloudAnalyst: A CloudSim-Based Visual Modeller for Analysing Cloud Computing Environments and Applications. 24th IEEE International Conference on Advanced Information Networking and Applications (AINA), 2010, pp. 446–452, doi: 10.1109/AINA.2010.32.

[53] Kliazovich, D.—Bouvry, P.—Audzevich, Y.—Khan, S. U.: GreenCloud: A Packet-Level Simulator of Energy-Aware Cloud Computing Data Centers. Global Telecommunications Conference (GLOBECOM 2010), IEEE, 2010, pp. 1–5, doi: 10.1109/GLOCOM.2010.5683561.

[54] Garg, S. K.—Buyya, R.: NetworkCloudSim: Modelling Parallel Applications in Cloud Simulations. 2011 Fourth IEEE International Conference on Utility and Cloud Computing (UCC), 2011, pp. 105–113, doi: 10.1109/UCC.2011.24.

[55] General Purpose Compute: Basic Tier. `https://azure.microsoft.com/en-in/pricing/details/virtual-machines/`, online; accessed 29-March-2016.

[56] Xu, J.—Fortes, J. A. B.: Multi-Objective Virtual Machine Placement in Virtualized Data Center Environments. IEEE/ACM International Conference on Green Computing and Communications (GreenCom), 2010, pp. 179–188, doi: 10.1109/GreenCom-CPSCom.2010.137.

[57] Calheiros, R. N.—Buyya, R.: Meeting Deadlines of Scientific Workflows in Public Clouds with Tasks Replication. IEEE Transactions on Parallel and Distributed Systems, Vol. 25, 2014, No. 7, pp. 1787–1796, doi: 10.1109/TPDS.2013.238.

[58] Juve, G.—Chervenak, A.—Deelman, E.—Bharathi, S.—Mehta, G.—Vahi, K.: Characterizing and Profiling Scientific Workflows. Future Generation Computer Systems, Vol. 29, 2013, No. 3, pp. 682–692.

[59] Abbott, R.—Garcia-Molina, H.: Scheduling Real-Time Transactions. ACM SIGMOD Record – Special Issue on Real-Time Database Systems, Vol. 17, 1988, No. 1, pp. 71–81.

Ashok Kumar received his M.Sc. degree in information technology from Punjab Technical University, Jalandhar. Currently he is pursuing his doctoral degree in cloud computing from Thapar University, Patiala. His research interests include cloud computing, internet of things and fog computing. He has five research publications in reputed journals and conferences.

Rajesh Kumar is currently working as Professor in the Computer Science and Engineering Department, Thapar University, Patiala. He received his M.Sc., M.Phil. and Ph.D. degrees from IIT Roorkee. He has more than 21 years of UG & PG teaching and research experience. He wrote over 101 research papers for various international and national journals and conferences. He has guided 10 Ph.D. and 23 M.E./M.Sc. theses so far. His current areas of research interests include FANETs, resource scheduling and fault tolerance in clouds.

Anju Sharma is working as Assistant Professor in the Computer Science and Engineering Department, MRSPTU, Bathinda. Her research interests include smart grid computing, cloud computing, IoT and fog computing. She has varied numbers of publications in international journals and conferences of repute. She is Senior Member of International Association of Computer Science and Information Technology (IACSIT) and a professional member of ACM India, IEEE. She is an active member (TCM and reviewer) of varied conferences.

Computing and Informatics, Vol. 37, 2018, 815–837, doi: 10.4149/cai_2018_4_815

PERFORMANCE MODELS FOR FROST PREDICTION IN PUBLIC CLOUD INFRASTRUCTURES

Lucas E. Iacono

ITIC Research Institute
Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina
&
Facultad de Ingeniería, Universidad Nacional de Cuyo
Mendoza, Argentina
e-mail: `liacono@uncu.edu.ar`

José Luis Vázquez Poletti

Departamento de Arquitectura de Computadores y Automática
Facultad de Informática, Universidad Complutense de Madrid
Madrid, Spain
e-mail: `jlvazquez@fdi.ucm.es`

Carlos García Garino

ITIC Research Institute
Facultad de Ingeniería, Universidad Nacional de Cuyo
Mendoza, Argentina
e-mail: `cgarcia@itu.uncu.edu.ar`

Ignacio Martín Llorente

Departamento de Arquitectura de Computadores y Automática
Facultad de Informática, Universidad Complutense de Madrid
Madrid, Spain
e-mail: `llorente@dacya.ucm.es`

Abstract. Sensor Clouds have opened new opportunities for agricultural monitoring. These infrastructures use Wireless Sensor Networks (WSNs) to collect data on-field and Cloud Computing services to store and process them. Among other applications of Sensor Clouds, frost prevention is of special interest among grapevine producers in the Province of Mendoza – Argentina, since frost is one of the main causes of economic loss in the province. Currently, there is a wide offer of public cloud services that can be used in order to process data collected by Sensor Clouds. Therefore, there is a need for tools to determine which instance is the most appropriate in terms of execution time and economic costs for running frost prediction applications in an isolated or cluster way. In this paper, we develop models to estimate the performance of different Amazon EC2 instances for processing frosts prediction applications. Finally, we obtain results that show which is the best instance for processing these applications.

Keywords: Cloud computing, wireless sensor networks, frost prediction, virtual clusters, sensor clouds, Amazon EC2

1 INTRODUCTION

In the Province of Mendoza, region of Cuyo, Argentina, frost is one of the main causes of crop damage. This meteorological event causes damage in vineyards and fruit trees, which are the main agricultural products of the province. In some cases, like in the year 2013, frost damages affected up to 80 % of crops and resulted in the economic emergency of the region. Currently, there are different defense methods that can be used to minimize frost damage, the most commonly used are sprinklers, heaters and wind turbines [26].

Defense systems are activated by frost alarms, which are provided by Frost Alarm Systems (FAS). FAS perform on-field data acquisition and data management. Moreover, FAS ensure production quality and guarantee crops traceability. On the one hand, the on-field data acquisition process can be performed using traditional instruments like thermometers, weather stations or Wireless Sensor Networks (WSNs) [1, 23]. When making a comparison of traditional measurement instruments and weather stations, WSNs have the advantage of covering extensive areas with low cost devices called sensor nodes. This advantage is of special interest when studying frost due to the dependence of this phenomenon with terrain characteristics, like presence of weeds, trees or closeness to mountains. Sometimes frost occurrence has only been observed in a few hectares of the farm (such as those at the base of mountains) while it has not been observed in other hectares of the same farm (such as those surrounded by trees).

WSNs data management process includes data remote access, storage and data processing. This process can be reliably and easily performed using Cloud Computing technologies [2, 10, 12, 13, 19]. The use of Cloud Computing for data management allows to incorporate the benefits of this technology (data replication, fault tolerance, and resources scalability, among others) to FAS. There are two main reasons for using public Clouds in order to process and store WSNs data. The first one is the large volume of data generated by WSNs. As an example, in the region of Cuyo there are up to 170 000 cultivated hectares that can be instrumented with at least one sensor node per hectare. Therefore, there are 170 000 potential sensors that can generate data, which must be processed and stored in a proper infrastructure. The second reason is the traffic bottleneck from the WSNs to an isolated private data center.

Nowadays, the offer of Public Cloud Services is wide (Google Compute Cloud, Microsoft Azure, Amazon and others). One of the top providers included in that offer is Amazon. The Elastic Compute Cloud (EC2) toolkit service [4] provides different types of virtual machines (also called instances) for data processing and storage. Due to the wide range of instances offered by Amazon, there is a need for tools to identify which of these instances has better performance in terms of execution time and economic cost when processing frost prediction applications. In addition, these tools can provide information about a better way (single or cluster) to run these instances in order to minimize economic costs and execution times. In this paper, we propose a set of models constructed from empirical data that can be used to estimate the performance and economic costs of Amazon EC2 instances, applied for processing frost prevention applications. The proposed models allow to predict which is the instance that can process more sensor nodes in a certain time when the instance is working isolated. Although there are other costs associated with the use of Amazon EC2 instances – like the data transfer ones – the target of our study is the economic costs for WSNs data processing, taking into account that they are more relevant than the data transfer ones.

This paper is structured as follows: Section 2 introduces Frost Monitoring Systems based in WSNs. Next, Section 3 discusses related works, while Section 4 describes our application for frost prediction. Then Section 5 presents our performance estimation models for each Amazon EC2 instance and the methodology used to build them. Section 6 discusses the performance of Amazon EC2 instances in a typical use case; and Section 7, the accuracy of our models in this typical use case. Section 8 presents experiments about the instances' performance when they are executed in virtual clusters. Finally, Section 9 concludes this paper and details future works.

2 FROST ALARM SYSTEMS BASED ON WSNS

In this section, we provide an introduction of technologies used to perform data acquisition and management in FAS based on WSNs.

2.1 Data Acquisition with WSN

As shown in Figure 1, sensor nodes are composed of a micro-controller, memory, different sensors (e.g. temperature and humidity), battery and a radio module.

Sensor nodes can be interconnected into networks called WSNs, interacting among them. WSNs are used to study the environment and to acquire different variables related to weather, like temperature, humidity, pressure, solar radiation and others.

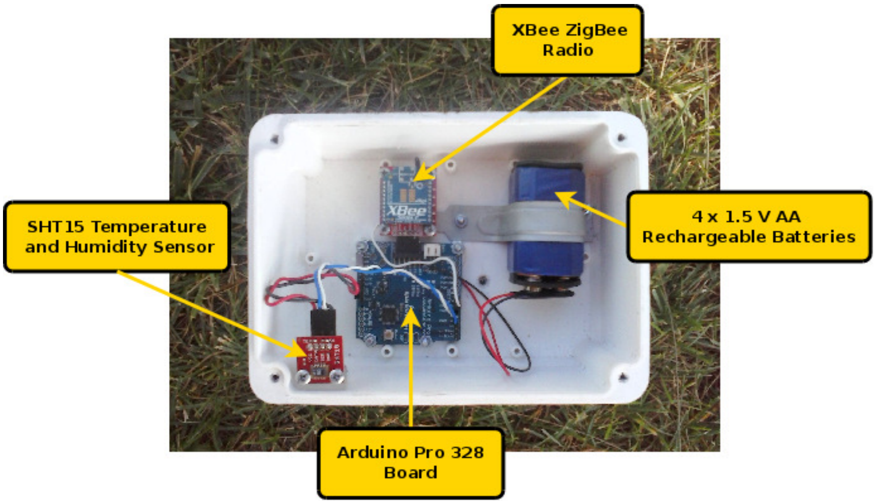

XBee ZigBee Radio
SHT15 Temperature and Humidity Sensor
4 x 1.5 V AA Rechargeable Batteries
Arduino Pro 328 Board

Figure 1. Sensor node based on Arduino Pro 328 board and ZigBee transceiver

Figure 2 illustrates a WSN for frost prediction that is deployed in a farm in the south of Mendoza, Argentina. In this WSN, data are acquired by source nodes and sent via ZigBee to a special node (known as sink). The sink node is connected to a personal computer (PC) or embedded system. The join of both sink node and PC is also called base station. This base station coordinates all operations of the WSN and transmits the information collected by sensor nodes to the final user, through the Internet.

WSN sensor nodes must meet requirements such as autonomy, low power consumption, low cost, robustness and reliability. Unlike traditional wireless networks, WSN nodes use communication protocols specifically designed for working with scarce energy and hardware resources. Some of these protocols are IEEE 802.15.4 [17], 6lowPAN [5] and ZigBee [29]. These ones are not compatible with TCP/IP networks, therefore, Base Station must include a gateway, which acts as translator between WSNs and the Internet communication protocols.

2.2 Data Management

Data collected by FAS through WSNs are used to provide solutions to frost prevention damage in crops [24, 27, 28]. Data management process starts when WSN data are sent to the Internet, then they are stored and processed into Public Clouds in order to obtain useful information for predicting frost. Next subsections cover the details of different technologies used to WSN data management.

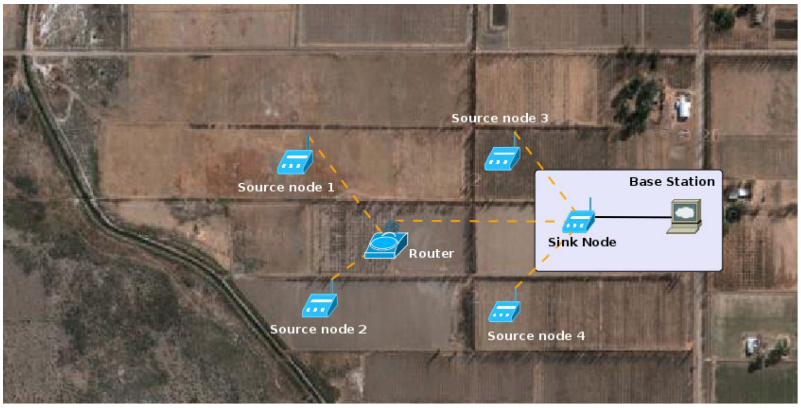


Figure 2. WSN for frost prediction in Mendoza, Argentina

2.2.1 Traditional Technologies

Generally, the use of single machines like typical PCs or mainframes is enough to process low volumes of non-critical WSN data. A typical case of single machine use is when low volumes of data (in the order of kilobytes) are sent from the base station (which is deployed in the field) to remote machines. Data transmission can be achieved using different technologies like TCP sockets [11], RSS services [25] and others. Next, the outside-WSN machine stores the data and proceeds to run the processing application.

Although this technology is suitable for processing WSN data, it presents some problems for processing large volume of data, scaling to a large number of WSN nodes and ensuring availability 24 hours a day – 365 days a year. A possible solution to solve these issues is using powerful servers, mainframes and clusters in appropriate data center infrastructures. However, this solution generates prohibitive economic costs, at least for associations of small farmers. Since the use of traditional technologies is not always suitable, different authors propose the use of Cloud Computing services for processing WSN data [2, 10, 12, 13, 19].

2.2.2 Cloud Computing

Cloud Computing is a paradigm for application development and for the use of computing and storage resources [6]. Through the use of virtualization techniques and web services, hardware resources and applications can be dynamically provided to the user.

Foster et al. [7] define Cloud Computing as *"A large-scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet"*. One of the main advantages of Clouds is resources scalability. In this way, Clouds can solve the computational and storage requirements of the applications. Cloud providers offer their services according to three fundamental models which are described below.

Infrastructure as a Service (IaaS)

Where *"service"* means resource. Through infrastructure services, users can access to virtualized high-performance computing (HPC) resources (CPUs and storage devices, among others). The service provider delivers resources to a client in accordance with the specific requirement, such as CPU type and power, memory, storage, and operating system among other. Amazon EC2 [4] can be cited as an example of IaaS, being a set of Cloud services which allow to run applications on custom virtual machines (VM) deployed in Amazon data center servers. Amazon offers various types of VMs with different processing power and memory capability.

Platform as a Service (PaaS)

Where *"service"* means platform-level functionality. These services provide Application Programming Interfaces (APIs) and standard development kits (SDKs) in order to allow users to develop and implement their own applications for Clouds. Some examples of these platforms are Google App Engine [8] and Windows Azure [22].

Software as a Service (SaaS)

Where *"service"* means application. The SaaS Cloud providers deliver applications that can be accessed by an end user through an Internet connection and a standard web browser. Furthermore, the applications can be developed with Platform Services and executed with Infrastructure Services. Among other SaaS, we can cite Google Drive [9] and SAP Business Suite[1].

3 RELATED WORKS

In the last six years, different authors have proposed the use of Cloud technologies for managing WSNs resources. In [18] Lee et al. describe concepts of Cloud like virtualized resources, SaaS, pay-per-use price model; and apply them to create a Cloud infrastructure capable of integrating devices with sensing capabilities. This infrastructure is called Tangible Cloud and uses Amazon EC2 instances in order to process data from sensor nodes [19]. In the paper, the authors demonstrate that the platform solves (through resources scalability) the computational power requirements of environmental monitoring and modeling applications.

Another work proposed by Ahmed and Gregory [2] introduces an integration framework between WSN and Cloud Computing. The main objective of the proposed framework is to *"facilitate the shift of data from WSN to the Cloud Computing*

[1] `http://go.sap.com/solution/cloud.html`

environment". In addition, the authors suggest that the join of Cloud Computing with sensor networks allows the possibility of WSNs data storage in publics domains. Then different users and applications can access to the sensors' information, resulting in a better data usage.

Aneka is another platform to integrate WSN into Clouds [10]. Aneka uses resources of Private and Public Clouds in order to provide support to applications of smart environments, including health-care, transportation, monitoring and others.

Regarding the use of Clouds in agricultural environments, Hirafuji et al. [12] developed an Ambient Sensor Cloud System for High-throughput Phenotyping. This platform allows the storage and access to data collected by means of sensor nodes using Twitter Cloud services. The main goal of the system is to provide a simple and economical solution to solve the access and storage of large datasets from various sensor nodes. Hori et al. [13] present a commercial solution to storage and process WSNs data. The platform allows the integration with business management, production history, traceability and good agricultural practice systems provided as a SaaS model.

In [20] Mazurek and Fukuda present MASS, a library for multi-agent spatial simulation and parallelizing temperature prediction programs. The authors propose the use of MASS for the processing of sensor data *"on the fly"*. This library allows the parallelization of frost prediction models, which is necessary in order to minimize the execution times of frost prediction methods. MASS can work in Cloud Computing in multi-core instances and virtual clusters. The authors implement the library and a frost prediction method based on Artificial Neural Networks, Prediction Polynomials and Inverse Distance Weighting. Finally, they prove that MASS parallelism improves the performance of frost prediction methods by 55 %.

Dinh and Kim present in [21] an efficient interactive model designed for providing WSNs services to multiple applications on Sensor Clouds. The model proposed by Dinh and Kim has three main goals: providing on-demand WSN services to different applications at the same time, minimizing the number of requests sent to physical sensor nodes, and optimizing energy consumption on WSN nodes. This model could be applied in agriculture because one of the main issues of agricultural WSNs is the minimization of battery consumption on sensor nodes.

Based on the works studied in this section, it can be concluded that Cloud is a promising technology for solving the management and processing of data in WSN's based FAS. Although most of the studied works use Amazon EC2, to the best of our knowledge, there are no works oriented to model the performance and economic cost of Amazon EC2 instances when they are processing agricultural monitoring applications.

4 FROST PREDICTION APPLICATION

In this section, the frost prediction application (FPA) is introduced. The main objective of this application is to compute the minimum temperature reached during the night. Then, according to this temperature value, frost occurrence on the farm can be predicted. The section is organized as follows. In Subsection 4.1, we present the method for frost prediction used in our application. Next, in Subsection 4.2 the application implementation is detailed.

4.1 Frost Prediction Method

The application was developed using the frost prediction method (FPM) belonging to Snyder and Melo-Abreu [26], which is based on Allen's equation [3]. The FPM predicts the minimum temperature that will occur in nights without both clouds and cold fronts. Therefore, it is only suitable to predict radiation frosts.

In order to carry out the prediction, the method takes temperature, humidity and dew point from days on which radiation frosts occurred. These days must belong to the month in which the prediction is performed (regardless the year). In this paper, we use data from a historical ten-year dataset. In addition, FPM needs the temperature, humidity and dew point recorded two hours after sunset in the prediction day.

Formally, the minimum temperature is calculated by the following linear regression (LR) equation:

$$T_p = s_T * T_o + s_D * D_0 + i \tag{1}$$

where T_p is the minimum temperature to be predicted. T_o is the temperature and D_0 dew point, both recorded the same day of the prediction two hours after sunset. Finally, i is the LR intercept, s_T temperature slope and s_D dew point slope. The values of s_T and i are calculated from the Equations (2) and (3), respectively.

$$s_T = \frac{\sum(T_{h0} - \bar{T}_{h0})(T_m - \bar{T}_m)}{\sum(T_{h0} - \bar{T}_{h0})^2}, \tag{2}$$

$$i = \frac{\sum T_m - s_T \sum T_{h0}}{n} \tag{3}$$

where T_{h0} are historical temperatures recorded two hours after sunset, T_m minimum temperatures that occurred in the night, and n is the number of historical data. Finally, $\bar{T}_{h0}$ and $\bar{T}_m$ account for the average data temperatures.

The slope s_D is calculated by using the Equation 4.

$$s_D = \frac{\sum(D_{h0} - \bar{D}_{h0})(R - \bar{R})}{\sum(D_{h0} - \bar{D}_{h0})^2} \tag{4}$$

where D_{h0} are historical dew points two hour after sunset and R the residuals. The parameters $\bar{D}_{h0}$ and $\bar{R}$ are the average of D_{h0} and R, respectively. Finally, the residual is calculated with the expression: $R = T_m - s_T * T_o + i$.

4.2 Application Implementation

In order to develop the FPA, we have implemented the Snyder and Melo-Abreu [26] FPM, using Java and MySQL. MySQL was used to store the data collected by sensor nodes and the results obtained after running the FPM. The application was executed using Amazon EC2 instances.

The integration of WSN data with Cloud infrastructures was performed with a WSN – Cloud integration platform called Sensor Cirrus [14, 15, 16]. Sensor Cirrus manages the WSN data using Cloud services and includes the developed FPA for data processing.

Figure 3 illustrates a scheme corresponding to the implemented FPA. The information collected by WSN sensors on the field is stored in a proper database, as it is seen in process (1). Next, in process (2), the application performs a query to catch a sample of fifty days in which the radiation frost has happened. This sample includes all the collected data (temperature, humidity, solar radiation, and wind speed, among others) by the WSNs. Then in process (3) the application retrieves only the FPM input data (T_o, D_o, etc.) from the sample of fifty days. Finally, in process (4) the FPM is executed on Amazon EC2 instances, providing the minimum temperature that would occur the following night (5). Process (4) can be performed using a single Amazon EC2 instance to process the FPM or using several instances working in parallel on a cluster of virtual machines.

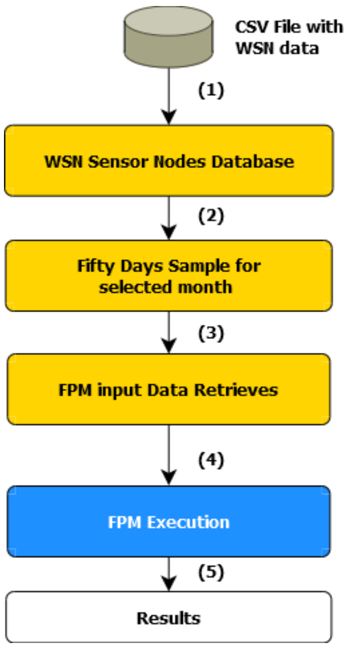


Figure 3. Frost prediction application

Figure 4 illustrates the processing of FPA on a virtual cluster. Within the FPA, processing tasks are distributed equitably on each virtual cluster node. Each job consists of processing a sensor node, therefore in each instance the same number of sensor nodes is processed. Hence, for processing 1 000 sensor nodes in a cluster of 4 instances, the master node of the cluster sends data from 250 sensor nodes to each slave node. Then those data are processed in order to determine the frost occurrence probability.

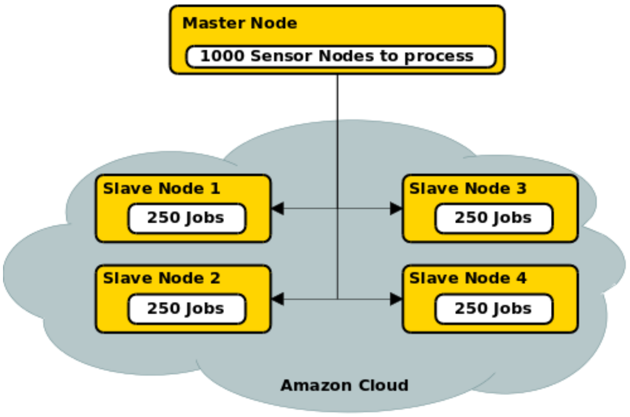


Figure 4. Processing of FPA on a cluster of Amazon EC2 instances

5 PERFORMANCE ESTIMATION MODELS

In this section, we present our models to estimate the performance of EC2 instances for processing FPA. The methodology used to construct the models is the following: first, we execute the FPA in each instance to obtain empirical results of performance metrics, specifically execution time and economic cost. Then we use polynomial expressions and empirical results in order to generate the performance models. Finally, we draw conclusions about the accuracy of the proposed models.

5.1 FPA Execution

The execution consists of running the FPA in different Amazon EC2 instances and measuring the execution time. In order to obtain the average value of the execution time, the procedure is repeated four times for different number of sensor nodes (from 10 to 1 000) in each instance. Finally, we use the average execution time and the Amazon's pricing list to calculate the economic cost required to execute the application.

Table 1 details the four instances to be modelled. Each row in Table 1 represents a different instance type, i.e., m1.small, m1.large, m1.xlarge and c3.xlarge. Each column of the same table indicates the instance characteristics, i.e., number of virtual CPUs (vCPUs), Amazon EC2 Compute Unit (ECU), Memory (expressed in GBytes) and Instance Pricing. Regarding the Amazon's pricing model used in our work, we use the *"on demand"* pricing model. It is worthy of remark that in this paper we do not make an analysis of the accuracy of the minimum temperature predicted by the FPA. However, based on agronomists' experience, we can affirm that an error of $+/-1.5$ Celsius degrees is an acceptable error value to predict frost, and the used FPM meets this requirement.

Amazon EC2 Instance	vCPUs	ECU	Memory [GBytes]	Pricing *on demand* [US$]
m1.small	1	1	1.7	0.047
m1.large	2	4	7.5	0.190
m1.xlarge	4	8	15	0.379
c3.xlarge	4	14	7.5	0.239

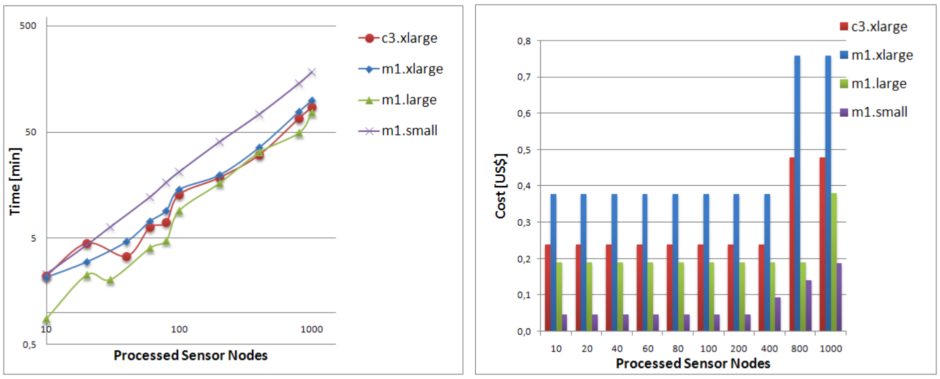
Table 1. Tested Amazon EC2 instances

The application execution allows to obtain empirical performance results in each EC2 instance. This execution was performed using the Screen window manager[2] in each tested Amazon EC2 instances. Screen multiplexes a physical terminal between several virtual terminals. Thanks to Screen, a process executed in a virtual terminal can be run completely and independently of any other terminal process executed in the same physical machine. In this work, we use Screen because it allows to avoid the influence of the SSH connection in the FPA execution, since we have noticed that the SSH connection affects the execution of the application. In some cases, SSH connection delays increase the execution time of frost prediction application; while in others, an interruption in SSH connection halts the application execution. Figure 5 illustrates the empirical results obtained from the FPA execution on each instance. Particularly, Figure 5 a) shows the execution time versus the number of processed sensor nodes, and Figure 5 b) details the economic cost versus the number of processed sensor nodes.

In Figure 5 a), it can be observed that m1.large is the instance that has achieved the shortest execution time for the frost prediction application. In other words, when processing up to 200 sensor nodes, the m1.large instance performance is noticeable. For more than that number of sensors, the m1.large performance becomes similar to the m1.xlarge and c3.xlarge performances. Furthermore, results show that for multiprocessor machines, like m1.large and c3.xlarge, the processing times decrease for 30 and 40 sensor nodes, respectively. Regarding the observed decrease, the one in m1.large instance is lower (about 9 % over the previous calculation) than the one in c3.xlarge instance (20 % compared to the previous point).

Analysis of hardware features of such instances (m1.large and c3.xlarge) shows that they have:

[2] `https://www.gnu.org/software/screen/`



a) Execution times b) Execution costs

Figure 5. Empirical results

1. two and four vCPUs, respectively, and
2. the same RAM memory (7.5 GBytes).

It can therefore be concluded that the decrease of processing times could be due to the load balancing between processors and the access to shared resources such as memory, and buses, among others.

Figure 5 b) shows the empirical economic costs. It can be verified that economic costs are the same from 10 to 400 nodes. Since Amazon sets the pricing of instances per hour of use, the cost is the same provided the processing time is less or equal than one hour. In a like manner, for processing times longer than one hour but shorter than two hours (for example 800 to 1000 nodes), the cost value is similarly doubled and so on.

5.2 Performance Estimation Models

In this subsection, we introduce the proposed models in order to estimate each instance performance when running the FPA in single way. These models are obtained through polynomials up to second degree of the form:

$$t = ax^2 + bx + c$$

where x is the number of sensor nodes processed and t is the estimated execution time. The values of the coefficients a, b and c for each scenario are shown in Table 2. These coefficients are obtained by statistical approximation using polynomials (up to second degree) applied to results obtained in Subsection 5.1.

In order to evaluate the proposed models we calculate the execution time and economic cost for each instance. Figure 6 shows a comparison of the empirical execution times and the ones predicted with our performance models for each tested

Amazon EC2 Instance	a	b	c
m1.small	1.84×10^{-6}	1.78×10^{-1}	1.80
m1.large	6.02×10^{-6}	6.50×10^{-2}	9.98×10^{-1}
m1.xlarge	1.40×10^{-5}	8.24×10^{-2}	2.25
c3.xlarge	1.66×10^{-5}	6.73×10^{-2}	2.59

Table 2. Coefficients of theoretical model of each instance

instance. Figure 7 illustrates the comparison between empirical and theoretical economic costs for the same instances.

Specifically, Figure 6 shows that execution times calculated through the proposed model differ in seconds or few minutes (depending on the instance) from the ones obtained through experiments. Then the proposed models are able predict the results with a reasonably good accuracy.

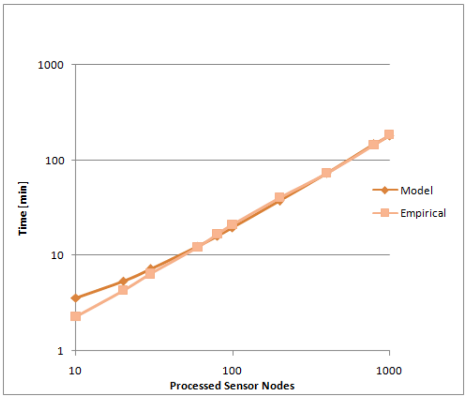
Regarding economic cost, a particular case is when execution times are close to an hour. In this situation, if the execution time calculated by the model is longer than one hour, the costs predicted by the model will be twice those empirical ones. This is because the price of EC2 instances is fixed per hour of use, as it is here in above explained. Likewise, if the model predicts less time than one hour, the corresponding calculated cost would be half of the empirical costs. However, when the proposed model is used, this situation does not happen, so we can say that the accuracy of the models regarding economic cost is suitable.

6 INSTANCE PERFORMANCE IN TYPICAL USE CASE

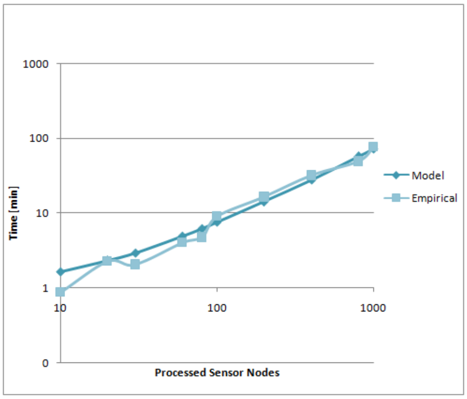
In order to select the instance with the best performance for running FPAs, we present in this section a comparison of a typical use case of frost prediction. The typical use case consists of WSNs deployed in different farms in the Province of Mendoza. The prediction is made for one day of July because this month belongs to the frost season, which begins in April and ends in October.

The FPA runs in a single instance of the Cloud and predicts the minimum temperature, allowing the agronomist's alert. Finally, the agronomist decides if the guard procedure against frosts must be conducted. Frost guard procedure consists in moving the staff to the farm and wait for the specialist's decision – based on data collected in real time – to activate proper defense systems, like heaters, sprinklers or wind turbines.

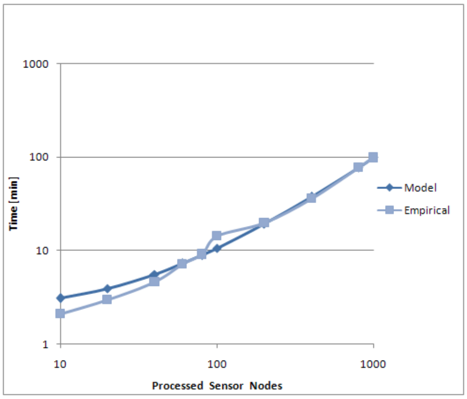
Data Processing time constraint is another aspect to take into consideration. Frost prediction application can only be launched after the measurement of T_0 temperature, which is obtained on the day of the prediction, specifically two hours after sunset. During the month of July in Mendoza province, the sunset takes place at 7.00 pm approximately, then T_0 temperature must be taken at 9.00 pm. Additionally, because of logistics, the frost defense system requires estate farm staff to be alerted before 10.00 pm. According to the above-mentioned reasons, the maximum execution time allowed for the application must be less or equal to one hour.



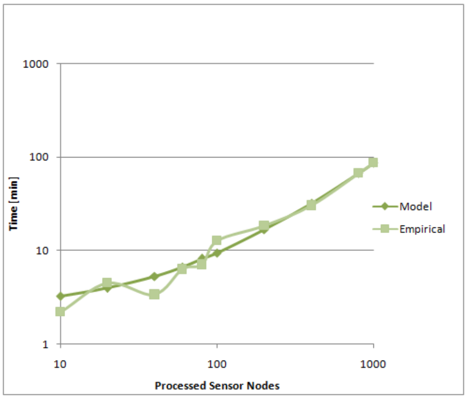
a) M1.small instance



b) M1.large instance



c) M1.xlarge instance



d) C3.xlarge instance

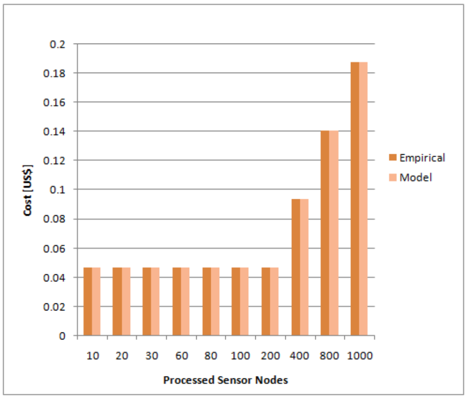
Figure 6. Execution times comparison in tested Amazon EC2 instances

Table 3 shows the execution cost and the number of nodes processed by each EC2 instance model, for one hour predicted by the proposed performance models.

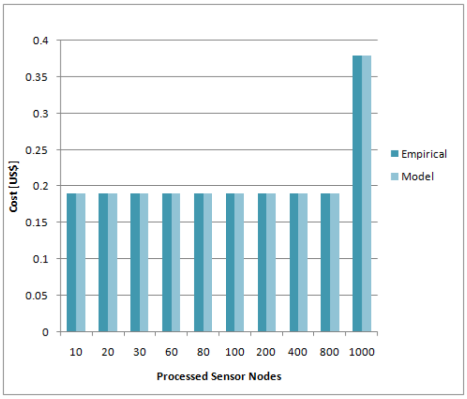
Amazon EC2 Instance	Nodes	Economic Cost [US$]
m1.small	324	0.047
m1.large	841	0.190
m1.xlarge	632	0.379
c3.xlarge	722	0.239

Table 3. Processed sensor nodes in one hour predicted by performance models

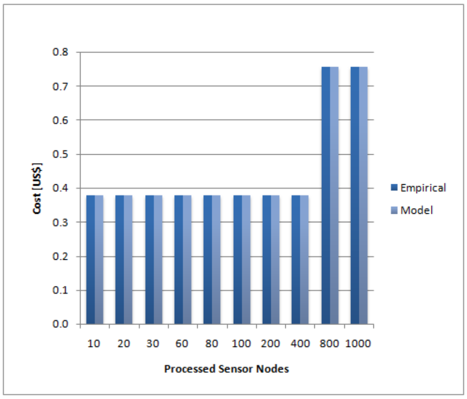
Results demonstrate that the instance m1.large is the most suitable machine for this application type. The reason is because the m1.large is the machine that



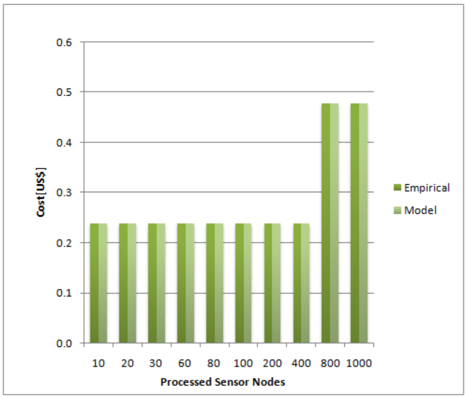
a) M1.small instance



b) M1.large instance



c) M1.xlarge instance



d) C3.xlarge instance

Figure 7. Economic costs comparison in tested Amazon EC2 instances

can process the largest number of sensor nodes in one hour and its economic cost is smaller than those belonging to the m1.xlarge and c3.xlarge instances. Therefore, results presented in Table 3 suggest that m1.small instance can be used in parallel for processing more sensor nodes at a lower economic cost than what the m1.large machine is able to. Thus, the need to conduct more experiments in order to study the model accuracy of this instance in the typical use case becomes apparent.

7 ACCURACY OF MODELS FOR TYPICAL USE CASE

Execution times and costs are predicted by the proposed models through statistical approaches, hence it is necessary to perform experiments with the purpose of studying the accuracy of these models for a specific number of sensor nodes.

This section discusses the theoretical performance model accuracy of m1.small and m1.large instances for the typical use case presented in Section 6.

This section is organized as follows. Subsection 7.1 details the execution of the FPA for the number of sensor nodes that can be processed in each instance in one hour. Next, Subsection 7.2 presents a comparison between the execution times obtained in Subsection 7.1 and the ones predicted through theoretical models. The main objective of the comparison is to calculate the errors of the models for the typical use case. Then, based on these errors, we can correct the number of sensor nodes that can be processed in one hour. Finally, in Subsection 7.3 the FPA is executed for the corrected number of sensor nodes in each instance.

7.1 FPA Execution for Best Instances

This subsection details the results obtained through the execution of the FPA in m1.small and m1.large instances. These instances are selected because they can process more sensor nodes than the m1.xlarge and c3.xlarge instances, in an isolated way (m1.large) or parallel way (m1.small) at lower prices. The experiments are conducted in each instance (running single) for the number of sensor nodes that can be processed in one hour, predicted by theoretical performance models. In order to obtain the average value of execution time, the FPA is executed four times in each Amazon EC2 instance.

Table 4 presents the average execution times obtained by the execution of the FPA in each instance.

Amazon EC2 Instance	Nodes	Average Execution Time [min]
m1.small	324	48.77
m1.large	841	53.30

Table 4. Empirical execution times obtained for typical use case

7.2 Comparison of Models and Experimental Results

In order to determine the error of the proposed models, Table 5 shows a comparison between the empirical execution time and theoretical execution time obtained in the m1.small and m1.large instances. Column two details the number of sensor nodes processed in each instance, columns three and four show the execution times predicted with theoretical models and experiments, respectively. Finally, column five presents the error percentage observed between empirical results and theoretical models for each instance.

Experiments conducted in m1.small instance detailed in row two of Table 5 show that the error between the theoretical time and empirical time is 18.83 %. Results of experiments conducted in m1.large instance showed in row three of Table 5 present an error of 11.16 % between the empirical time and theoretical execution time.

Amazon EC2 Instance	Nodes	Theoretical Execution Time [min]	Empirical Execution Time [min]	Error [%]
m1.small	324	60	48.77	18.83
m1.large	841	60	53.30	11.16

Table 5. Comparison between empirical and predicted execution times for typical use case

7.3 Execution of FPA for Theoretical Corrected Results

Errors observed in our models allow the correction of the number of sensor nodes that can process each instance in one hour (typical use case). Table 6 shows the corrected number of sensor nodes that can be processed in m1.small and m1.large instances in a typical use case. Column two of Table 6 presents the predicted number of sensor nodes, Column three details the corrected number of sensor nodes and Column shows four errors used for correcting the number of sensor nodes.

Amazon EC2 Instance	Predicted Number of Sensor Nodes	Corrected Number of Sensor Nodes	Error [%]
m1.small	324	385	18.83
m1.large	841	934	11.16

Table 6. Correction of processed sensor nodes for typical use case

Once the number of sensor nodes to process in an hour is corrected, we conduct the execution of FPA for those values. Table 7 shows execution times obtained through experiments conducted in m1.small and m1.large instances for corrected sensor nodes.

Amazon EC2 Instance	Processed Sensor Nodes	Empirical Execution Times [min]
m1.small	385	54.88
m1.large	934	58.40

Table 7. Processed sensor nodes for typical use case

Row two of Table 7 presents the number of experiments conducted in m1.small instance. Results show that this machine can process 385 sensor nodes in 54.88 minutes, reason why we can affirm that theoretical models have been fixed correctly for typical use case in m1.small instance.

Row three of Table 7 details the execution times when FPA is executed considering the new number of nodes in m1.large instance. Like the m1.small instance, the obtained execution time (58.40 min) has validated the correction made in the theoretical model.

8 FPA EXECUTION IN VIRTUAL CLUSTERS

Experiments conducted in Subsection 7.3 determine the Amazon EC2 instance that can process more sensor nodes when working in an isolated way (m1.large). However, the above mentioned experiments suggest that m1.small instances can be used in parallel through a virtual cluster for processing larger number of sensor nodes at lower cost than the instance m1.large. In this section, we perform experiments in order to determine how many sensor nodes can be processed in m1.small instances when they are executed in parallel. The execution is conducted considering a virtual cluster for which the hour price is lower than a m1.large instance hour price.

The experimental methodology is the following: first, we consider the number of sensor nodes that can be processed by each instance in one hour, working in an isolated way. Then we implement a virtual cluster composed of the number of virtual machines whose total cost is less than 0.19 US$ (an m1.large instance's hour price). Next, the FPA is processed in each node of the virtual cluster for the number of sensor nodes that can process each slave node (instance) in an hour. Finally, the run time is measured in each execution. The experiment is achieved four times in the virtual cluster in order to obtain the average execution time.

Table 8 shows the number of sensor nodes processed using the m1.small instance on a virtual cluster. Second and third columns show the number of processed sensor nodes and slave nodes (instances) used in parallel way, respectively. Fourth column indicates the recorded execution time, and finally fifth column shows the resulting economic cost of FPA executed in the virtual cluster.

Amazon EC2 Instance	Sensor Nodes	Instances Executed in Parallel	Execution Time [min]	Economic Cost [US$]
m1.small	1 540	4	57	0.188

Table 8. Execution times of virtual cluster in a typical use case

Making a comparison with the results obtained for m1.large instance from Table 7, the results in Table 8 indicate that m1.small cluster can process 606 more sensor nodes than one m1.large instance in one hour at lower economic cost. This result is obtained using four m1.small instances in parallel way. Furthermore, the recorded average execution time is 57 minutes, which fulfills the time constraint requirement defined in the typical use case.

9 CONCLUSIONS AND FUTURE WORKS

In this paper, we have studied the use of Amazon EC2 instances for frost prediction. In the first place, we have presented an application developed to predict the occurrence of frost based on data collected on-field by Wireless Sensor Networks. This application has been used to generate performance models of different Amazon EC2 instances when they were applied to process frost prediction applications. The metrics used to evaluate the performance were execution time and economic cost.

In order to generate the models, we have conducted experiments in four test scenarios. Each scenario has corresponded to a different Amazon EC2 instance. The obtained model of each instance was compared with empirical data of the frost prediction application executions. From the results, we have concluded that the proposed models were suitable to estimate both the execution time and the economic cost. However, the proposed models have presented some problems when they have been used to predict economic cost and execution time in a specific number of sensor nodes. That was why we have performed more tests in a typical application case, allowing us to determine the models' errors. These experiments have been only performed for the instances with better performance working in a single (m1.large) or parallel way (m1.small).

Experiments conducted in a typical use case have showed that models' error were 18.83 % and 11.16 % for m1.small instance and m1.large instance, respectively. Once the error of each model has been determined, the number of sensor nodes that could be processed in the typical use case has been corrected. Then we have performed tests to validate if the corrected number of sensor nodes were fulfilling the time constraints of typical use case. Results have showed that the models were fixed correctly, then m1.small and m1.large instances were able to process 385 and 934 sensor nodes, respectively, in an hour.

A typical use case has been used to determine which instance working in an isolated way was more suitable for processing frost prediction applications. Results have showed that the m1.small instance and m1.large were the correct instances to use for WSNs made up by 140–385 and by 386–934 sensor nodes, respectively.

While the m1.xlarge and c3.xlarge were the instances with the best performance, we have not observed important differences in the performance when comparing them to the other tested instances. Moreover, if we also consider the m1.xlarge and c3.xlarge high costs, their use is not recommended for this type of applications.

The execution of frost prediction application in individual machines have allowed us to determine that for the case of WSNs – made up by more than 934 sensor nodes – multiple EC2 instances were to be used in parallel way to run the application. Hence, we have performed an experiment on a virtual cluster. The virtual cluster was composed of four m1.small instances. These instances have been selected because they were able to process more than 934 sensor nodes running in parallel way at a cost lower than the best instance running in single way (m1.large) in the typical use case.

Results of experiments on virtual clusters have showed that the m1.large cluster can process 1 540 sensor nodes in one hour. Consequently, taking into account the above mentioned number of WSNs sensor nodes, we can affirm that the m1.large cluster is more suitable because it allows the processing of data from more sensor nodes than one single m1.large instance.

In conclusion, this paper demonstrates that second degree polynomials are a simple and suitable way to estimate the performance of Amazon's EC2 instances. Regarding future works: first, we are going to improve the frost prediction application for parallel execution. Next, we will continue the validation of our models by studying the processing of the frost prediction application on virtual clusters scheduled with algorithms based on meta-heuristics and managed by specific Cloud tools like Star Cluster. The purpose of these future experiments is to extend the proposed models to estimate not only how many machines are needed to optimize the execution time/economic cost relationship for frost prediction applications but also how these machines should be managed.

Acknowledgments

The authors acknowledge the financial support provided by the Argentinean Agency for R & D activities (ANPCyT) through projects PAE-PID 146 and PICT-2012-2731, and to the Spanish National Plan for Research, Development and Innovation for financial support provided by the project TIN2012-31518 (ServiceCloud). Lucas Iacono also thanks the Consejo Nacional de Investigaciones Científicas y Técnicas of Argentina (CONICET) for the postdoctoral scholarship, as well as Norberto Faraz for the source code provided for the frost prediction application, and finally we are grateful for the Bec.AR program.

REFERENCES

[1] ur Rehman, A.—Abbasi, A. Z.—Islam, N.—Shaikh, Z. A.: A Review of Wireless Sensors and Networks' Applications in Agriculture. Computer Standards and Interfaces, Vol. 36, 2014, No. 2, pp. 263–270.

[2] Ahmed, K.—Gregory, M.: Integrating Wireless Sensor Networks with Cloud Computing. Seventh International Conference on Mobile Ad-Hoc and Sensor Networks (2011 MSN), IEEE, 2011, pp. 364–366, doi: 10.1109/MSN.2011.86.

[3] Allen, C. C.: A Simplified Equation for Minimum Temperature Prediction. Monthly Weather Review, Vol. 85, 1957, No. 119–120.

[4] Amazon EC2: Amazon Web Services. 2012, available at: `http://aws.amazon.com/es/ec2/`.

[5] Bormann, C.—Mulligan, G.—Arkko, J.—Townsley, M.—Schumacher, C.: IPv6 over Low Power WPAN (6lowpan). IETF Working Group, 2006, available at: `http://datatracker.ietf.org/wg/6lowpan/charter/`.

[6] Buyya, R.—Yeo, C.—Venugopal, S.—Broberg, J.—Brandic, I.: Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5^{th} Utility. Future Generation Computer Systems. Vol. 25, 2009, No. 6, pp. 599–616.

[7] Foster, I.—Zhao, Y.—Raicu, I.—Lu, S.: Cloud Computing and Grid Computing 360-Degree Compared. Grid Computing Environments Workshop 2008 (GCE '08), IEEE, 2008, pp. 1–10, doi: 10.1109/GCE.2008.4738445.

[8] Google: Google App Engine. 2013, available at: `https://developers.google.com/appengine/`.

[9] Google: Google Docs. 2013, available at: `https://drive.google.com/`.

[10] Gubbi, J.—Buyya, R.—Marusic, S.—Palaniswami, M.: Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. Future Generation Computer Systems, Vol. 29, 2013, No. 7, pp. 1645–1660.

[11] Higuera, J. E.—Polo, J.: IEEE 1451 Standard in 6LoWPAN Sensor Networks Using a Compact Physical-Layer Transducer Electronic Datasheet. IEEE Transactions on Instrumentation and Measurement, Vol. 60, 2011, No. 8, pp. 2751–2758, doi: 10.1109/TIM.2011.2129990.

[12] Hirafuji, M.—Yoichi, H.—Kiura, T.—Matsumoto, K.—Fukatsu, T.—Tanaka, O. et al.: Creating High-Performance/Low-Cost Ambient Sensor Cloud System Using OpenFS (Open Field Server) for High-Throughput Phenotyping. Proceedings of 2011 SICE Annual Conference (2011 SICE), IEEE, 2011, pp. 2090–2092.

[13] Hori, M.—Kawashima E.—Yamazaki, T.: Application of Cloud Computing to Agriculture and Prospects in Other Fields. Fujitsu Scientific and Technical Journal, Vol. 46, 2010, No. 4, pp. 446–454.

[14] Iacono, L.: Acceso Remoto a Redes de Sensores Inalámbricas Mediante Tecnologías de Computación Distribuída. Thesis Proposal, Facultad de Ingeniería, Universidad de Mendoza, 2013 (in Spanish).

[15] Iacono, L.: Sensor Cirrus. June 2014, `https://sites.google.com/site/sensorcirrus/`.

[16] Iacono, L.—García Garino, C.—Marianetti, O.—Párraga, C.: Wireless Sensor Networks: A Software as a Service Approach. In: García Garino, C., Printista, M. (Eds.): Prospective and Ongoing Projects, VI Latin American Symposium on High Performance Computing (HPCLatAm 2013), Mendoza, Argentina, 2013, pp. 184–195.

[17] IEEE: IEEE Standard for Information Technology – Telecommunications and Information Exchange Between Systems – Local and Metropolitan Area Networks – Specific Requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs), IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003), 2006, pp. 1–305.

[18] Lee, K.—Hughes, D.: System Architecture Directions for Tangible Cloud Computing. International Workshop on Information Security and Applications (IWISA 2010), Vol. 25, Qinhuangdao, China, 2010, doi: 10.1109/CDEE.2010.57.

[19] Lee, K.—Murray, D.—Hughes, D.—Joosen, W.: Extending Sensor Networks into the Cloud Using Amazon Web Services. 2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications (NESEA), IEEE, 2010, pp. 1–7, doi: 10.1109/NESEA.2010.5678063.

[20] Mazurek, E.—Fukuda, M.: A Parallelization of Orchard Temperature Predicting Programs. 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PacRim), IEEE, 2011, pp. 179–184, doi: 10.1109/PACRIM.2011.6032889.

[21] Dinh, T.—Kim, Y.: An Efficient Interactive Model for On-Demand Sensing-as-a-Services of Sensor-Cloud. Sensors (Basel), Vol. 16, 2016, No. 7, Art. No. 992, 28 pp.

[22] Microsoft: Windows Azure. 2013, available at: `http://www.windowsazure.com/es-es/`.

[23] Oliveira, L. M. L.—Rodrigues, J. J. P. C.: Wireless Sensor Networks: A Survey on Environmental Monitoring. Journal of Communications, Vol. 6, 2011, No. 2, pp. 143–151, doi: 10.4304/jcm.6.2.143-151.

[24] Pierce, F. J.—Elliott, T. V.: Regional and On-Farm Wireless Sensor Networks for Agricultural Systems in Eastern Washington. Computers and Electronics in Agriculture, Vol. 61, 2008, No. 1, pp. 32–43, doi: 10.1016/j.compag.2007.05.007.

[25] Reddy, S.—Chen, G.—Fulkerson, B.—Kim, S. J.—Park, U.—Yau, N.—Cho, J.—Hansen, M.—Heidemann, J.: Sensor-Internet Share and Search: Enabling Collaboration of Citizen Scientists. Proceedings of the ACM Workshop on Data Sharing and Interoperability on the World-Wide Sensor Web, 2007, pp. 11–16.

[26] Snyder, R. L.—de Melo-Abreu, J. P.: Frost Protection: Fundamentals, Practice and Economics. Vol. 1, Food and Agriculture Organization of the United Nations (FAO), 2005.

[27] Yoo, S. E.—Kim, J. E.—Kim, T.—Ahn, S.—Sung, J.—Kim, D.: A2S: Automated Agriculture System Based on WSN. IEEE International Symposium on Consumer Electronics (ISCE 2007), 2007, pp. 1–5.

[28] Zerger, A.—Viscarra Rossel, R. A.—Swain, D. L.—Wark, T.—Handcock, R. N.—Doerr, V. A. J.—Bishop-Hurley, G. J.—Doerr, E. D.—Gibbons, P. G.—Lobsey, C.: Environmental Sensor Networks for Vegetation, Animal and Soil Sciences. International Journal of Applied Earth Observation and Geoinformation, Vol. 12, 2010, No. 5, pp. 303–316, doi: 10.1016/j.jag.2010.05.001.

[29] ZigBee Alliance: ZigBee Specification. ZigBee Document 053474r13, 2006, pp. 344–346.

Lucas E. Iacono received his Engineering degree in electronics and electrical from Universidad de Mendoza, Argentina, in 2007 and his Ph.D. from Universidad de Mendoza in 2015. Currently, he is Posdoctoral Fellow at the Consejo Nacional de Investigaciones Científicas y Técnicas of Argentina (CONICET). His research interests include internet of things technologies, particularly WSNs and cloud computing applied to frost prediction and environmental monitoring.

José Luis Vázquez Poletti is Tenure-Track Assistant Professor at UCM, from where he received his Ph.D. in computer architecture. His research interests include high-performance computing, cloud computing, and grid technology, focusing on their application to real life problems.

Carlos García Garino graduated in engineering at the University of Buenos Aires, Argentina in 1978 and received his Ph.D. degree from Universidad Politécnica de Cataluña, Barcelona, Spain in 1993. Currently he is Full Professor at the School of Engineering and Head of the ITIC Research Institute, Universidad Nacional de Cuyo, Argentina. His research interests include computational mechanics, computer networks, and distributed computing. He has more than 50 papers published in scientific journals and proceedings of international conferences carried out in Argentina, Brazil, Chile, Canada, Spain, France, Portugal, Belgium and Japan.

Ignacio Martín Llorente is co-founder and Director of OpenNebula, and Full Professor at UCM. He received his Ph.D. in computer architecture from UCM and an executive MBA from the Instituto de Empresa. His research interests include high-performance computing, virtualization, cloud computing, and grid technology. He is a senior member of IEEE.

Computing and Informatics, Vol. 37, 2018, 838–864, doi: 10.4149/cai_2018_4_838

COST-EFFICIENT SCHEDULING FOR DEADLINE CONSTRAINED GRID WORKFLOWS

Alireza Dehlaghi-Ghadim

School of Electrical and Computer Engineering
University of Tehran
Tehran, Iran
e-mail: a.dehlaghi@ut.ac.ir

Reza Entezari-Maleki

School of Computer Science
Institute for Research in Fundamental Sciences (IPM)
Tehran, Iran
e-mail: entezari@ipm.ir

Ali Movaghar

Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
e-mail: movaghar@sharif.edu

Abstract. Cost optimization for workflow scheduling while meeting deadline is one of the fundamental problems in utility computing. In this paper, a two-phase cost-efficient scheduling algorithm called *critical chain* is presented. The proposed algorithm uses the concept of slack time in both phases. The first phase is deadline distribution over all tasks existing in the workflow which is done considering critical path properties of workflow graphs. *Critical chain* uses slack time to iteratively select most critical sequence of tasks and then assigns sub-deadlines to those tasks. In the second phase named mapping step, it tries to allocate a server to each task considering task's sub-deadline. In the mapping step, slack time priority in selecting ready task is used to reduce deadline violation. Furthermore, the algorithm tries to locally optimize the computation and communication costs of sequential tasks exploiting dynamic programming. After proposing the scheduling algorithm, three measures for the superiority of a scheduling algorithm are introduced, and the proposed algorithm is compared with other existing algorithms considering the measures. Results obtained from simulating various systems show that the proposed algorithm outperforms four well-known existing workflow scheduling algorithms.

Keywords: Grid computing, workflow, slack time, critical path, cost-based scheduling

Notations

T	Set of all tasks in the application
E	Set of all dependencies in the application
t_i	Task i
e_{ij}	Dependency between task i and task j
δ	Deadline of the application
S	Set of servers
s_i	Server i
T_{ij}	Processing time of task t_i on server s_j
C_{ij}	Processing cost of task t_i on server s_j
$S(t_i)$	The server allocated to execute task t_i
$T_{iS(t_i)}$	Execution time of task t_i on $S(t_i)$
$\text{Delay}(t_i, t_j)$	Data transmission time between task t_i and task t_j on the link between $S(t_i)$ and $S(t_j)$
$imm\_preds(t_i)$	All tasks in the workflow graph in which t_i is their immediate successor
$imm\_succs(t_i)$	All tasks in the workflow graph in which t_i is their immediate predecessor
$\text{MET}(t_i)$	Minimum time for the execution of task t_i on the fastest server
$\text{MTT}(e_{ij})$	Minimum data transmission time between task t_i and task t_j
$\text{EST}(t_i)$	Earliest start time of task t_i
$\text{LFT}(t_i)$	Latest finish time of task t_i
$\text{ST}(t_i)$	Slack time of task t_i
CST	Chain start time
CFT	Chain finish time
SST	Schedule start time
SFT	Schedule finish time
NC	Normalized cost
θ	Deadline factor
T_{min}	Minimum execution time of the application

1 INTRODUCTION

Grid is an infrastructure with the aim of solving high scale problems in science, economy, aerology, engineering and many other fields [1]. Resource sharing is one of the most significant advantages of grid computing [2]. The most important resources shared in grids include CPU, main memory, secondary memory, network bandwidth, and data. Traditional resource management techniques provide no incentive for users to share resources fairly. Consequently, to support different levels of Quality of Service (QoS) and manage priority between user applications, utility grid computing has been emerged [3, 4]. In this paradigm, users have to pay for each time they use servers with specific QoS. How to allocate grid servers to the tasks to satisfy the specific needs is one of the important challenges in this area.

This paper focuses on workflow scheduling with the aid of heuristics. In this case, workflows are composed of several tasks with partial order, in the way that some tasks have control or data dependencies on the others. Many complex applications in different domains such as e-science as bioinformatics and astronomy, and e-business can be modeled as workflows [5]. To solve the applications, the resulted workflows need to be processed, so the tasks should be executed based on their dependencies [6]. We can describe workflows with Directed Acyclic Graphs (DAGs) in which each node in DAG represents a specific task in the corresponding workflow. Therefore, the scheduling problem can be stated as assigning a DAG of tasks to the limited processing units according to their requirements and transposition constraints. To solve this type of scheduling problems, two different approaches can be used: *approximation* and *heuristic*. In the approximate algorithms, since it is unlikely that there can ever be efficient polynomial-time exact algorithms solving NP-hard problems, one settles for polynomial-time sub-optimal solutions so called approximation, which uses formal computational models to obtain sufficiently good solution instead of searching the entire solution space for an optimal solution. Heuristic represents the class of algorithms which makes more realistic assumptions about a priori knowledge concerning process and system loading characteristics [7, 8].

Generally, mapping tasks on distributed servers is an NP-hard problem, and workflow scheduling as an optimization problem produces large scheduling overhead, especially for problems with two-dimensional constraints such as time and cost [9, 10]. The most well-known goal considered for workflow scheduling is minimizing the makespan of the application. Although many research papers have addressed this problem [9, 11], in the economic scheduling, cost reduction along with satisfying the deadline is very important and that should be taken into consideration in workflow scheduling [12]. Consequently, traditional approaches for scheduling tasks in grid community are no longer suitable for utility grids. Therefore, some new methods have been proposed in the past years to fulfill this requirement [4, 6, 9, 10, 13, 14, 15, 16, 17, 30]. Many recent approaches in workflow scheduling consider critical path as a hint to assign sub-deadlines to the tasks [11, 13, 17, 18, 19], but deadline distribution with those methods is not efficient enough to decrease deadline violations. Another disadvantage of the previously presented scheduling methods is the lack of priority between tasks in the mapping step. To overcome these shortcomings, we develop a new scheduling algorithm with two steps. In the first step, an efficient method for initial distribution of tasks deadlines is presented, and in the second step, scheduling priority for the task with minimum slack time is considered to reach a better result. In order to evaluate the proposed algorithm and compare it with others, we simulate three types of well-known workflows under various assumptions and system configurations. Simulation results show the advantage of the proposed algorithm in comparison with four most-cited recent algorithms.

The remainder of this paper is organized as follows. In Section 2, some related work done on scheduling problem, especially on workflow scheduling in grid environments, is presented. In Section 3, the scheduling problem in general case and its details in our context are described. The main proposed *critical chain* algorithm together with other sub-methods is presented in Section 4. In Section 5, experimental results obtained from simulation are given. Finally, Section 6 concludes the paper and presents the future work which can be done in this research field.

2 RELATED WORK

There are several research works addressing the problem of mapping workflows on multiprocessors [18, 20, 21]. However, some constraints like communication delays and specifically budget issues on economic grids make the previously done research work on multiprocessor systems inefficient when they are applied to the grids.

Foster et al. [22] have described a General-purpose Architecture for Reservation and Allocation (GARA) that supports QoS specification. Dogan et al. [23] have studied the scheduling of a set of independent tasks considering some QoS factors such as reliability, security and timeliness. Tabbaa et al. [24] have presented a new scheduling algorithm for DAG applications in clusters. The algorithm considers the failure of resources and tries to schedule tasks to the cluster servers to tolerate the faults occurred in the system. Entezari-Maleki et al. [25] have proposed a genetic-based task scheduling algorithm to minimize the makespan of grid applications. The algorithm proposed in [25] only considers the makespan as a QoS factor. However, there are few papers considering the cost of scheduling as a QoS factor. Kardani-Moghaddam et al. [26] have proposed a hybrid genetic algorithm and variable neighborhood search which uses a fitness function to balance between makespan and execution cost of each scheduling solution. Agrawal et al. [27] have explored linear workflow scheduling for linear workflows, and found an approximation algorithm to maximize throughput in the one-port model. Moreover, they proved that finding a schedule respecting a given period and a given latency is NP-hard.

Yu et al. [9] have proposed the deadline-MDP algorithm that divides a DAG to partitions and then distributes the deadline over the partitions. Finally, deadline-MDP algorithm tries to locally optimize the cost for each partition using Markov models. It has been shown that deadline-MDP algorithm outperforms previous methods such as DTL and greedy cost [9, 10]. The genetic algorithm was used to optimize the time of scheduling under budget constraint in [6]. Zhao et al. [28] have proposed two algorithms to schedule workflows with budget constraints. The first algorithm initially schedules all tasks to faster servers and then reschedules some tasks to meet the desires. Similarly, the second algorithm assigns each task to its cheapest server, and reschedules the tasks to the faster and more expensive servers as long as the budget is acceptable. According to Yuan et al. [29], Deadline Bottom Level (DBL) is a simple and efficient heuristics for workflow scheduling. In this method, all tasks are divided into several groups based on their bottom level with a backward method. The overall deadline is distributed over the groups considering maximum processing cycle of tasks in that group. All tasks in a group inherit a unique deadline of the corresponding group. Unlike the DTL method [9], the start time of each task is determined by the latest finish time of its immediate predecessors instead of the finish time of the last task in the previous level. Although DBL and DTL are effective and efficient, these algorithms show poor performance in firm deadlines. Yuan et al. [10] have presented the Deadline Early Tree (DET) method. First, they create *Early Tree* which is a spanning tree for primary schedule. Then they exploit dynamic programming to assign time window to each critical task, and consequently, find time window for non-critical tasks. Finally, the method tries to assign cheaper servers to each task according to its time window. The number of servers was assumed to be unlimited which is unrealistic assumption in most cases.

Cost-effective scheduling of deadline-constrained applications have been also investigated in hybrid clouds [15, 30, 31, 32, 33, 34]. Henzinger et al. [15] have designed a framework to handle cost-time trade-off in economic workflow scheduling called FlexPRICE. They tried to present the cost-time curve to enable users to select the appropriate deadline with an acceptable price. In fact, FlexPRICE was presented to solve cloud workflow programming, but the type of the problem is similar to the grid computing. Fard et al. [35] have introduced a pricing model and a truthful mechanism for scheduling single tasks considering monetary cost and completion time. With respect to the social cost of the mechanism, they extended the mechanism for dynamic scheduling of scientific workflows. Calheiros et al. [31] have presented an architecture for coordinated dynamic provisioning and scheduling which is able to cost-effectively complete applications within their deadlines. They considered the whole organization workload at individual tasks level, and their accounting mechanism was used to determine the share of the cost of utilization of public cloud resources to be assigned to each user. Poola et al. [30] considered deadline and budget constraints as QoS demanded by users, and tried to design a robust algorithm for scheduling of scientific workflows.

Abrishami et al. [13] have proposed a partial critical path scheduling based on properties of critical path. In deadline assignment step, the deadline is distributed over tasks, and then the cost of each allocation is locally optimized to provide the best possible result in each allocation. For deadline distribution, the method iteratively selects a sequence of tasks in the DAG and assigns the deadline to each member of that sequence. For this assignment, authors apply three different policies on the deadline distribution method: optimized policy, decrease cost policy, and fair policy. The optimized policy iteratively tests all feasible assignments and selects the best one. It is obvious that this approach is time consuming, and it is not feasible for large-scale problems. The decrease policy is based on a greedy method which tries to approximate the optimized policy. In this policy, each task is assigned to the fastest server, and it is tried to decrease the cost by rescheduling a task to a cheaper server. The fair policy is similar to the decrease policy except that starting from the first task towards the last task in path, it substitutes the assigned server with the next slower server without exceeding sub-deadline. This procedure continues iteratively until no substitution can be made. According to the results reported in [13], the proposed algorithms show high performance in absence of server limitation.

3 PROBLEM DEFINITION

Directed Acyclic Graph (DAG) is one of the most acceptable models to represent workflow applications. Let $G(T, E)$ denote a DAG representing an application where T is the task set $T = \{t_1, t_2, \dots, t_n\}$ in which n is the number of all tasks in the application. Moreover, edge set E represents the edges of the related DAG and shows the control or data dependencies between the tasks. The notation e_{ij} denotes an edge from the vertex t_i to t_j, and means that the task corresponding to the vertex t_j requires input data or command produced by execution of task t_i. Suppose that all tasks are topologically numbered in which each arc demonstrates the priority of $i < j$, means that execution of task t_j only depends on the tasks with lower numbers. The tasks having no input (output) edges are named *entry* (*exit*) tasks. For simplicity and without loss of generality, we suppose that always there is only one entry task in the application. If an application has more than one entry task, we can simply add a *dummy task* (a task that requires no processing) to it to produce our DAG of interest. Similarly, we can do the same for exit task in the graph. The number attached to each node represents the processing cycle of the corresponding task in the form of Million Instructions (MI). Also, the number attached to each arc e_{ij} shows the amount of data which should be sent from t_i to t_j. Figure 1 shows an example of DAG representation. In the graph represented in Figure 1, a node with index of i shows task t_i.

A service model in the utility grid computing consists of Grid Service Providers (GSPs) in which each of them provides some servers with specific QoS. The cost of processing in each server is proportional to the speed of process which means that if the scheduler allocates a faster server to execute a task, the user has to pay more cost [9]. Each server supports limited number of task types. We consider each of GSPs as a grid node. Assume that there are m service providers represented by set S where $S = \{s_1, s_2, \dots, s_m\}$. Hence, for each task, there are several candidate servers which can execute the task. Assume T_{ij} is the processing time of task t_i executed on server s_j, and C_{ij} is its corresponding processing cost. If s_j is not capable of processing t_i, we consider both T_{ij} and C_{ij} to simply being infinity. It is assumed that dummy tasks can be processed on any server.

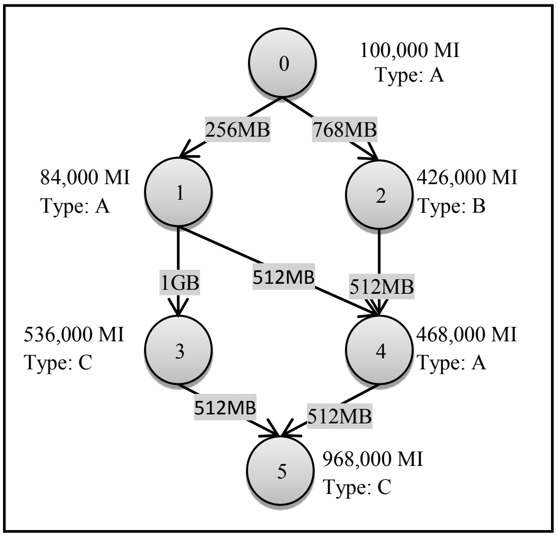


Figure 1. DAG representation of a sample workflow

In this paper, two well-known QoS measures in grids, execution time and cost, are considered. Therefore, our objective in this paper is to assign an appropriate server to the tasks to execute them with the goal of minimizing the overall execution cost while both tasks' precedence and application deadline are taken into account. To achieve this, we can consider workflow scheduling as an optimization problem with trade-off between time and cost [36]. Let δ denote a given deadline showing the latest possible finish time of the application or exit task. Let st_i and f_i denote the start and finish times of task t_i, respectively. Therefore, the workflow scheduling problem can be formulated as Equation (1).

$$\min \Sigma_{i \in T} \Sigma_{1 \le k \le m} C_{ik} x_{ik}$$

$$S.t. \begin{cases} \Sigma_{1 \le k \le m} x_{ik} = 1, & i \in T, \\ f_i < \delta, & i \in T, \\ f_i - st_i = T_{iS(t_i)}, & i \in T, \\ st_i > f_j + \mathrm{Delay}(t_j, t_i), & j \in imm\_preds(i), \\ x_{ik} \in \{0, 1\}, & 1 \le i \le n, 1 \le k \le m, \\ S(t_i) = S(t_j) \Rightarrow (st_i > f_j) \vee (f_i < st_j), & i, j \in T \end{cases} \tag{1}$$

where

$$x_{ij} = \begin{cases} 1, & t_i \text{ is assigned to } s_j, \\ 0, & \text{otherwise.} \end{cases}$$

The constraint $\Sigma_{1 \leq k \leq m} x_{ik} = 1$ in Equation (1) checks to make sure that there is a unique executor for each task. Similarly, condition $f_i < \delta$ ensures meeting the overall deadline. Moreover, the constraint $f_i - st_i = T_{iS(t_i)}$ checks the feasibility of task execution on the server in a given time slice, where $S(t_i)$ is the server assigned to execute task t_i, and $T_{iS(t_i)}$ is execution time of task t_i on $S(t_i)$. Each task would be executed on a resource only if its required data is transferred to the resource. This constraint is checked by $st_i > f_j + \text{Delay}(t_j, t_i)$, where $\text{Delay}(t_j, t_i)$ is data transmission time on the link between $S(t_i)$ and $S(t_j)$ which is computed as Equation (2).

$$\text{Delay}(t_i, t_j) = \frac{e_{ij}}{\text{bandwidth}(S(t_i), S(t_j))} \tag{2}$$

where $\text{bandwidth}(S(t_i), S(t_j))$ denotes the bandwidth of the link between servers executing tasks t_i and t_j.

In our model, it is considered that the number of servers is limited and some of the servers are busy in some cases, so they cannot be allocated to the tasks. The constraint $S(t_i) = S(t_j) \Rightarrow (st_i > f_j) \vee (f_i < st_j)$ checks if the same server is allocated to execute both tasks t_i and t_j. If it is, the start time of one of the tasks (e.g., task t_j) has to be greater than the finish time of the another one (e.g., task t_i). De et al. [36] showed that the time-cost optimization problem for DAG scheduling is a Discrete Time-Cost Trade-off Problem (DTCTP). DTCTP is an NP-hard problem, and the best-known solutions use dynamic programming, and branch and bound method to solve the problem. Unfortunately, these solutions are extremely time-consuming when the number of tasks and/or servers gets larger.

4 THE PROPOSED ALGORITHM

In order to efficiently schedule the tasks on the servers, we need an initial estimation of execution times of the tasks on servers. This estimation could help us to identify critical tasks of the application and schedule them on fast servers to meet the overall deadline of the application. So, in the first step, we propose an algorithm to divide the deadline on all tasks. After applying the algorithm of the first step, each task will have its own deadline in which the task should be processed before that deadline. The algorithm of the first step works based on the concept of *slack time* that is well-known in multiprocessor scheduling community. We use slack time in the deadline distribution algorithm to obtain the critical path not only for the whole graph, but also for all its sub-graphs. This method helps us to have fair deadline distribution. In the second step, in order to meet deadline as much as possible, critical tasks should be scheduled on faster servers. So, it is a good idea to have task priority in mapping step. Therefore, some models of priority are used in the mapping step and then dynamic programming technique is exploited to have an efficient sequential task scheduling.

After describing the totality of the proposed algorithm, details of the algorithm are provided in the following sections. Some notations used in the algorithm are introduced in Section 4.1. Deadline distribution over all the tasks is described in Section 4.2. In Section 4.3, an illustrative example of the proposed deadline distribution algorithm is provided. After deadline distribution phase, partitioning technique is explained in Section 4.4. This technique is used to have better mapping for pipeline branches. Finally, in Section 4.5, the mapping algorithm based on priority method is explained.

4.1 Basic Definitions

Minimum Execution Time (MET) for task t_i refers to the minimum time for an execution of task t_i on the fastest available server which is capable of processing task t_i. Equation (3) shows MET calculation.

$$\mathrm{MET}(t_i) = \min_{s \in S} ET(t_i, s) \tag{3}$$

where S and $ET(t_i, s)$ denote the set of servers in the system and the execution time of task t_i on server s, respectively. In fact, $\min_{s\in S} ET(t_i, s)$ is the execution time of task t_i on the fastest available resource for executing t_i.

Minimum Transfer Time (MTT) for arc e_{ij} denoted by MTT(e_{ij}) refers to the minimum time for data transmission of e_{ij} on the maximum available bandwidth of the grid. The calculations of MTT(e_{ij}) is shown in Equation (4).

$$\mathrm{MTT}(e_{ij}) = \min_{\forall S(t_i), S(t_j) \in S} \mathrm{Delay}(t_i, t_j). \tag{4}$$

Earliest Start Time (EST) for each task refers to the earliest possible time that the task can start its execution. In other words, EST shows earliest possible start time of task t_i if all predecessors of t_i are executed on the fastest server(s). Similarly, Latest Finish Time (LFT) of a task refers to the latest possible finish time of the task while executing all successors of that task on the fastest server does not cause absolute deadline violation. Therefore, if execution of task t_i terminates at the time LFT + ϵ and then all other tasks are executed on the fastest server, the execution of entire application will be finished on the time $\delta + \epsilon$. We can compute EST and LFT as Equation (5) and Equation (6), respectively.

$$\begin{aligned} \mathrm{EST}(t_{entry}) &= 0, \\ \mathrm{EST}(t_i) &= \max_{t_p \in imm\_preds(t_i)} \big(\mathrm{EST}(t_p) + \mathrm{MET}(t_p) + \mathrm{MTT}(e_{pi})\big), \end{aligned} \tag{5}$$

$$\begin{aligned} \mathrm{LFT}(t_{exit}) &= \delta, \\ \mathrm{LFT}(t_i) &= \min_{t_c \in imm\_succs(t_i)} \big(\mathrm{LFT}(t_c) - \mathrm{MET}(t_c) - \mathrm{MTT}(e_{ic})\big). \end{aligned} \tag{6}$$

Assume all predecessors and successors of task t_i are planned to be executed on the fastest server(s), so we have a wide range of time to schedule t_i. In other words, we are free to schedule task t_i in each part of this time period. Subtracting minimum execution time of task t_i from this time period, slack time of t_i shown by $\mathrm{ST}(t_i)$, is obtained. In fact, $\mathrm{ST}(t_i)$ is the maximum possible extra time for executing task t_i minus its minimum execution time. Equation (7) shows the slack time of task t_i.

$$\mathrm{ST}(t_i) = \mathrm{LFT}(t_i) - \mathrm{EST}(t_i) - \mathrm{MET}(t_i). \tag{7}$$

4.2 Deadline Distribution

Assigning a sub-deadline to each task according to the overall deadline of an application is the main objective of this step. In other words, we wish to assign a time window to each task for execution. This time window is determined by Schedule Start Time (SST) and Schedule Finish Time (SFT). During the deadline assignment, we are dealing with two types of tasks: *assigned* and *unassigned* tasks. If we assign a sub-deadline to a task (specifying both the start and finish times for the task), it is flagged as an assigned task, otherwise, it is called unassigned task. Dedicating the time interval of SST and SFT to the tasks does not mean that the tasks should be executed in these intervals; but these intervals give us an offline approximation of execution time of tasks based on other tasks within the workflow. The final goal of deadline distribution phase is assigning fair SST and SFT to each task. For this purpose, first, we have to compute earliest start time (EST) and latest finish time (LFT) of each task. Note that, for the assigned tasks, we do not compute EST and LFT measures, because they are equal to SST and SFT, respectively. Deadline assignment procedure is represented in Algorithm 1.

This algorithm begins with computing MET and MTT for all tasks, and then iteratively updates EST, LFT, and ST for each task and chooses a chain of unassigned tasks having minimum slack time. Considering this procedure, we certainly have a sequence of consecutive unassigned tasks that have minimum slack time. The first element of this sequence has assigned parent tasks and the last element has assigned child tasks. We call this sequence as *critical chain*. In other words, a *critical chain* is a sequence of unassigned tasks in which each task t_i is the immediate successor of task t_{i-1} in the chain. Moreover, all tasks in the *critical chain* have the same slack time which is the minimum one amongst slack times of all unassigned tasks within the application. If there is more than one *critical chain* with the aforementioned characteristic, one of them is arbitrarily chosen.

After selecting a *critical chain*, a time window can be assigned to each task of the chain. Let CST (Chain Start Time) denote a maximum LFT of immediate predecessors of the first task in the chain, and similarly, CFT (Chain Finish Time) denote a LFT of the last task in the chain. Now, we have a chain interval time CFT–CST which should be distributed over chain's tasks. We have already distributed this time interval to all tasks of the chain considering MET of each task. Actually, SST of the first task in the chain is its CST. Therefore, to obtain SFT

Algorithm 1: Deadline Distribution

```
1 Input: Application
  /* initialization                                                        */
  forall $t_i \in Application$ do
      Compute MET($t_i$);
      Compute MTT($t_i$);
  end
  while (there are unassigned tasks) do
      Update EST and LFT for all unassigned tasks;
      minSlack $\longleftarrow \infty$;
      chain $\leftarrow$ null;
      /* Check for critical sequence of task to schedule                   */
      for $i = 0$ to $i = numberOfTasks - 1$ do
          if $IsAssigned(t_i)$ then
              continue;
          end
          ST($t_i$) $\longleftarrow$ LFT($t_i$) – EST($t_i$) – MET($t_i$) ;
          if $ST(t_i) < minSlack$ then
              sequence $\longleftarrow$ null ;
              add $t_i$ to the end of sequence ;
              minSlack $\longleftarrow$ ST($t_i$);
          else if $TF(t_i) = minSlack$ then
              if sequence.lastItem is imm_pred of $t_i$ then
                  add $t_i$ to the end of sequence
              end
          end
      end
      /* distribute deadline over some partitions                          */
      CST $\longleftarrow \max_{t \in imm\_Preds(\text{Sequence.firstItem})}$ LFT($t$);
      CFT $\longleftarrow$ LFT(sequence.lastItem) ;
      Distribute deadline (CFT – CST) over all partitions in sequence
        proportional to MET($t_i$) ;
  end
```

(sub-deadline) of a task, it is sufficient to only add time interval assigned to the task to its SST. SST of the next task in the chain is also equal to SFT of the previous task in that chain. It turns out that SFT of the last task in the chain is equal to CFT. We mark all the tasks in the chain as assigned tasks and update unassigned tasks' EST and LFT measures and then continue with selecting a new *critical chain*.

This procedure goes on until all tasks are assigned. It should be mentioned that for each assigned task, EST and LFT are considered as SST and SFT, respectively. By assigning time window to the last task, sub-deadline assignment procedure is completed.

Finding computational complexity of the proposed deadline distribution algorithm is beyond the scope of this paper, but intuitively, it can be seen that in the worst-case, the algorithm assigns sub deadline to at least one task in each iteration. Moreover, the complexity of each iteration for updating SST and SFT is $O(n^2)$, where n is the number of tasks in the workflow. Therefore, the computational complexity of the algorithm is $O(n^3)$, but the upper bound can be more tight than stated. This bound is pessimistic and the practical complexity for real applications is much less than this value. From another point of view, we can consider that the complexity of updating SST, SFT and ST is $O(e)$, where e is the number of dependency relations between the tasks, or the number of edges in DAG representing of the workflow. Therefore, the computational complexity of the algorithm is $O(ne)$. Since e can take $(n)(n+1)/2$ in worst case, the computational complexity of our proposed algorithm is $O(n^3)$.

4.3 An Illustrative Example

Figure 2 is an example of *critical chain* deadline distribution for workflow application shown in Figure 1 while a sample grid environment with the specification presented in Table 1 is considered. Each processing node in the sample grid has its own specific processing power indicating million instructions that the node can process per second (MIPS). Obviously, different processing nodes have different usage prices based on their processing speeds. In this sample, bandwidths of all links among the servers are assumed to be 512 Mbps. In Figure 2 A), MET and MTT measures are shown for each task. As an example, task t_2 has 426 000 MI with computation requirement type of B. The fastest server which can execute this task is server s_2. Therefore, MET(t_2) is 426 000 MI/2 000 MIPS = 213 S. Similarly, we can compute MTT and MET measures for all tasks.

We can compute sub-deadlines for this example in three iterations shown in Figure 2 B) to Figure 2 D). The measures EST, LST and ST are computed for all tasks, and *critical chain* is obtained (e.g., $\{t_0, t_2, t_4, t_5\}$). This step is shown in Figure 2 B). In the first iteration, chain interval is equal to whole deadline (1250). Since, task t_2 needs 225 seconds to be completed (MET + MTT = 213 s + 12 s = 225 s), and task t_4 needs 125 seconds, the time assigned to task t_2 is 225/125 = 9/5 times greater than the time assigned to task t_4. Similarly, the time interval will be distributed over all members of the *critical chain.* Iteratively, we update EST, LST and ST measures to obtain another *critical chain* (chain $\{t_1\}$ in Figure 2 C)). Since, there exists only one task in *critical chain* shown in Figure 2 C), the whole time interval is assigned to task t_1, and SST and SFT measures of task t_1 are set to 50 and 500, respectively (Figure 2 D)). Finally, SST and SFT measures related to task t_3 are forced to be 500 and 750, respectively.

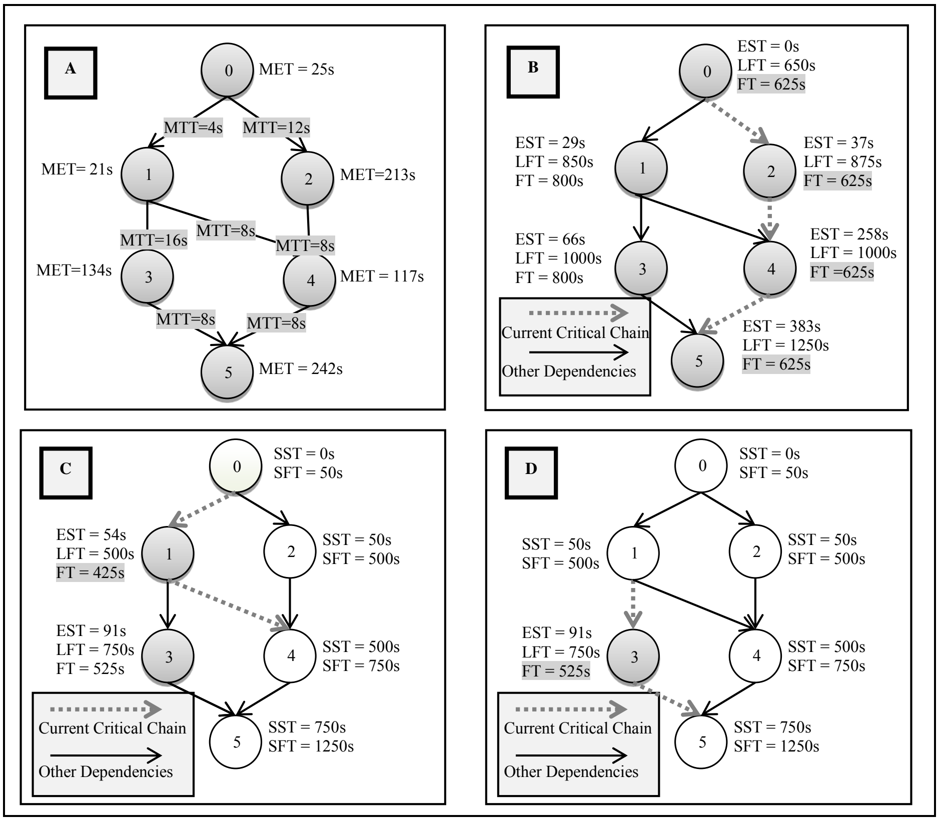


Figure 2. Deadline distribution over a sample workflow

4.4 Partitioning Technique

Partitioning is a method to optimally solve branches with several sequential tasks in grid workflows [9]. In partitioning model, tasks are divided into two different categories: simple tasks and synchronization tasks. If a task has at most one immediate predecessor, and at most one immediate successor, then the task is named simple task, otherwise, it is called synchronization task. We use the word branch

Server	Server Type	Processing Power (MIPS)	Price per Second ($)
s_1	A, B, D	1 000	0.001
s_2	B, C, E	2 000	0.004
s_3	A, C, D, E	4 000	0.016

Table 1. Sample grid specification with three processing nodes

to refer to several sequential simple tasks (a sub-graph like a pipeline workflow). The mapping problem of branches can be solved using dynamic programming and hidden Markov model. In Markov model, each of the states can be considered as a triple like [*termination time, number of scheduled tasks in a branch, last scheduled server*]. Afterwards, one can calculate value of each state with the help of previous states using dynamic programming. In this problem, the value of each Markov state is the cost of scheduling. The detailed description of partitioning technique can be found in [9]. Moreover, it should be mentioned that partitioning does not take part in deadline distribution phase of the proposed algorithm. This technique is used besides mapping phase. First, we partition the workflow in the branches, and then, in the mapping phase, we allocate a resource to all tasks in the branch with dynamic programming if we encounter a branch.

It remains a subtle point that dynamic programming is very time consuming, and calculation time extremely increases with increasing the problem size. Size of this problem depends on three different factors: the number of branch tasks, number of servers, and termination time range. The number of branch tasks and servers are constant values and if those numbers get higher, dynamic programming becomes inefficient. In almost all cases in our problem, we encounter few task branches with limited number of servers which can process each task. Therefore, dynamic programming can appropriately tolerate those factors in our case. As mentioned earlier, the third factor is termination time. If the time interval is wide, then we can segment the time to larger time pieces. Actually, we can handle larger problems in exchange for a bit of accuracy using segmentation.

4.5 Mapping Based on Priority

In the mapping phase, we try to map servers to the tasks to optimize the overall cost besides meeting the overall deadline. To do this, it is tried to have a local optimization in server allocation process to hopefully reach the global optimization. In the mapping phase, we iteratively pass over three steps. Mapping algorithm firstly selects a ready task (partition), a task that all of its immediate predecessors have been executed. Secondly, it updates the start time of unscheduled tasks to the minimum finish time of immediate predecessor plus delay of incoming link of predecessor. When there are one or more processors for processing a task before expiring its deadline, we choose the cheapest processor among those. If there is no server available to fulfill task's deadline, a server with minimum deadline violation is selected. Since both the number and power of the servers existing in the system are limited, it is important to have a priority algorithm to select a ready task to be assigned to the limited servers. The following three different methods for priority-based mapping are used in this paper:

Simple priority: A ready task with lower task ID has higher priority for scheduling in this method. Using this mechanism, we can simulate random priority.

Start time priority: A ready task with minimum start time has higher priority during the mapping phase. The main idea of start time priority method is that if a task has a lower start time, then there are probably more tasks waiting for that task, so it should be scheduled as soon as possible. The other idea behind start time priority is that, as soon as there is a ready task available for scheduling, it may be better to schedule it and do not wait for other tasks to become ready.

Slack time priority: For each ready task, a measure named *slack time* is computed using SST and SFT measures. Equation (8) computes slack time for each task.

$$\mathrm{ST}(t_i) = \mathrm{SFT}(t_i) - \mathrm{SST}(t_i) - \mathrm{MET}(t_i). \tag{8}$$

In slack time priority, the ready task with minimum slack time is prior to be scheduled. The main idea of slack time priority method is that if a task has lower slack time, then scheduling this task is more likely to be critical, and postponing its scheduling may lead to local sub-deadline violation and, consequently, possible global deadline violation.

According to the results presented in Section 5, slack time priority shows better performance in comparison with two other approaches. Finally, for the sake of brevity, we only present the performance analysis of the slack time priority.

5 PERFORMANCE EVALUATION

Accurate performance evaluation not only depends on the perfect implementation of the proposed and benchmark algorithms, but also it highly relies on the test data and experimental setup. In this section, it is described how workflow test sets are generated, and what are the main experimental setups. After that, the results obtained from the experiments are presented.

5.1 Generating Workflows

Three types of common workflow structures, pipeline, parallel and hybrid workflows [9, 10, 11, 13, 14, 19, 37], are considered in this paper. These structures are shown in Figure 3. Pipeline workflow consists of numbers of tasks in a sequential order (Figure 3 A)). Parallel workflows include multiple pipelines with some middle synchronizer tasks (Figure 3 B)). Hybrid workflows are combinations of pipeline and parallel workflows in an arbitrary order (Figure 3 C)). Structure of pipeline workflows is simple, but many factors influence on construction of parallel and hybrid workflows. Most important factors in parallel workflow construction are the maximum width of graph (the number of parallel pipeline chains), and maximum number of possible sub-pipeline tasks. We set the maximum width and the maximum pipeline length to 10 and 20, respectively. There is no constraint on choosing other numbers for width and maximum pipeline length. We just choose those numbers according

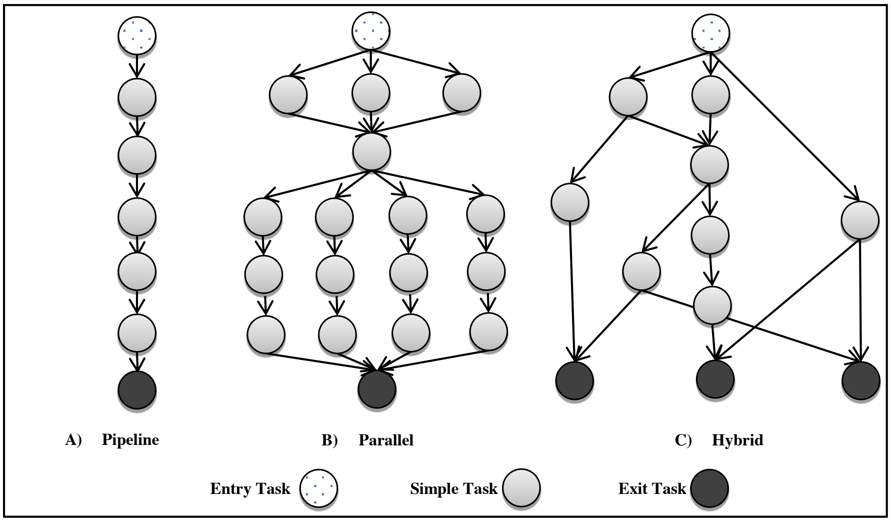


Figure 3. Different types of random workflow structures

to previously done similar work [10]. Generating hybrid workflow structures is more complex. There are some papers addressing this problem [37, 38, 39]. They study parametric features of each generation method such as average critical path length and average edge. In [39], a tool for random graph generation, named TGFF, which is approximately fair in graph generation was presented. According to study done by Cordeiro et al. [37], the Max-in/Max-out degree method is one of the best methods for workflow generation. Hence, TGFF is used in this paper to produce hybrid workflows with Max-in/Max-out method. For TGFF method, the Max-in (Max-out) parameters are considered from 1 to 3 ($Max_{in}(Max_{out}) \in \{1, 2, 3\}$), and the number of tasks varies from 30 to 1000. Figure 3 C) shows a sample output of TGFF with specified parameters. In addition to random workflows, we compare the proposed method with other algorithms applying the standard workflows used in [13] called CyberShake, Montage, LIGO, Gnome, and SIPHT, which are real workflows in the scientific or business communities.

5.2 Experimental Setup

We consider details of real environments in our experiments such as grid structure, network bandwidth, processing power of each processing node, largeness of workflows, deadline determination and so forth. All assumptions made in this work are consistent with previously done research work [6, 9, 10, 13, 14]. In this section, we have a glance on implementation details.

Workflow Specification: The structures of workflows are chosen randomly as illustrated in Section 5.1. From the viewpoint of granularity, we consider three types of workflows: small, medium and large. The number of tasks vary from 30 to 200, from 200 to 600, and from 600 to 1 000 in small, medium and large workflows, respectively. For the heterogeneity of workflows, it is assumed that each task has specific processing requirements. We assume there are 15 different types of tasks. Moreover, the size of each task varies from 100 000 MI to 900 000 MI in this study. For each workflow, there are several exit tasks and only one entry task. Each task can be executed as soon as its corresponding resource receives the required data for processing that task. The input/output data for the task changes from 10 MB to 1 GB, as well.

Grid Specification: To simulate the server heterogeneity in grid network, we consider 15 different types of services, each service supported by 10 different grid servers. Network bandwidth between each two grid servers is considered to be in the range of 200 Mbps to 512 Mbps. Grid server processing power shown by MIPS (Million Instructions Per Second) varies from 1 000 MIPS to 5 000 MIPS for each node. For each processing node there is a price proportional to its power ranging from 0.001 \$ to 0.025 \$. This means that executing a task on a server with a processor n times faster than another, imposes n times more cost to the scheduling.

Hardware Specification: We run the simulation on a regular PC with Intel® Core$^{\text{TM}}$ i5-4200M (3M Cache, up to 3.10 GHz) and 4 GB RAM. Based on this hardware configuration, each test case runs in 1 to 2 seconds (except for huge pipeline workflows), and consequently, each test set containing more than 500 different test cases runs in less than 30 minutes. For a pipeline workflow test case with 200 tasks, it takes up to 5 minutes to run. It turns out that the time required to run the simulation reduces by using more powerful hardware.

Deadline Assignment: Each workflow is tested with 9 different deadlines, i.e., $\delta_n = T_{min} \times \theta$, where T_{min} is the minimum completion time for that workflow on the specified grid, and $\theta \in \{1, 1.05, 1.1, 1.15, 1.2, 1.5, 2, 2.5, 3\}$. Practically, there is no need to consider $\theta > 3$, because in that range, all tasks are delivered to lower and cheaper processing nodes, and approximately all methods show the same efficiency. Determining T_{min} is an NP-hard problem, so we use HEFT greedy algorithm [28] to estimate minimum completion time.

5.3 Result Analysis

We compare our method with four recent most-cited methods: PCP Fair, PCP Decrease, DTL, and DBL [9, 13, 29]. As mentioned earlier, we use three types of workflows to do experiments. Each of random workflows is tested by over 800 instances to achieve more reliable results. Each instance is tested by 5 different methods with 8 different deadlines, and the experiment is done with near 100 000 iterations. Three different priority-based mapping methods are tested, but for the sake of brevity, only one of the results is reported here.

		DBL	DTL Decrease	PCP Fair	PCP	Critical Chain
PipeLine	**Small**	0	0	0	0	100
	Medium	0	0	0	0	100
	Large	0	0	0	0	100
Parallel	**Small**	10.4	10.5	0.33	0.22	89
	Medium	11.88	11.78	2.78	1.56	83.69
	Large	17	17	0.22	13.44	69.33
Hybrid	**Small**	26.02	25.4	0.34	11.23	48.87
	Medium	20.09	22.10	2.91	22.27	31.58
	Large	29.89	24.66	0	16.31	36.62

Table 2. Best result percentage

In our study, an algorithm is superior to another if it appropriately satisfies the following three properties:

Best results: if results of two different schedulings violate the deadline, the result that has the minimum time is the best result for our case study. If none of them has deadline violation, the result with the minimum scheduling price is the best one. An algorithm with maximum percentage of best results is named best suitable algorithm in our study.

Deadline Violation Rate (DVR): if an algorithm has minimum deadline violation rate, then it is superior to the others.

Average cost: the algorithm with minimum average cost is preferred to the others.

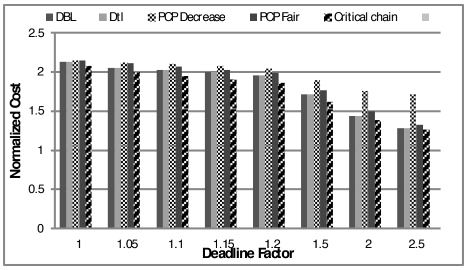
The best result percentages of all algorithms for all test sets are given in Table 2. As it can be seen in Table 2, from the aspect of the best results percentage, the proposed *critical chain* method is dominant. This means that for most of the applications, *critical chain* shows the best performance compared to the other benchmarks.

In pipeline workflows, *critical chain* uses dynamic programming for branches, and computes optimal solution; therefore, this method is dominant in this type of workflows. Since the algorithm produces best result in pipeline workflows, the percentage of the test cases on which any other method can produce best result is *zero*. This is shown in Table 2 by elements 0 inside cells related to the pipeline workflows. It should be mentioned that since the scheduling problem considered in this paper belongs to the set of NP-hard problems, and there is no polynomial solution to solve it, we tried to solve it by dynamic programming that is very time consuming. Therefore, the best solution with 100% confidence in pipeline workflows is achieved by imposing extra scheduling time on the scheduler. Dynamic programming imposes heavy computational overhead to the scheduler, so that, computing the best solution for over 200 sequential tasks in a reasonable time is impractical for schedulers. For the parallel workflows, our algorithm is dominant too with a slight decrease in performance in comparison with other algorithms. Parallel workloads contain many pipeline chains and our algorithm uses dynamic programming and fair deadline distribution to receive this performance. If we consider a very long length for a parallel workload, then we have to solve many long pipeline problems to schedule the long-length parallel workload, and as a result, we may encounter the performance problem described above. It is worthwhile to mention that scheduling parallel workloads has no polynomial optimal solution. Therefore, performance degradation in comparison with pipeline workflows is reasonable. In the hybrid workflows, the pipeline chains are in minority, so dynamic programming cannot be useful further, but our algorithm still shows relatively good performance. We believe that our deadline distribution method has an important role to achieve this. Based on the simulations done, we found that increasing the size of the graph does not considerably change the results. Hence, the proposed algorithm shows the same performance for different workflow sizes. In other words, the structure of a workflow has much greater impact on the results. We continue to analyze the algorithms in the view of deadline violation rate and average cost properties for each of the workflow structures. For the sake of brevity, we depict only two classes of diagrams for each workflow type. The left-side diagrams in Figure 4 show average cost resulted by algorithms. Since these diagrams show the average cost for all small, medium, and large applications, they should be normalized to match with each other in a single diagram. The Normalized Cost (NC) is computed as Equation (9).

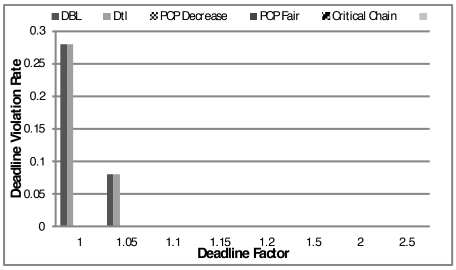
$$\mathrm{NC} = \frac{\text{Scheduling Cost}}{\text{Minimum Scheduling Cost}}. \tag{9}$$

After scheduling each workflow, the cost resulting from the scheduling is divided by the minimum scheduling cost obtained from a greedy algorithm and then NC is achieved. The minimum scheduling cost is obtained from greedy scheduler that assigns each task to the cheapest resource with unlimited deadline. The right-side diagrams in Figure 4 show the deadline violation rate of the algorithms. If a violation rate of one algorithm is α with β as a deadline factor, it means that α is the probability of deadline violation by applying that scheduling algorithm with $\beta \times T_{min}$ as a deadline. All algorithms try to schedule all tasks to faster servers whenever deadlines are firm (deadline = 1). This tends to increase the scheduling cost resulted from all methods. Moreover, the rate of deadline violation increases in this situation. Expanding the deadline associated with each task, and as a result, increasing the overall deadline of the workflow, deadline violation rate and execution cost resulted from all algorithms decrease.

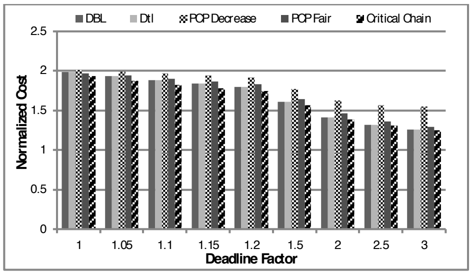
For the pipeline (Figure 4 A) and Figure 4 B)) and parallel (Figure 4 C) and Figure 4 D)) workflows, DVR and NC for *critical chain* get minimum values compared to the other algorithms. This shows that the best method for scheduling these types of workflows amongst all of the algorithms implemented in this paper is *critical chain.* For hybrid workflows (Figure 4 E) and Figure 4 F)) DVR in all deadlines and NC for soft deadlines are minima in *critical chain* method, but in the firm deadlines, NC for *critical chain* is slightly more than DBL algorithm. It is



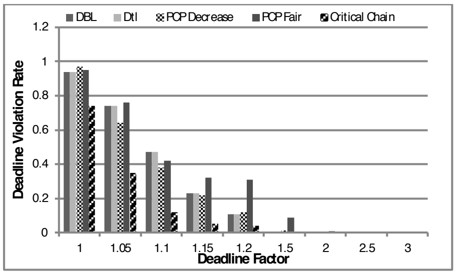
A- NC for Pipeline Workflows



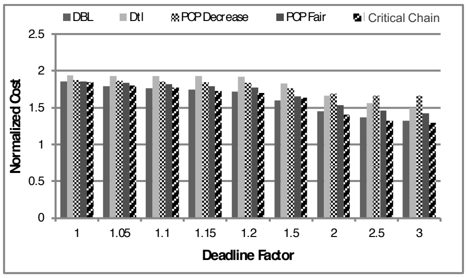
B- DVR for Pipeline Workflows



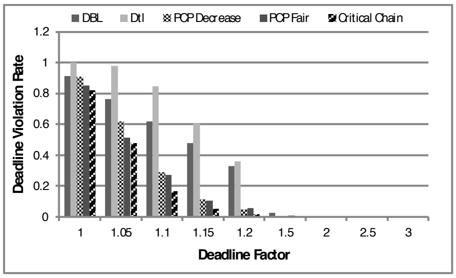
C- NC for Parallel Workflows



D- DVR for Parallel Workflows



E- NC for Hybrid Workflows

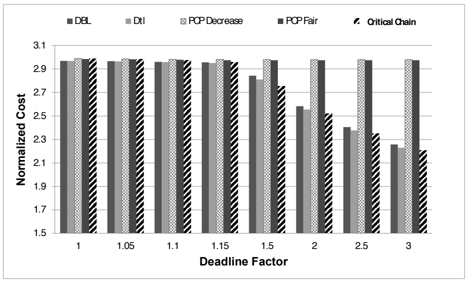


F- DVR for Hybrid Workflows

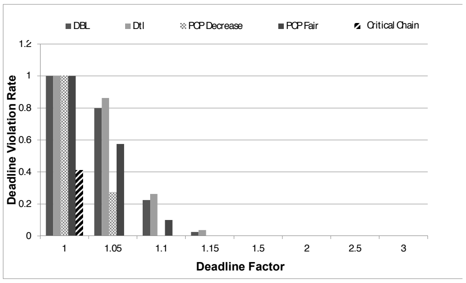
Figure 4. Comparison of Normalized Cost (NC) and Deadline Violation Rate (DVR) resulted from all algorithms for random workflows

predictable, because in a firm deadline, *critical chain* tries to not exceed deadline by scheduling tasks on the faster (consequently more expensive) resources. Finally, it can be stated that DVR and average NC for *critical chain* are minima in comparison with other algorithms. Hence, the proposed *critical chain* method outperforms previous methods from the viewpoint of best results, deadline violation rate, and average cost.

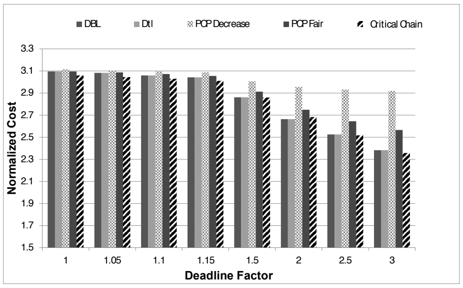
As mentioned in Section 5.1, we apply our proposed algorithm on realistic standard workflows presented in [13]. The results obtained from applying the proposed algorithm on medium size workflows including CyberShake, Montage, and LIGO are presented in Figure 5, and similarly the results gained from workflows with large and small sizes, Gnome and SIPHT, are presented in Figure 6. As can be seen in



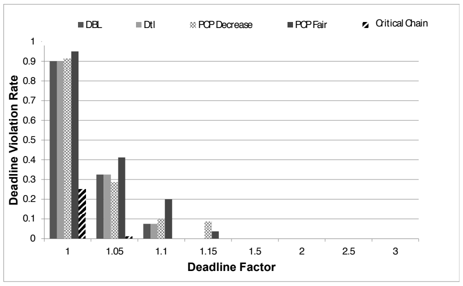
A – NC for CyberShake Workflow



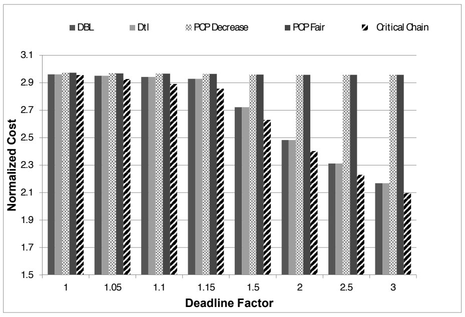
B – DVR for CyberShake Workflow



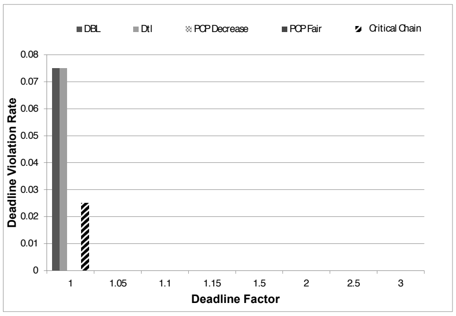
C – NC for Montage Workflow



D – DVR for Montage Workflow



E – NC for LIGO Workflow



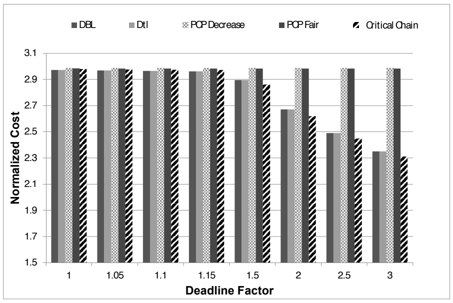
F – DVR for LIGO Workflow

Figure 5. Comparison of Normalized Cost (NC) and Deadline Violation Rate (DVR) resulted from all algorithms for standard benchmark workflows with medium sizes

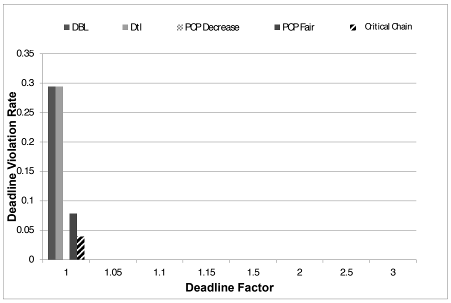
both Figure 5 and Figure 6, our algorithm dominates other algorithms in average by minimizing DVR and reducing the scheduling cost.

6 CONCLUSIONS AND FUTURE WORK

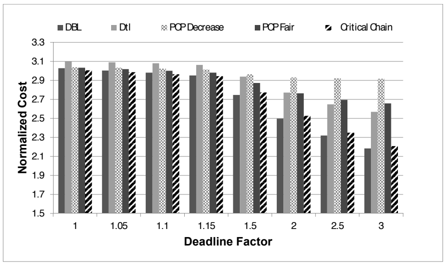
The basic principle used in utility computing and grid computing is identical which is providing computational resources as a service. One of the most important problems in such environments is scheduling deadline constrained bag-of-tasks applications on computational resources. In this paper, we study the deadline of applications as one



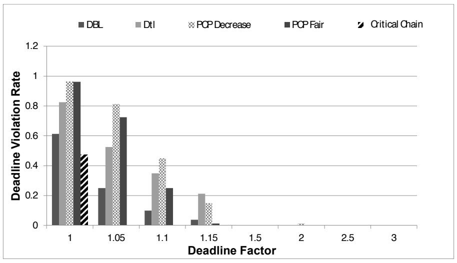
A – NC for Gnome Workflow (Large size)



B – DVR for Gnome Workflow (Large size)



C – NC for SIPHT Workflow (Small size)



D – DVR for SIPHT Workflow (Small size)

Figure 6. Comparison of Normalized Cost (NC) and Deadline Violation Rate (DVR) resulted from all algorithms for standard benchmark workflows with large and small sizes

of the more interesting factors in grid computing, and try to deliver a service with specified QoS and minimum cost. Therefore, our problem can be considered as a time-cost trade-off problem. To solve this optimization problem, two-step *critical chain* heuristic is presented. In deadline distribution phase, the algorithm applies a fairer mechanism to better deadline distribution, which finally, leads to lower cost of service. In the second step named resource allocation phase, resources are allocated to the tasks efficiently according to the priority of tasks. Finally, applying the proposed method to different scenarios and system settings, it is shown that the proposed approach outperforms other similar existing methods.

Critical chain is applicable in other distributed computing domains such as clouds. In cloud systems, customers can select their desired service based on their budget and time constrains, and pay for using these services. Therefore, the problem of scheduling workflow applications on limited resources considering time and budget constraints is an interesting problem in clouds. So, one possible extension to this work is applying the proposed critical chain method on scheduling workflows in cloud infrastructures considering their specific characteristics and requirements.

Critical path-based deadline distribution has deficiency in bursting tasks. In other words, if the workflow width becomes unpredictable during the specified time period, the deadline distribution based on critical path with any available method would be inefficient. So, one can modify the proposed method to handle this type of workflows. Furthermore, considering the map-reduce structure as a new type of workflows and modifying the proposed method to have high performance in map-reduce workflows can be assumed as another objective. Evaluating other performance measures such as resource utilization and taking server setup cost into account can be considered as an important direction for improvements of the current work.

REFERENCES

[1] Foster, I.—Kesselman, C.: The Grid 2: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco, CA, USA, 2004.

[2] Czajkowski, K.—Fitzgerald, S.—Foster, I.—Kesselman, C.: Grid Information Services for Distributed Resource Sharing. Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing, San Francisco, CA, USA, August 2001, pp. 181–194, doi: 10.1109/HPDC.2001.945188.

[3] Broberg, J.—Venugopal, S.—Buyya, R.: Market-Oriented Grids and Utility Computing: The State-of-the-Art and Future Directions. Journal of Grid Computing, Vol. 6, 2008, No. 3, pp. 255–276, doi: 10.1007/s10723-007-9095-3.

[4] Toporkov, V.—Yemelyanov, D.—Potekhin, P.—Toporkova, A.—Tselishchev, A.: Metascheduling and Heuristic Co-Allocation Strategies in Distributed Computing. Computing and Informatics, Vol. 34, 2015, No. 1, pp. 45–76.

[5] Yu, J.—Buyya, R.—Ramamohanarao, K.: Workflow Scheduling Algorithms for Grid Computing. In: Xhafa, F., Abraham, A. (Eds.): Metaheuristics for Scheduling in Distributed Computing Environments. Springer, Berlin, Heidelberg, Studies in Computational Intelligence, Vol. 146, 2008, pp. 173–214.

[6] Yu, J.—Buyya, R.: A Budget Constrained Scheduling of Workflow Applications on Utility Grids Using Genetic Algorithms. The Workshop on Workflows in Support of Large-Scale Science, Paris, France, June 2006, pp. 1–10, doi: 10.1109/WORKS.2006.5282330.

[7] Dong, F.—Akl, S. G.: Scheduling Algorithms for Grid Computing: State of the Art and Open Problems. Technical Report No. 2006-504: School of Computing, Queen's University, Kingston, Ontario, 2006, pp. 1–55.

[8] Atanak, M. M.—Dogan, A.: Improving Real-Time Data Dissemination Performance by Multi Path Data Scheduling in Data Grids. Computing and Informatics, Vol. 34, 2015, No. 2, pp. 402–424.

[9] Yu, J.—Buyya, R.—Tham, C. K.: Cost-Based Scheduling of Scientific Workflow Applications on Utility Grids. Proceedings of the 1st International Conference on e-Science and Grid Computing, Melbourne, Australia, December 2005, pp. 140–147.

[10] Yuan, Y.—Li, X.—Wang, Q.—Zhu, X.: Deadline Division-Based Heuristic for Cost Optimization in Workflow Scheduling. Information Sciences, Vol. 179, 2009, No. 15, pp. 2562–2575.

[11] Rahman, M.—Venugopal, S.—Buyya, R.: A Dynamic Critical Path Algorithm for Scheduling Scientific Workflow Applications on Global Grids. Proceed-

ings of the 3rd IEEE International Conference on e-Science and Grid Computing (e-Science 2007), Bangalore, India, December 2007, pp. 35–42, doi: 10.1109/E-SCIENCE.2007.3.

[12] Alkhanak, E. N.—Lee, S. P.—Khan, S. U. R.: Cost-Aware Challenges for Workflow Scheduling Approaches in Cloud Computing Environments: Taxonomy and Opportunities. Future Generation Computer Systems, Vol. 50, 2015, pp. 3–21, doi: 10.1016/j.future.2015.01.007.

[13] Abrishami, S.—Naghibzadeh, M.—Epema, D. H. J.: Cost-Driven Scheduling of Grid Workflows Using Partial Critical Paths. Proceedings of the 11th IEEE/ACM International Conference on Grid Computing, Brussels, Belgium, October 2010, pp. 81–88, doi: 10.1109/GRID.2010.5697955.

[14] Yao, Y.—Liu, J.—Ma, L.: Efficient Cost Optimization for Workflow Scheduling on Grids. Proceedings of the International Conference on Management and Service Science, Wuhan, China, August 2010, pp. 1–4, doi: 10.1109/ICMSS.2010.5577645.

[15] Henzinger, T. A.—Singh, A. V.—Singh, V.—Wies, T.—Zufferey, D.: Flex-PRICE: Flexible Provisioning of Resources in a Cloud Environment. Proceedings of the 3rd IEEE International Conference on Cloud Computing, Miami, USA, July 2010, pp. 83–90, doi: 10.1109/CLOUD.2010.71.

[16] Arabnejad, H.—Barbosa, J. G.—Prodan, R.: Low-Time Complexity Budget–Deadline Constrained Workflow Scheduling on Heterogeneous Resources. Future Generation Computer Systems, Vol. 55, 2016, pp. 29–40, doi: 10.1016/j.future.2015.07.021.

[17] Rahman, M.—Hassan, R.—Ranjan, R.—Buyya, R.: Adaptive Workflow Scheduling for Dynamic Grid and Cloud Computing Environment. Concurrency and Computation: Practice and Experience, Vol. 25, 2013, No. 13, pp. 1816–1842.

[18] Kwok, Y. K.—Ahmad, I.: Dynamic Critical-Path Scheduling: An Effective Technique for Allocating Task Graphs to Multiprocessors. IEEE Transactions on Parallel and Distributed Systems, Vol. 7, 1996, No. 5, pp. 506–521.

[19] Lin, M.—Lin, Z.: A Cost-Effective Critical Path Approach for Service Priority Selections in Grid Computing Economy. Decision Support Systems, Vol. 42, 2006, No. 3, pp. 1628–1640.

[20] Tabbaa, N.—Entezari-Maleki, R.—Movaghar, A.: Reduced Communications Fault Tolerant Task Scheduling Algorithm for Multiprocessor Systems. Procedia Engineering, Vol. 29, 2012, pp. 3820–3825, doi: 10.1016/j.proeng.2012.01.577.

[21] Yang, T.—Gerasoulis, A.: DSC: Scheduling Parallel Tasks on an Unbounded Number of Processors. IEEE Transactions on Parallel and Distributed Systems, Vol. 5, 1994, No. 9, pp. 951–967.

[22] Foster, I.—Fidler, M.—Roy, A. J.—Sander, V.—Winkler, L.: End-to-End Quality of Service for High-End Applications. Computer Communications, Vol. 27, 2004, No. 14, pp. 1375–1388, doi: 10.1016/j.comcom.2004.02.014.

[23] Dogan, A.—Ozguner, F.: Scheduling Independent Tasks with QoS Requirements in Grid Computing with Time-Varying Resource Prices. In: Parashar, M. (Ed.): Grid Computing – GRID 2002. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2536, 2002, pp. 58–69.

[24] Tabbaa, N.—Entezari-Maleki, R.—Movaghar, A.: A Fault Tolerant Scheduling Algorithm for DAG Applications in Cluster Environments. In: Snasel, V., Platos, J., El-Qawasmeh, E. (Eds.): Digital Information Processing and Communications (ICDIPC 2011). Springer, Berlin, Heidelberg, Communications in Computer and Information Science, Vol. 188, 2011, pp. 189–199.

[25] Entezari-Maleki, R.—Movaghar, A.: A Genetic-Based Scheduling Algorithm to Minimize the Makespan of the Grid Applications. In: Kim, T. H., Yau, S. S., Gervasi, O., Kang, B. H., Stoica, A., Ślęzak, D. (Eds.): Grid and Distributed Computing, Control and Automation (GDC 2010, CA 2010). Springer, Berlin, Heidelberg, Communications in Computer and Information Science, Vol. 121, 2010, pp. 22–31.

[26] Kardani-Moghaddam, S.—Khodadadi, F.—Entezari-Maleki, R.—Movaghar, A.: A Hybrid Genetic Algorithm and Variable Neighborhood Search for Task Scheduling Problem in Grid Environment. Procedia Engineering, Vol. 29, 2012, pp. 3808–3814, doi: 10.1016/j.proeng.2012.01.575.

[27] Agrawal, K.—Benoit, A.—Magnan, L.—Robert, Y.: Scheduling Algorithms for Linear Workflow Optimization. Proceedings of the IEEE International Symposium on Parallel and Distributed Processing (IPDPS), Atlanta, GA, April 2010, pp. 1–12, doi: 10.1109/IPDPS.2010.5470346.

[28] Zhao, H.—Sakellariou, R.: An Experimental Investigation into the Rank Function of the Heterogeneous Earliest Finish Time Scheduling Algorithm. In: Kosch, H., Böszörményi, L., Hellwagner, H. (Eds.): Euro-Par 2003 Parallel Processing. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2790, 2003, pp. 189–194.

[29] Yuan, Y.-C.—Li, X.—Wang, Q.—Zhang, Y.: Bottom Level Based Heuristic for Workflow Scheduling in Grids. Chinese Journal of Computers, Vol. 31, 2008, No. 2, pp. 280–290.

[30] Poola, D.—Garg, S. K.—Buyya, R.—Yang, Y.—Ramamohanarao, K.: Robust Scheduling of Scientific Workflows with Deadline and Budget Constraints in Clouds. Proceedings of the IEEE 28th International Conference on Advanced Information Networking and Applications, Victoria, Canada, May 2014, pp. 858–865, doi: 10.1109/AINA.2014.105.

[31] Calheiros, R. N.—Buyya, R.: Cost-Effective Provisioning and Scheduling of Deadline-Constrained Applications in Hybrid Clouds. In: Wang, X. S., Cruz, I., Delis, A., Huang, G. (Eds.): Web Information Systems Engineering (WISE 2012). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7651, 2012, pp. 171–184.

[32] Qiu, X.—Yeow, W. L.—Wu, C.—Lau, F. C. M.: Cost-Minimizing Preemptive Scheduling of Mapreduce Workloads on Hybrid Clouds. Proceedings of the IEEE/ACM 21st International Symposium on Quality of Service (IWQoS), Montreal, Canada, June 2013, pp. 1–6.

[33] Van den Bossche, R.—Vanmechelen, K.—Broeckhove, J.: Cost-Optimal Scheduling in Hybrid IaaS Clouds for Deadline Constrained Workloads. Proceedings of the IEEE 3rd International Conference on Cloud Computing, Miami, FL, USA, July 2010, pp. 228–235, doi: 10.1109/CLOUD.2010.58.

[34] Wang, M.—Zhu, L.—Zhang, Z.: Risk-Aware Intermediate Dataset Backup Strategy in Cloud-Based Data Intensive Workflows. Future Generation Computer Systems, Vol. 55, 2016, pp. 524–533, doi: 10.1016/j.future.2014.08.009.

[35] Fard, H. M.—Prodan, R.—Fahringer, T.: A Truthful Dynamic Workflow Scheduling Mechanism for Commercial Multicloud Environments. IEEE Transactions on Parallel and Distributed Systems, Vol. 24, 2013, No. 6, pp. 1203–1212.

[36] De, P.—Dunne, E. J.—Ghosh, J. B.—Wells, C. E.: The Discrete Time-Cost Tradeoff Problem Revisited. European Journal of Operational Research, Vol. 81, 1995, No. 2, pp. 225–238.

[37] Cordeiro, D.—Mounié, G.—Perarnau, S.—Trystram, D.—Vincent, J.-M.—Wagner, F.: Random Graph Generation for Scheduling Simulations. Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques (SIMUTools '10), Torremolinos, Spain, March 2010, pp. 60:1–60:10, doi: 10.4108/ICST.SIMUTOOLS2010.8667.

[38] Li, H.: Workload Characterization, Modeling, and Prediction in Grid Computing. Doctoral Thesis. LIACS, Computer Systems Group, Faculty of Science, Leiden University, 2008.

[39] Dick, R. P.—Rhodes, D. L.—Wolf, W.: TGFF: Task Graphs for Free. Proceedings of the 6th International Workshop on Hardware/Software Codesign (CODES/CASHE '98), Seattle, USA, March 1998, pp. 97–101.

Alireza Dehlaghi-Ghadim is currently Ph.D. candidate in the Department of Electrical and Computer Engineering, University of Tehran, Tehran, Iran. He received his M.Sc. degree from Sharif University of Technology, Tehran, Iran, in 2012, and his B.Sc. degree from Iran University of Science and Technology, Tehran, Iran in 2009. His main research interests are grid and cloud computing, task scheduling algorithms, load balancing and resource allocation methods.

Reza Entezari-Maleki is Post-Doctoral Researcher in the School of Computer Science at Institute for Research in Fundamental Sciences (IPM) in Tehran, Iran. He received his Ph.D. in computer engineering (software discipline) from Sharif University of Technology, Tehran, Iran in 2014, and M.Sc. and B.Sc. degrees in computer engineering (software discipline) from Iran University of Science and Technology, Tehran, Iran in 2009 and 2007, respectively. He visited the Seoul National University in Seoul, South Korea, Duke University in NC, USA, and Instituto Superior Técnico in Lisbon, Portugal in 2012, 2013, and 2015, respectively, His main research interests are performance/dependability modeling and evaluation, grid and cloud computing, and task scheduling algorithms.

Ali Movaghar is Professor in the Department of Computer Engineering at Sharif University of Technology in Tehran, Iran and has been on the Sharif faculty since 1993. He received his B.Sc. degree in electrical engineering from the University of Tehran in 1977, and M.Sc. and Ph.D. degrees in computer, information, and control engineering from the University of Michigan, Ann Arbor, in 1979 and 1985, respectively. He visited the Institut National de Recherche en Informatique et en Automatique in Paris, France and the Department of Electrical Engineering and Computer Science at the University of California, Irvine in 1984 and 2011, respectively, worked at AT & T Information Systems in Naperville, IL in 1985–1986, and he was teaching at the University of Michigan, Ann Arbor in 1987–1989. His research interests include performance/dependability modeling and formal verification of wireless networks and distributed real-time systems. He is a senior member of the IEEE and the ACM.

Computing and Informatics, Vol. 37, 2018, 865–893, doi: 10.4149/cai_2018_4_865

BACKGROUND SUBTRACTION BASED ON PERCEPTION-CONTAINED PIECEWISE MEMORIZING FRAMEWORK

Songbo Liu

School of Computer Science, Harbin Institute of Technology
No. 92 Dazhi Street, Harbin, China
e-mail: sbliu@hit.edu.cn

Xudong Zhao

College of Information and Computer Engineering
Northeast Forestry University
No. 26 Hexing Road, Harbin, China

Xianglong Tang

School of Computer Science, Harbin Institute of Technology
No. 92 Dazhi Street, Harbin, China

Abstract. A key issue for full-time video surveillance is to search or establish a reference image of background which corresponds to current video frame. However, background that was ever in presence long time ago is enclosed or discarded due to background forgetting assumption. How to rapidly pick up or even rebuild long-term background needs to be discussed. This paper aims to present a framework for background maintenance in order to solve the problem. A piecewise memorizing framework is proposed for matching, updating and even rebuilding long-term background. Based on the metaphors of psychological selective attention theory, the framework is composed of a prior piecewise perception processor for intensity stationary test. Besides, a hierarchical memorizing mechanism constitutes the other part of the framework for overcoming the exponential forgetting of long period background appearances. Applied to Gaussian mixture model (GMM), this framework is capable of maintaining short-term background states, identifying long period background appearances, and rapidly adjusting to new background states according to different expressions derived from the prior perception of scene intensity changes. Its effectiveness can be demonstrated by experimental results for solving various typical problems.

Keywords: Long-term background memory, piecewise stationary test, Gaussian mixture model, background subtraction, foreground detection

Mathematics Subject Classification 2010: 68-T45

1 INTRODUCTION

Background subtraction, which compares each current frame with a dynamic reference image of background derived from previous frames, forms a favorite and powerful anterior mechanism for applications such as foreground detection [1], face recognition [2], object tracking [3], etc. Prevailing methods mainly exist in three aspects. One focuses on statistical features, for instance, Gaussian mixture model (GMM) [4], a kernel density estimation [5], Bayesian modeling [6], auto-regressive model [7], etc. Other integrates structural information with local binary pattern (LBP) [8], self-organizing network [9], codebook model [10], co-occurrence matrix [11], etc. The last concentrates on the statistical and structural hybrid accompanied with a three-stage model [12], spatio-temporal information saliency [13], a statistical multi-point-pair model [14], an efficient hierarchical method [15], etc.

Despite its importance, background subtraction is far from a complete solution to the disturbances of complex environments. The problems listed mainly in [12] can be re-categorized as follows: background varying illumination including *time of day*, *light switch* and *shadows*; background periodic motion (e.g. *waving trees*); background consistency containing *camouflage*, *bootstrapping* and *foreground aperture*; and semantic feedback (e.g. *moved objects*, *sleeping person* and *waking person*, namely *ghosts*) which is not able to be handled using the background subtraction module only, as indicated in [12]. Except for background consistency, these problems seem to be probably solved once we memorize a series of full-time background appearances with one of which the current frame is matched. However, it is hard for memorizing background that was ever in presence long time ago (namely, long-term or long period background) considering the restriction of memory capacity.

On account of the storage limitation, time constraint is always taken into consideration. A background forgetting assumption is made that states often appearing in high frequency are treated as background. A forgetting factor is employed to control the contribution of earlier observations. Thus, only background appearances corresponding to recent video frames are memorized. Inevitably, states representing long period background are discarded. The strategy works as to the presence of the stationary states such as *time of day*; while, it lapses when non-stationary states such as *light switch* appear. In order to solve this problem, a local principal component analysis transformation accompanied with an adaptive learning (ALPCA) [16] is proposed to learn separate feature eigensubspaces representing different background appearances. ALPCA is effective especially when *light switch* appears, for it uses spatial information to guide switches among short-term background appearances representing different illumination conditions. Meanwhile, parameters corresponding to the forgetting factor are rapidly updated. However, it will inevitably lapse into frequent switches among recent background appearances with different illumination conditions due to absence of long-term background memory.

Instead of memorizing long-term background, prevailing methods mainly focus on concrete problems mentioned above. Narrowing down to GMM as an effective background model, there have been various improvements contributed to address these problems. A texture-contained GMM (TGMM) [17] and a spatial hierarchical GMM (HGMM) [18] are utilized for solving the problem of *light switch*. *Shadows* are eliminated using a Gaussian mixture shadow model (GMSM) [19] and TGMM. As to the problem of *ghosts*, a memorizing GMM (MGMM) [20] is applied to tackling the exponential decay [21] of GMM. Although these algorithms are competent to handle certain problems, there is a lack of a common way to simultaneously solve all these problems because of neglecting long period background memory. Even if we can break through the restriction of memory capacity, it is still difficult for us to seek the matched states of background associated with current video frame in real time due to the large amount of full-time background storage.

In order to tackle the problem about long-term background, a piecewise memorizing framework is proposed. Three major contributions can be claimed. Taking the metaphors of psychological selective attention theory into consideration, hypotheses of background subtraction indicating what to perceive and memorize are firstly proposed. Second, a prior perception-concerned recognition of non-stationary intensity change is presented based on a segmented stationarity test. Third, a memorizing hierarchy corresponding to GMM is put forward for the storage of long period background and the adaptation to rapid intensity change. Particularly, the memorizing framework accompanied with a prior perception processor based on segmented stationarity test is an adaptive multi-scale hierarchical system, which is constituted by a conventional mixture of short-term background models, spatial states memorizing a long-term background, and a sequence of global differences adapting to fast intensity change of background. Unlike these referenced methods including our previous research [22], a prior piecewise perception processor for intensity stationary test is proposed to search or establish long-term background rapidly that corresponds to current video frame. Regarding GMM as the basic background memorizing model in this paper, this framework is named as P-MGMM (piecewise memorizing GMM). The rest of the paper is organized as follows: in Section 2, we revisit GMM, point out the limitations of related methods, discuss the corresponding metaphors of cognitive psychology, and make related assumptions; in Section 3, we present a prior segmented stationarity test, which is viewed as a pre-perception of background variation; in Section 4, a memorizing hierarchy applied to GMM is proposed; the experimental results are discussed in Section 5; and conclusion is made in Section 6.

2 METAPHORS AND HYPOTHESES

In this section, we will first review GMM [4] and point out the limitations of GMM-based models. Afterwards, an analogy will be made between the attention theory of cognitive psychology and the corresponding background subtraction methods including both GMM-based and non-GMM approaches. Then, related hypotheses will be made.

2.1 GMM Revisited and the Limitations of Related Models

In pixel-wise GMM, the current intensity belongs to an existing Gaussian that represents either background or foreground. The probability of the current intensity X_t is expressed as $P(X_t) = \sum_{k=1}^{K} \omega_{k,t} \cdot \eta(X_t, \mu_{k,t}, \Sigma_{k,t})$, where η is a Gaussian probability density function. K represents the number of Gaussians. $\mu_{k,t}$, $\Sigma_{k,t}$ and $\omega_{k,t}$ refer to the current mean, the covariance matrix and the weight estimation of Gaussian k, respectively. Furthermore, the covariance matrix $\Sigma_{k,t}$ is $\Sigma_{k,t} = \sigma_{k,t} \cdot I$, assuming the red, green, and blue pixel values are independent[1]. A new pixel intensity X_{t+1} is checked against each Gaussian of the pixel-wise GMM in order to find the appropriate one to which it belongs. First, the parameters of the pixel-wise GMM are updated as follows:

$$\begin{aligned}
\omega_{k,t+1} &= (1-\alpha) \cdot \omega_{k,t} + \alpha \cdot M_{k,t+1}, \\
\mu_{k,t+1} &= \mu_{k,t} + \rho \cdot (X_t - \mu_{k,t}) \cdot M_{k,t+1}, \\
\sigma^2_{k,t+1} &= \sigma^2_{k,t} + \rho \cdot \left((X_t - \mu_{k,t})^T (X_t - \mu_{k,t}) - \sigma^2_{k,t} \right) \cdot M_{k,t+1}
\end{aligned} \tag{1}$$

where α and ρ denote the learning rates. Let ρ be $\rho = \alpha \cdot \eta(X_t|\mu_{k,t}, \sigma_{k,t})$. If the k^{th} Gaussian is matched, $M_{k,t+1} = 1$; otherwise, $M_{k,t+1} = 0$. Then Gaussians are re-ordered by $\omega_{k,t}/\sigma_{k,t}$. If intensity X_{t+1} is matched with none of the Gaussian models, the mean $\mu_{K,t}$, the standard deviation $\sigma_{K,t}$ and the weight estimation $\omega_{K,t}$ of the last Gaussian are to be replaced by the current intensity X_{t+1}, an initially high variance and a low prior weight, respectively. Ultimately, the weight estimation $\omega_{k,t+1}$ is re-normalized. The first b Gaussians are chosen as the background model. That is

$$B_{t+1} = \arg\min_b \left(\sum_{k=1}^{b} \omega_{k,t+1} > T \right) \tag{2}$$

where B_{t+1} denotes the current Gaussians representing background. T is a user-defined threshold.

[1] While this is certainly not the case, the assumption allows us to avoid a costly matrix inversion at the expense of some accuracy.

Pixel-wise GMM is especially suitable for the problem of *time of day* and *waving trees*. Yet, it also has shortcomings which is mainly presented as three aspects. Firstly, the most frequently matched state k is supposed to hold a much lower standard deviation $\sigma_{k,t}$ and a firmly high weight $\omega_{k,t}$ as expressed in Equation (1), particularly when the background appearance remains unchanged. That will make the state keep to the first place and hold the priority for matching with the new pixel intensity. Secondly, the weight $\omega_{k,t}$ of matched Gaussian is updated steadily with a fixed learning rate α. Thirdly, Gaussians are chosen to represent background according to Equation (2), which accumulates $\omega_{k,t+1}$ and is regarded as a conservative way to keep background especially when the problem such as *waving trees* occurs. Therefore, pixel-wise GMM keeps its own rhythm to fulfill parameter updating, no matter how rapidly the scene intensity varies. When background changes suddenly, the current state will be wrongly considered to be foreground (e.g. *light switch* and *moved objects*) without any doubt. Actually, these shortcomings can be rightly overcome using a fast learning strategy when immediate background change occurs, once the most frequently matched states can be memorized as a long-term background.

GMSM [19], which tackles the problem of *shadows*, develops pixel-wise GMM by supposing shadows to be associated with frequently seen states labeled as foreground. It improves the parameter updating strategy and incorporates frequently matched shadows into background. In addition, a second pixel-wise GMM is built for maintenance of shadows. Using GMSM, shadows are distinguished from foreground. However, GMSM keeps only short-term background even though it improves the parameter updating strategy of pixel-wise GMM. Moreover, it does not concern any spatial information. When global change of background (e.g. *light switch*) occurs, states corresponding to a former background before the time of change will be pushed to perform a low ranking Gaussian and ultimately discarded.

Suppose that background sudden change keeps spatial consistency. Local texture information or spatial statistic is combined with pixel-wise GMM, and that forms TGMM [17] or HGMM [18]. Some non-GMMs (e.g. LBP [8] and ALPCA [16]) are also this type. Due to consideration of spatial information, these models are effective to *light switch* especially when rapid parameter learning is performed. Yet, they are invalidated once faced with the problems such as *moved objects*, *sleeping person* and *waking person* (namely, *ghosts*), especially when foreground scale is comparative to the size of scene. Except for the storage of short-term background, it is needed to be further discussed how to memorize long-term background.

To the best of our knowledge, MGMM [20] is one of the first GMM-based models that considers memorizing long period background. Despite of its competence for local background sudden change, it still has some problems to be discussed. First, it only focuses on pixel-wise states but not spatial background simultaneously, and that makes a lapse when *light switch* occurs. Second, it needs an initial period for model learning, which excludes *bootstrap* in consideration. Besides, it might be incapable of tackling *moved objects* and *ghosts*, when uncovered background never appears during the learning period. Third, it makes an alternate learning rate for matched states to rapidly adapt to sudden change, which also may absorb *sleeping person* into background. Fourth, it stores the longest memorized short-term background states, which currently keeps a low rank order, into its long-term background memory. That will lead to frequent switches of background states between short-term and long-term memory, especially when background keeps changing repeatedly.

In order to make a robust background subtraction model, improvements have been made to memorize long period background in our previous work [22]. Base on GMM, a framework that contains components representing not only short-term but also long-term background memory is presented. This paper makes a further effort to expound how this framework works and why it is composed of real-time state record, spatial background memory and global difference memory, in order to solve most of the problems except *camouflage* and *foreground aperture*. Besides, a prior perception processor is precisely proposed to replace the simple stationary test in [22]. Before that, statements on the origin of the piecewise memorizing framework are to be made, which is derived from the metaphors of cognitive psychology.

2.2 The Metaphors of Cognitive Psychology

In cognitive psychology, the term selective attention [23] refers to the fact that we usually focus our attention on one or a few tasks or events rather than on many. In other words, the information that people process is divided into the attended and unattended message. Selective attention theory is mainly classified into three categories: filter theory, attenuation theory and late-selection theory, as shown in Figure 1.

We firstly discuss these theories and make several extracts as follows:

- Filter theory predicts that all unattended messages will be filtered out, that is to say, not processed for recognition or meaning. A filter is set to make a selection of what message to process early in the processing, typically before the meaning of the message is identified. Meanwhile, a lower working-memory capacity refers to a decreased ability to actively block the unattended message.
- As for attenuation theory, some meaningful information in unattended messages may still be available, even if it is difficult to recover. Furthermore, incoming messages are subjected to an alternative hierarchical analysis. Moreover, only a few firmed meaningful messages, together with those in a special context, have permanently lowered thresholds.
- Late-selection theory holds that all messages are routinely processed but which message to respond to is selected "late" in processing. In fact, the selection is made in the response to "stimuli", rather than due to the recognition of it. In other words, the selective filter performs after the perception model.

Using metaphor, computers can be also simulated to have "attention". In this research, prevailing background subtraction methods can be described as the result of selective attention. Applying this analogy to GMM, the "matching" step can be

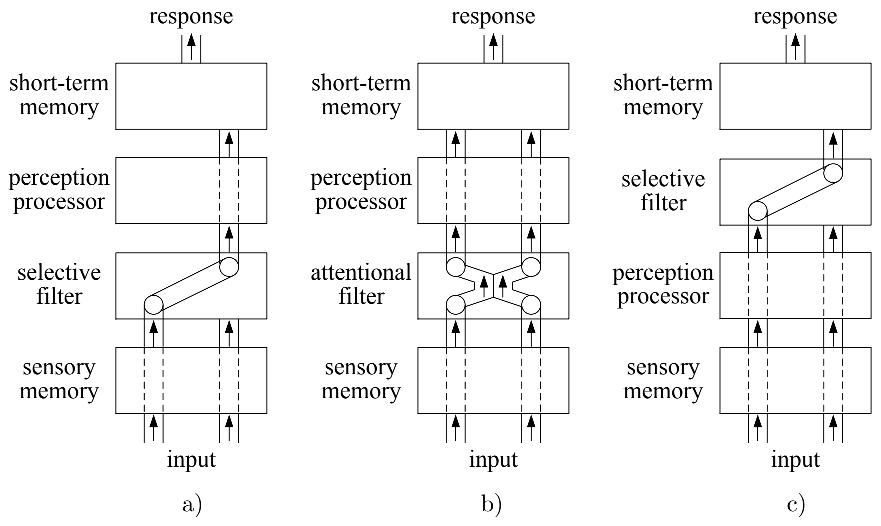


Figure 1. Illustration of selective attention theory [23]. a) Filter theory, b) attenuation theory, c) late-selection theory.

regarded as the act of paying "attention". In other words, the matched Gaussian refers to the attended message; whereas unmatched states correspond to unattended messages. Moreover, the working-memory capacity is equivalent to the number of Gaussians. Thus, prevailing background subtraction methods can be described as the result of selective attention. Correspondingly, background modeling approaches (not solely GMM-based methods) can be classified into following three categories:

Match-concerned model pays attention to matched states. Only frequently matched states are regarded as attended messages and viewed as background. A limited number of states lead to a frequent replacement of background appearances. Just like setting a selective filter before a perception processor as shown in Figure 1 a), the always un-matching state equivalent to the unattended message are discarded. In match-concerned model, states may correspond to Gaussians derived from temporal statistics (for example, GMM [4] and TGMM [17]) or histograms generated from structural information, such as LBP [8].

Meaning-concerned model concentrates on the potential background. Just like setting an attentional filter before a perception processor as illustrated in Figure 1 b), matched states together with meaningful unmatched states are memorized. In fact, the attenuation of unmatched states depends on the meaningful information they contain. Commonly, the meaning of the unmatched states is derived from a spatial consistency (e.g. HGMM [18]), from a permanently lowered threshold that enables frequently seen states labeled as foreground to be background (e.g. GMSM [19]), or even from a tag labeling a background state that was ever in presence (e.g. MGMM [20]). In meaning-concerned model, each state may represent a pixel-wise Gaussian [19, 20], a hierarchical Gaussian [18] or an eigensubspace (e.g., ALPCA [16]).

Perception-concerned model focuses more on the identification of complex environments rather than on state discrimination. That is to say, the perception processor module is laid before the selective filter, as illustrated in Figure 1 c). In fact, even an early stage of perception processing may contribute to good state discrimination results, i.e., states representing the appearances of a long period background can be purposefully memorized. A representative perception-concerned model is P-MGMM, about which details are to be discussed in Section 3 and Section 4.

2.3 Hypotheses of Background Subtraction

According to the metaphors mentioned above, robust background models should firstly possess memorizing ability. The fact that GMM uses exponential forgetting is a case in point. However, exponential forgetting contradicts the purpose of memorizing long period background due to the constraint of memory capacity. To overcome this, a primary stage of perception should be made prior to the attention selection for memorizing. Before that, a long-term image sequence should be divided into short ones and then processed for better perception and memorization of long-term background, which is named as a "piecewise" action. Now what to perceive and memorize becomes a central issue. In order to answer these questions, we propose the following three hypotheses:

- Background modeling essentially derives from a segmented long-term sequence. According to the exponential forgetting of background appearances, long period background must be memorized.
- A prior stage of perception is needed, in accordance with perception-concerned model. That is, special attention to scene general change existing only in short-term image sequences should always be paid in advance.
- Spatial information should be included, considering scene change always meets a certain spatial consistency. The analogous status is also found in the data-driven attention model [23] in cognitive psychology.

Taking the memory capacity into account, we present a segmented stationarity test as the prior perception processor of late-selection theory [23] in order to identify different scene changes generally, and propose a memorizing mechanism applied to GMM to be the corresponding selective filter.

3 PRIOR PERCEPTION PROCESSOR

It is always necessary to detect in advance and rapidly adapt to scene change. Different intensity changes of scene exist only in short period of time. Thus, stationarity test is regarded as a perceptional criterion when we consider pixel-wise intensity values in short-term image sequences to be time series. Inversion number test is viewed as a common stationarity test approach [24]. Hence, we constitute the inversion number of pixel-wise temporal mean values to test the stationarity of segmented time series $\{X_t\}$ and subsequently identify different intensity changes.

3.1 Inversion Number Test

We firstly cut $\{X_t\}(t = 1, 2, \ldots, N)$ into l segmented sequences, each is at a length of M $(N = lM)$. The i^{th} segmented sequence is denoted by $\{X_{i,j}\}$ $(i = 1, 2, \ldots, l;$ $j = 1, 2, \ldots, M)$, where $X_{i,j} = X_{(i-1)M+j}$. The mean value of each segmented sequence is expressed as follows:

$$\mu_i = \frac{1}{M}\sum_{j=1}^{M} X_{i,j}. \tag{3}$$

Meanwhile, we select r pieces of $\{X_{i,j}\}$ to constitute a sub-sequence $\{X_{i,h_j}\}$, where $X_{i,h_j} = X_{i-r+h,j} = X_{(i-r-1+h)M+j}$ $(i = 1, 2, \ldots, l;\ h = 1, 2, \ldots, r; j = 1, 2, \ldots, M)$. Each two neighboring sub-sequences, for instance $\{X_{i,h_j}\}$ and $\{X_{i+1,h_j}\}$, share a $(r-1)-r^{\text{th}}$ overlap. Let X_{i,h_j} $(i = -r+1, -r+2, \ldots, 0)$ be $X_{i,h_j} = X_{1,h_j}$ for convenience. Correspondingly, we get a sub-sequence $\{\mu_{i,h}\}$ from the mean sequence $\{\mu_i\}$, where $\mu_{i,h} = \mu_{i-r+h}$ $(i = 1, 2, \ldots, l;\ h = 1, 2, \ldots, r)$.

As to each segmented sequence in a sub-sequence $\{X_{i,h_j}\}$, a reverse order of $\mu_{i,h}$ $(h = 1, 2, \ldots, r-1)$ is defined, when there is $\mu_{i,j} < \mu_{i,h}$ $(j = h+1, h+2, \ldots, r)$. An inversion number $A_{\mu_{i,h}}$ of $\mu_{i,h}$ is obtained after traversing all the $\mu_{i,j}$ from $\mu_{i,h}$ to $\mu_{i,r}$, i.e., $A_{\mu_{i,h}}$ records the counts when $\mu_{i,h} > \mu_{i,j}$ $(h < j)$. The sum of $A_{\mu_{i,h}}$ in $\{\mu_{i,h}\}$ is defined as follows:

$$A_{\mu_i} = \sum_{h=1}^{r-1} A_{\mu_{i,h}}. \tag{4}$$

We define a random variable $I_{h,j}$ that represents whether or not the ordered pair $\langle \mu_h, \mu_j \rangle$ in $\{\mu_{i,h}\}$ is reverse order. That is

$$I_{h,j} = \begin{cases} 1, & \text{if } \mu_{i,h} > \mu_{i,j}, \\ 0, & \text{others,} \end{cases} \qquad \forall h < j. \tag{5}$$

Thus, the inversion number is expressed as

$$A_{\mu_{i,h}} = \sum_{j=h+1}^{r} I_{h,j}. \tag{6}$$

Assuming the sub-sequence $\{X_{i,h_j}\}$ to be stationary, the random variable $I_{h,j}$ is equiprobable. That is,

$$P(\mu_{i,j} > \mu_{i,h}) = P(\mu_{i,j} \leq \mu_{i,h}) = \frac{1}{2} \tag{7}$$

where $h = 1, 2, \ldots, r$ and $j = h, h+1, \ldots, r$. From Equations (5) and (7) we obtain

$$E(I_{h,j}) = \frac{1}{2} \times 1 + \frac{1}{2} \times 0 = \frac{1}{2}. \tag{8}$$

Then we evaluate the expectation and the variance of the inversion number sum A_{μ_i} in $\{\mu_i\}$. The expectation $E(A_{\mu_i})$ is derived from Equations (4) and (6):

$$E(A_{\mu_i}) = E\left(\textstyle\sum_{h=1}^{r-1} A_{\mu_{i,h}}\right) = \textstyle\sum_{h=1}^{r-1} \sum_{j=h+1}^{r} E(I_{h,j}). \tag{9}$$

When we substitute Equation (8) in Equation (9), the expectation $E(A_{\mu_i})$ is expressed as follows:

$$E(A_{\mu_i}) = \frac{1}{2} \sum_{h=1}^{r-1} (r-h) = \frac{r(r-1)}{4}. \tag{10}$$

On account of the independence of $A_{\mu_{i,h}}$ and $A_{\mu_{i,j}}$ $\forall h, j \in (1, r)$ where $h \neq j$, we calculate the expression of the variance $Var(A_{\mu_i})$ from Equation (4). That is

$$Var(A_{\mu_i}) = \textstyle\sum_{h=1}^{r-1} \left(E\Big(A_{\mu_{i,h}}\Big)^2 - E^2\left(A_{\mu_{i,h}}\right)\right). \tag{11}$$

Substitute the $A_{\mu_{i,h}}$ by Equation (6), and we have

$$Var\left(A_{\mu_i}\right) = \sum_{h=1}^{r-1} \left(\sum_{j=h+1}^{r} E(I_{h,j})^2 + 2C_{r-h}^2\, E\left(I_{h,j} I_{h,k}\right)_{j \neq k} - \left(\sum_{j=h+1}^{r} E\left(I_{h,j}\right) \right)^2 \right). \tag{12}$$

Once Equation (8) is introduced in, $Var(A_{\mu_i})$ in Equation (12) can be simplified as follows:

$$Var(A_{\mu_i}) = \frac{r(r-1)(2r-1)}{24}. \tag{13}$$

3.2 Identification of Intensity Changes

The sum of the inversion number of each sub-sequence $\{\mu_{i,h}\}$, which is derived from Equation (4), forms a corresponding sequence as $\{A_{\mu_i}\}$ $(i = 1, 2, \ldots, l)$. Due to the histogram count of A_{μ_i} to all the pixels in a selected area, it is reasonable to assume that A_{μ_i} is a normal stochastic variable, which is expressed as follows:

$$u_{\mu_i} = \frac{|A_{\mu_i} + 0.5 - E(A_{\mu_i})|}{\sqrt{Var(A_{\mu_i})}}. \tag{14}$$

When $|u_{\mu_i}| \leq 1$, we believe that there is no significant difference between $X_{i,j}$ and $X_{i,k}$ $\forall j \neq k$. That is, $\{X_{i,j}\}$ is a stationary sequence. Due to the spatial hypothesis proposed in Section 2.3, we identify different intensity changes in each sub-sequence $\{X_{i,h_j}\}$ by calculating u_{μ_i} $(i = 1, 2, \ldots, l)$ from Equation (14). That is

$$\left| \max_{y \in \Gamma} (u_{\mu_i}(y)) \right| \leq 1 \tag{15}$$

where Γ represents the selected area. $\max_{y\in\Gamma}(u_{\mu_i}(y))$ denotes the max value of u_{μ_i} in Γ. If Inequation (15) holds, the last segmented sequence $\{X_{i,j}\}$ of the current sub-sequence $\{X_{i,h_j}\}$ contains gradual intensity change; otherwise, there is fast intensity change.

4 HIERARCHICAL MEMORIZING MECHANISM

Hypotheses proposed in Section 2.3 have answered what to perceive and memorize in perception-concerned background model. In order to tackle different scenarios of background such as various illumination conditions, long period background must be memorized. Considering the fast detection and rapid adaptation to scene change, image sequences or video frames are to be segmented into sub-sequences for prior perception of pixel-wise intensity change. It has been solved how to perceive the rapid intensity change that occurs in segmented image sequences in Section 3, let us now consider how to memorize long period background appearances and how to track the rapid intensity change of background.

4.1 P-MGMM Framework

Due to the hypothesis of spatial consistency proposed in Section 2.3, long period background memory must contain models of different scales. Supposing that a pixel-wise mode is regarded as a state, a spatial combination of background state models is identified to be a version, which has been specified in HGMM [18]. Once we perceive current sub-sequences to be non-stationary, parameters of matched pixel-wise states need to be rapidly updated. Therefore, we need to adjust learning rates expressed in Equation (1) as high as possible, which inevitably accelerates background forgetting. This traditional exponential forgetting way makes real-time state record. In order to avoid discarding long period background, additional states at each pixel together with the corresponding versions are utilized to memorize different scales of long-term background. Put another way, spatial background memory containing additional states and versions which represent long period background is needed. Once rapid intensity changes are detected, it is apparently not enough to match current frame with states or versions in either real-time state record or spatial background memory only, especially when new illumination condition appears in scene. Therefore, global difference memory is presented, which can be viewed as the first order difference of spatial background memory.

Narrowing down to GMM, modules of real-time state record, spatial background memory and global difference memory constitute a piecewise memorizing hierarchy, which together with prior perception processor forms P-MGMM framework. As illustrated in Figure 2 a), states, versions and differences are expressed as S, V and D, respectively. When labeled with '*', they represent background appearances. Real-time state record corresponds to a conventional mixture of states. Spatial background memory for long period background storage consists of states and versions.

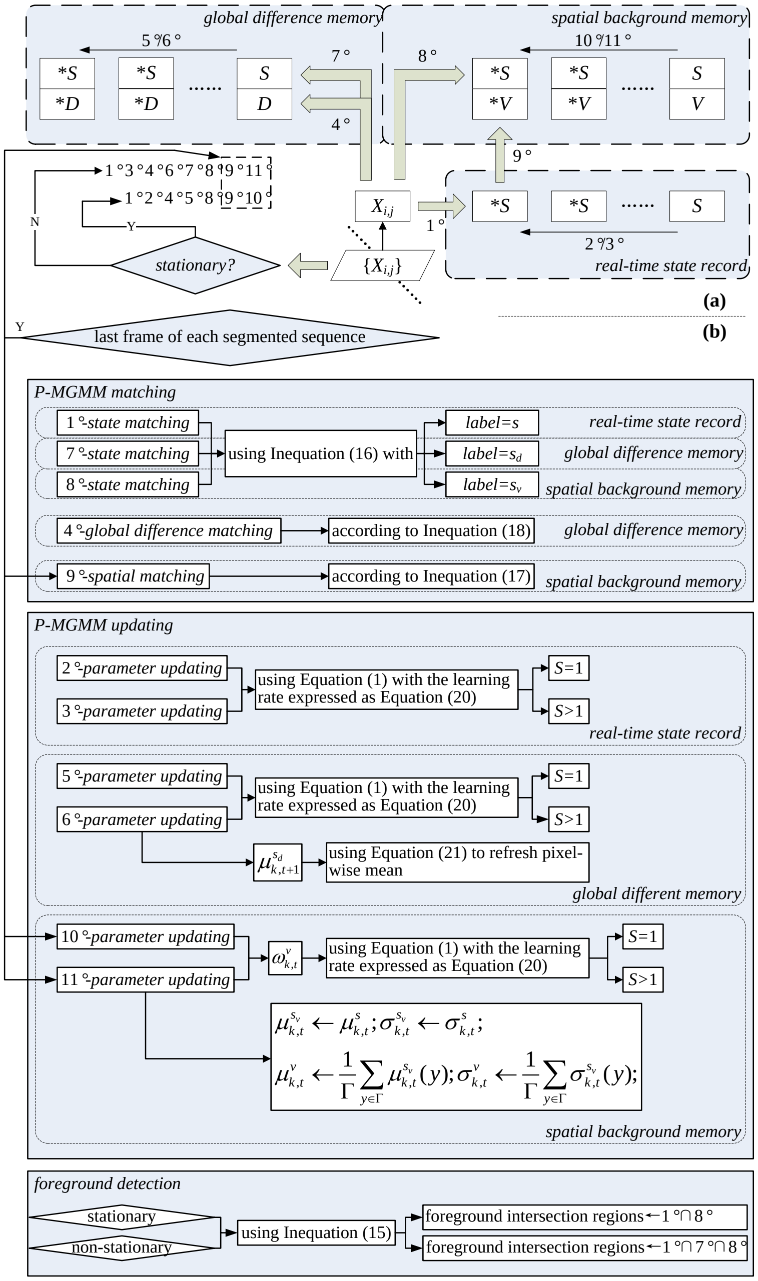

global difference memory
spatial background memory
real-time state record
5°/6°
10°/11°
2°/3°
7°
8°
4°
9°
1°
*S
*D
S
D
*V
V
$X_{i,j}$
$\{X_{i,j}\}$
stationary?
N
Y
1°3°4°6°7°8°/9°11°
1°2°4°5°8°/9°10°
last frame of each segmented sequence
(a)
(b)
P-MGMM matching
1°-state matching
7°-state matching
8°-state matching
using Inequation (16) with
label=s
label=s_d
label=s_v
4°-global difference matching
according to Inequation (18)
9°-spatial matching
according to Inequation (17)
P-MGMM updating
2°-parameter updating
3°-parameter updating
using Equation (1) with the learning rate expressed as Equation (20)
S=1
S>1
5°-parameter updating
6°-parameter updating
$\mu^{s_d}_{k,t+1}$
using Equation (21) to refresh pixel-wise mean
global different memory
10°-parameter updating
11°-parameter updating
$\omega^v_{k,t}$
$\mu^{s_v}_{k,t} \leftarrow \mu^s_{k,t}; \sigma^{s_v}_{k,t} \leftarrow \sigma^s_{k,t};$
$\mu^v_{k,t} \leftarrow \frac{1}{\Gamma}\sum_{y\in\Gamma}\mu^{s_v}_{k,t}(y); \sigma^v_{k,t} \leftarrow \frac{1}{\Gamma}\sum_{y\in\Gamma}\sigma^{s_v}_{k,t}(y);$
foreground detection
stationary
non-stationary
using Inequation (15)
foreground intersection regions ← 1°∩8°
foreground intersection regions ← 1°∩7°∩8°

Figure 2. Sketch map of perception-contained piecewise memorizing framework, i.e. P-MGMM for short. a) P-MGMM framework, b) P-MGMM maintenance algorithm from step 1° to step 10° or 11°.

Together with spatial differences between current intensity values and the means of k states in real-time state record, additional pixel-wise states constitute global difference memory, which is competent to adapt to rapid variations of background. According to P-MGMM, several considerable improvements are able to be made. Concrete improvements are to be introduced following the maintenance steps of P-MGMM framework, as illustrated in Figure 2 b).

4.2 P-MGMM Maintenance

First of all, P-MGMM matching is considered. Different hierarchical matches are defined as pixel intensity values within D times the standard deviations of a distribution, respectively. Then we have:

$$\left| X_t^s(y) - \mu_{k,t}^{label}(y) \right| \leq D \cdot \sigma_{k,t}^{label}(y), \quad label \in \{s, s_v, s_d\}, \tag{16}$$

$$\left| \frac{1}{\Gamma} \sum_{y \in \Gamma} \mu_{1,t}^s(y) - \mu_{k,t}^v \right| \leq D \cdot \sigma_{k,t}^v, \tag{17}$$

$$\left| \frac{1}{\Gamma} \sum_{y \in \Gamma} (X_t^s(y) - \mu_{k,t}^s(y)) - \mu_{k,t}^d \right| \leq D \cdot \sigma_{k,t}^d \tag{18}$$

where $D = 2.5$. Superscripts s, s_v and s_d correspond to state parameters in real-time state record, spatial background memory and global difference memory, respectively. Accordingly, superscripts v and d represent version parameters in spatial background memory and difference parameters in global difference memory. Inequation (16) expresses three state matchings at different parts of the framework. $X_t^s(y)$ represents the current intensity value at location y. $\mu_{k,t}^{label}(y)$ and $\sigma_{k,t}^{label}(y)$ denote the mean and deviation of the k^{th} state in real-time state record, spatial background memory and global difference memory with *label* varying from s to s_v and s_d. As to spatial matching expressed in Inequation (17), spatial mean of the first state in real-time state record is calculated and compared with k means of versions in spatial background memory, when reaching the last frame of each segmented sequence. Besides, a global difference matching is made by comparing spatial differences of current intensity values and the means of k states in real-time state record with k means of differences in global difference memory, as expressed in Inequation (18). Note that state matchings at different parts of the framework (i.e., step 1°, 7° and 8°) expressed in Inequation (16) are applied to foreground detection. Version and difference matching expressed in Inequation (17) and (18) are only used for parameter updating.

We consider real-time state record as a short-term background memory. Like classical GMM [4] which is a match-concerned model, Gaussians in this part only serve as a real-time record of matched states. Put another way, there are only pixel-level state models that exist in real-time state record. Inevitably, the always matched state k is supposed to hold a much lower standard deviation $\sigma_{k,t}$, particularly when the background appearance remains unchanged. Therefore, a fixed lower bound of standard deviation σ_{low} for each state is used. Besides, we re-ordered Gaussians only by $\omega_{k,t}$, while taking the over-learning of the above-mentioned standard deviation into consideration. Then each short-term state in sorted Gaussians is regarded as background, if its weight value is large enough. That is:

$$B_{t+1} = \{\omega_{k,t+1} > T \mid k = 1 \ldots K\} \tag{19}$$

where B_{t+1} and T keep the same meaning, as has been explained in Equation (2). Last but foremost, we define the learning rate as follows:

$$\alpha_t = \alpha \cdot S \tag{20}$$

where the parameter $S > 1$ when there is rapid intensity change, and $S = 1$ in the other case (i.e., step 2° or 3°). Parameters of real-time state record are updated using Equation (1), once state matching in real-time state record corresponding to step 1° is performed.

After parameters in real-time state record are refreshed, global difference matching regarded as step 4° is made using Inequation (18). Once variations in scene are perceived considering current segmented image sequence to be non-stationary, sequential background appearances derived from continuous intensity changes emerge. States and versions in either real-time state record or spatial background memory are incompetent at accommodating new background appearances in time. Therefore, global difference memory is devoted to adapting to the continuity of background intensities. Global difference memory consists of GMMs representing global differences and pixel-wise states, respectively. Following the parameter updating strategy in real-time state record, the current mean $\mu^d_{k,t}$, the standard deviation $\sigma^d_{k,t}$ and the weight estimation $\omega^d_{k,t}$ of the matched global difference are updated using Equation (1) with the learning rate α_t switching between different intensity changes as expressed in Equation (20), i.e. step 5° or 6°. According to the rightly matched global difference, corresponding states are selected for parameter updating. In fact, only states referring to the inversion of foreground detection results derived from both real-time state record and spatial background memory are needed to be refreshed. The standard deviation $\sigma^{s_d}_{k,t}$ of selected state k is updated using Equation (1). As to pixel-wise mean $\mu^{s_d}_{k,t}$, it is refreshed using Equation (1) when the intensity values appear stationary in segmented sequences. Once there is obvious intensity change in scene, the replacement of $\mu^{s_d}_{k,t}$ is expressed as follows:

$$\mu^{s_d}_{k,t+1} = \mu^{s_d}_{k,t} + \mu^d_{k,t}. \tag{21}$$

If no global match exists, the last global difference and pixel-wise state need to be re-initialized.

Parameter updating in real-time state record and global difference memory reinforce foreground detection results. Considering the stationarity of segmented image sequences, pixels representing foreground are labeled using either step 1° or an intersection of step 1° and 7°. In order to differ foreground from long period background appearance, state matching is also performed in spatial background memory using Equation (16) and considered as step 8°. Unlike MGMM [20], P-MGMM memorizes the whole of the first states in real-time state record as the latest background appearances once at the end of each segmented image sequence. That is, the most probable background of real-time state record is preserved. Concretely, the most probable background appearance B_t^1 representing the spatial mean of the first states in real-time state record is checked against each global Gaussian of background version in spatial background memory (i.e., step 9°), as expressed in Inequation (17). Nearest neighboring version matching is considered rather than order-first state matching. As to stationary sub-sequences, parameters of states and versions in spatial background memory are updated using Equation (1), as step 10°. When rapid intensity changes appear in scene, the weight estimation of the matched version $\omega_{k,t}^v$ and that of the corresponding states are updated using Equation (1) with its learning rate expressed in Equation (20). The mean $\mu_{k,t}^{s_v}$ and the standard deviation $\sigma_{k,t}^{s_v}$ of pixel-wise states corresponding to the matched version are derived from the first Gaussian in the real-time state record. Accordingly, the mean $\mu_{k,t}^v$ and the standard deviation $\sigma_{k,t}^v$ of the matched version are obtained from the spatial average of $\mu_{k,t}^{s_v}$ and $\sigma_{k,t}^{s_v}$. And that is step 11°. If no match exists, the last states in spatial background memory obtained from the descending order of $\omega_{k,t}^v$ should be replaced by B_t^1 at the end of each segmented sequence.

5 EXPERIMENTAL RESULTS AND ANALYSIS

In this section, the reason for introducing prior perception processor and hierarchical memorizing mechanism will be described. Using P-MGMM, experimental results of foreground detection and corresponding analysis are to be made. Matlab R2012b is selected as the experimental platform.

5.1 Reason for Introducing P-MGMM Using Experiment

First, an illustration why we propose three hypotheses of background subtraction is made. Let us examine a long-term image sequence from PETS01[2] (see Figure 3 a)). In Figure 3 b), a long-term intensity sequence labeled with line '-' is sampled at a fixed pixel of dataset PETS01-D3-T-C1. Stable background appearances frequently emerge; moreover, it can be seen that obvious intensity changes indicated in Figure 3 b) actually exist in short-term sequences. Therefore, a 'piecewise' action is made, i.e., we divide the long-term sequence into short-term ones of equal length with the endpoints labeled with '*'. The short-term sequences obtained are naturally categorized into two types. One represents a stationary sequence without any apparent intensity change. The other attends to the obvious intensity change with

[2] `http://ftp.pets.rdg.ac.uk/PETS2001`

great difference. The same process is in progress at the pixels of the selected area depicted in Figure 3 a), so that a certain spatial correlation is found. In light of these observations, we propose three hypotheses of background subtraction mentioned in Section 2.3.

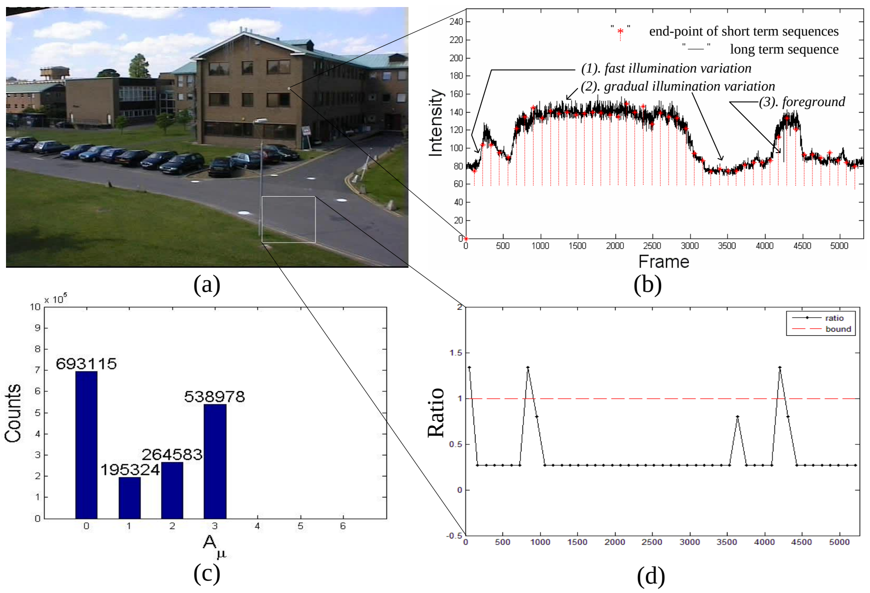


Figure 3. Illustration of the statistical characteristics derived from PET01. a) A long term image sequence with varying illumination, b) a time sequence of the selected pixel, c) a histogram of A_μ derived from the selected area, d) a segmented stationarity test result in each sub-sequence $\{X_{i,h_j}\}$ ($i = 1, 2, \ldots, 47$; $h = 1, 2, \ldots, 4$; $j = 1, 2, \ldots, 112$) of the selected area.

The proposed hypotheses have indicated the importance of piecewise memorizing framework with a primary stage of perception. Therefore, the reason for introducing prior perception processor is further illustrated. The sum of the inversion number of each sub-sequence $\{\mu_{i,h}\}$, which is derived from Equation (4), forms a corresponding sequence as $\{A_{\mu_i}\}$ ($i = 1, 2, \ldots, l$). We calculate A_{μ_i} in the selected area of Figure 3 a) to establish soundness of segmented stationarity test. In Figure 3 c), a histogram of A_{μ_i} is counted to all the pixels in the selected area. Let $r = 4$. Thus, $E(A_{\mu_i}) = 3$, as calculated by Equation (10). Due to the histogram count of A_{μ_i}, it is reasonable to assume that A_{μ_i} is a normal stochastic variable, as expressed in Equation (14).

We also identify different intensity changes in each sub-sequence $\{X_{i,h_j}\}$ of the selected area in dataset PETS01-D3-T-C1 by testing Inequation (15) to demonstrate its effectiveness, as shown in Figure 3 d). In order to indicate the reasonable structure

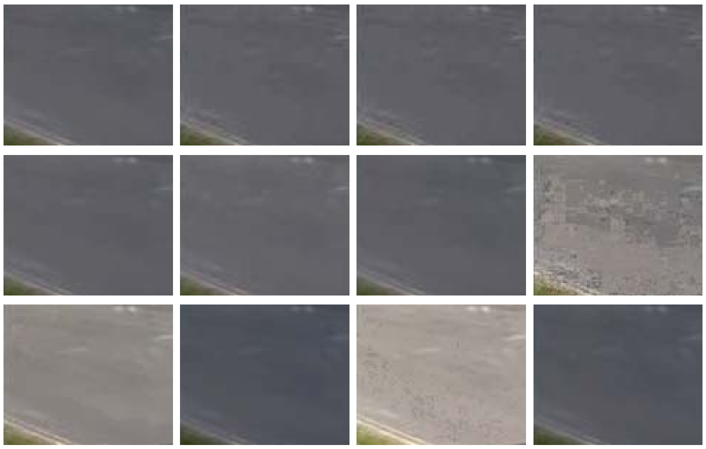


Figure 4. Long period background appearances of different video clips derived from PETS01-D3-T-C1. Each row depicts sorted background versions of the selected area in different clips, as shown in Figure 3. The 1st row corresponds to the 1st video clip. The 2nd row represents background states of the 8th video clip under fast illumination change. The last row illustrates long period background states kept in the 47th video clip. It is obvious that fast intensity change interferes with accurately memorizing long period background.

of our P-MGMM containing a spatial background memory for a long period memory, we experiment with the selected area illustrated in Figure 3 a). The memorized long period background states of different segmented image sequences are shown in Figure 4.

5.2 P-MGMM for Foreground Detection

In order to demonstrate the reasonability and the effectiveness of P-MGMM to foreground detection under complex environments, we tested and compared P-MGMM with seven statistical or structural background subtraction methods (including ALPCA [16], LBP [8], GMM [4], TGMM [17], HGMM [18], GMSM [19] and MGMM [20]) using 12 datasets from four databases. Each long-term video is simply segmented to ensure that each video clip has almost the same length, because of the observation that the detection results of foreground are independent of different l and M values. All the parameters of P-MGMM are user-settable. Besides, the optimal parameter values of comparative approaches for each dataset are selected. Experiments in [22] are repeated due to adding a prior perception processor in this work, as illustrated in Figure 5, Figures 6, 7 and 10. Let r and c be the row and column size of a frame. Therefore, the time and memory complexity of P-MGMM can be expressed as $O(Mrc)$ and $O(Klrc)$, respectively. The overall memory usage for processing each long-term video can be calculated by $bytes \times rc(KM + 2Kl)$, where *bytes* represents the bytes needed for storage of each pixel. In addition, we listed the processing time of P-MGMM and several comparative background subtraction methods on F & GLC[3], as shown in Table 1.

Background Model	Average Processing Time (second/frame)	
	Gradual Illumination Change	Fast Illumination Change
ALPCA [16]	3.6102	4.6426
TGMM [17]	0.6927	0.9545
HGMM [18]	0.8367	1.1354
MGMM [20]	0.0825	0.1037
P-MGMM	0.0992	0.1250

Table 1. Average processing time of a single frame on F&GLC

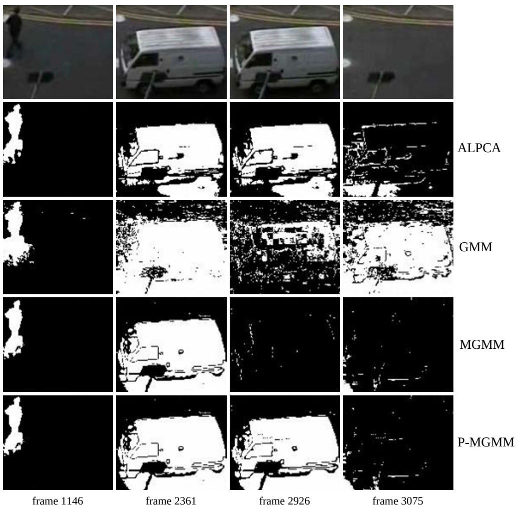


Figure 5. Experimental results on PETS01-D1-T-C2 (l = 27 and M = 119) containing *ghosts*. Each column depicts the detection results of the chosen area at different frames in different segmented video clips. Parameters of P-MGMM for PETS01-D1-T-C2 are given as follows: $K = 3$, $T = 0.34$, $\sigma_{low} = 2$, $\alpha = 0.001$, and $S = 3$ when the sub-sequence is non-stationary.

Ghosts. PETS01-D1-T-C2 is included for solving the problem of *ghosts* resulting from the exponential decay of long period background. Approaches such as ALPCA and MGMM keep the ability in memorizing background. That

[3] F & GLC is a dataset made by the second author of this paper in Harbin Institute of Technology where he did his doctorate.

is, these methods are immune from the appearances of *ghosts* when the occluded object used to be there still moves away. Therefore, we detect objects appearing in the selected area of PETS01-D1-T-C2 using ALPCA, GMM, MGMM and P-MGMM. Corresponding detection results display in rows, as illustrated in Figure 5. Still foreground is misclassified into background using GMM and MGMM (see column 3) due to absence of spatial consideration. GMM also performs poorly when the van is moving away (see column 4). As to ALPCA, there are several uncovered background pixels regarded as foreground because of the large scale of foreground compared to that of the scenario (see column 4). Due to its spatial background memory, it is observed that P-MGMM works well on resisting the object absorption from background and *ghosts*.

Light switch. Here we select PETS01-D3-T-C1 and PETS01-D3-T-C2 which contains *light switch* and *waving trees*. Methods including ALPCA, TGMM, HGMM and MGMM are viewed to be effective on solving the problem of *light switch*. First we perform P-MGMM and other algorithms on PETS01-D1-T-C1 and find that P-MGMM works better than most of the other methods except for some foreground apertures (see Figure 6). This phenomenon derives from using global difference memory, which can be overcome using erosion and dilation operations. Then we apply these algorithms to PETS01-D1-T-C2. The experimental results corresponding to ALPCA, GMM, TGMM, HGMM, MGMM and P-MGMM display in rows, as illustrated in Figure 7. It is observed that ALPCA is impervious to fast illumination change. However, it can be seen from Figure 7 that *waving trees* do interfere with the detection results due to frequent switches of components. Obviously, GMM, TGMM, HGMM and MGMM are incompetent at handling complex background illumination variations. P-MGMM generally performs better than the other comparative methods on discrimination between background varying illumination and foreground. Anyway, P-MGMM also shows its imperfections in *camouflage* and *foreground aperture*.

In order to present quantitative detection results, we select models adaptive to fast illumination change (i.e., ALPCA, TGMM, HGMM, MGMM and P-MGMM), and make comparisons on F&GLC. The experimental results are shown in Figure 8. ALPCA produces false positives when pedestrians of large scale pass by, because it is restricted by the number of learning frames and the scale of illumination scenarios compared to that of pedestrians. Background illumination is falsely considered to be foreground in TGMM confronted with weak texture features, although every parameter set of its regional texture measure is tried to operate on complex illumination variations. HGMM, which is only adaptive to ever-present fast illumination change, is to lapse when new global fast illumination change appears. In MGMM, initially memorizing step of limited pixel-wise states over training time is rendered ineffective by complex varying intensities, especially when new state with low learning rate appears in testing. With a much higher precision, P-MGMM performs better than

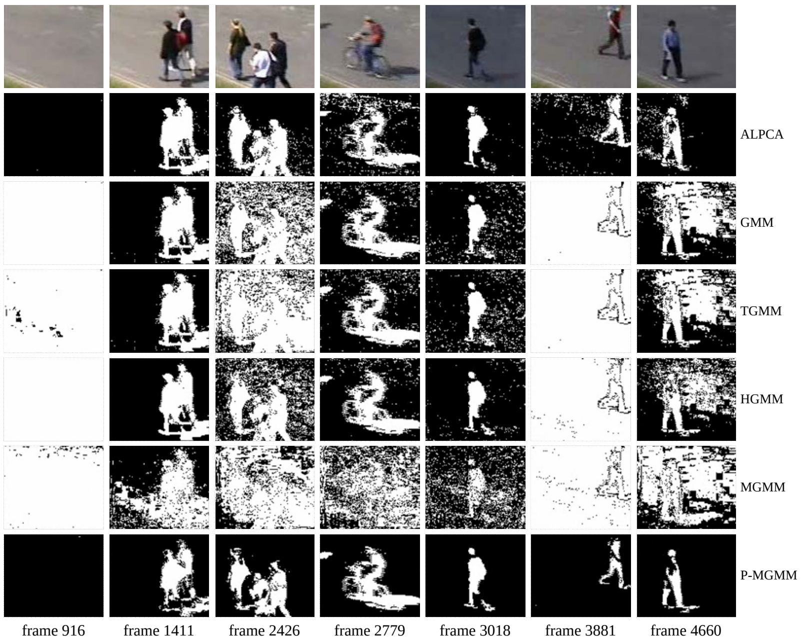


Figure 6. Experimental results on PETS01-D3-T-C1 under complex illumination variations with $l = 47$, $M = 102$. Each column illustrates the detection results of the chosen area at different frames in different video clips. Parameters of P-MGMM for PETS01-D3-T-C1 are given as follows: $K = 3$, $T = 0.34$, $\sigma_{low} = 1$, $\alpha = 0.005$, and $S = 3$ when the sub-sequence is non-stationary.

most of the other methods on discrimination between fast illumination change and foreground, for its instant identification of illumination changes based on a prior segmented stationarity test, rapid adaptation to intensity change of background and its capacity of memorizing long period background memory. On the other hand, more foreground *aperture* appears in the detection results of P-MGMM with a lower recall because of the similarity between foreground and long period background. However, this problem can be solved using morphological methods.

Shadows. CVRR-IR[4] are selected as an evaluation of shadow removal. Approaches such as TGMM and GMSM are viewed as effective methods for shadow removal. TGMM introduces an intensity integration at gray level. As to GMSM, a limitation of high appearance frequency is adopted in color space. In fact, elaborate experimental results indicate that TGMM and GMSM only works on weak shadow. Considering the real-time need of processing time, the gray-level intensity integration of TGMM is embedded into our P-MGMM. An experiment

[4] `http://cvrr.ucsd.edu/aton/shadow/`

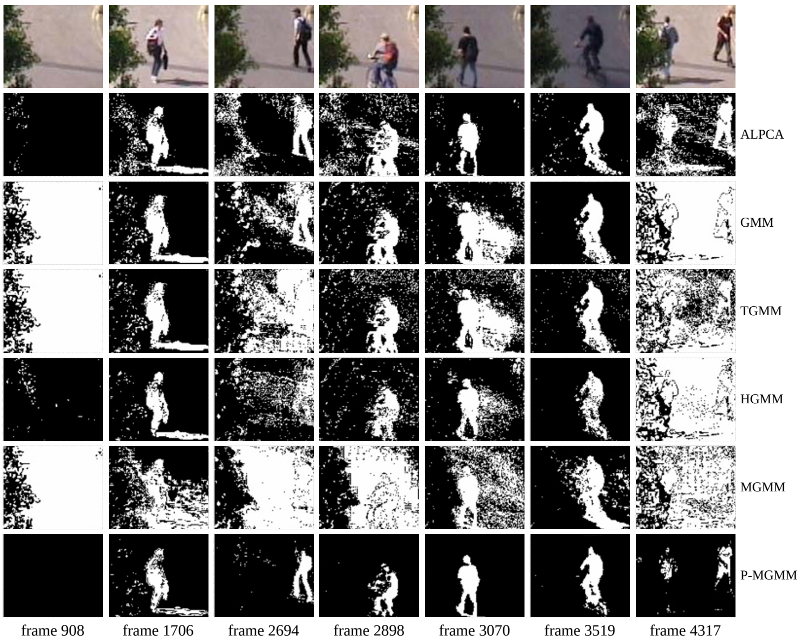


Figure 7. Experimental results on PETS01-D3-T-C2 accompanied with waving trees under complex illumination changes ($l = 53$, $M = 101$). Each column illustrates the detection results of the chosen area at different frames in different video clips. Parameters of P-MGMM for PETS01-D3-T-C2 are given as follows: $K = 5$, $T = 0.21$, $\sigma_{low} = 1$, $\alpha = 0.005$, and $S = 3$ when the sub-sequence is non-stationary.

is made on foreground detection of CVRR-S-I. The experimental results are illustrated in Figure 9. It is observed that P-MGMM keeps the same detection results as TGMM.

Wallflowers. In order to test our P-MGMM on problems besides *ghosts*, *light switch* and *shadows* and show its effectiveness with various disturbances of complex environments, qualitative and quantitative experiments are made on Wallflower[5] which includes seven sequences representing typical problems (i.e., LS, TD, WT, C, B, FA, MO). Together with P-MGMM, comparative algorithms are implemented. Qualitative and quantitative results are listed in Figure 10 and Table 2, which demonstrate the effectiveness of P-MGMM on foreground detection.

[5] `http://research.microsoft.com/en-us/um/people/jckrumm/wallflower/testimages.htm`

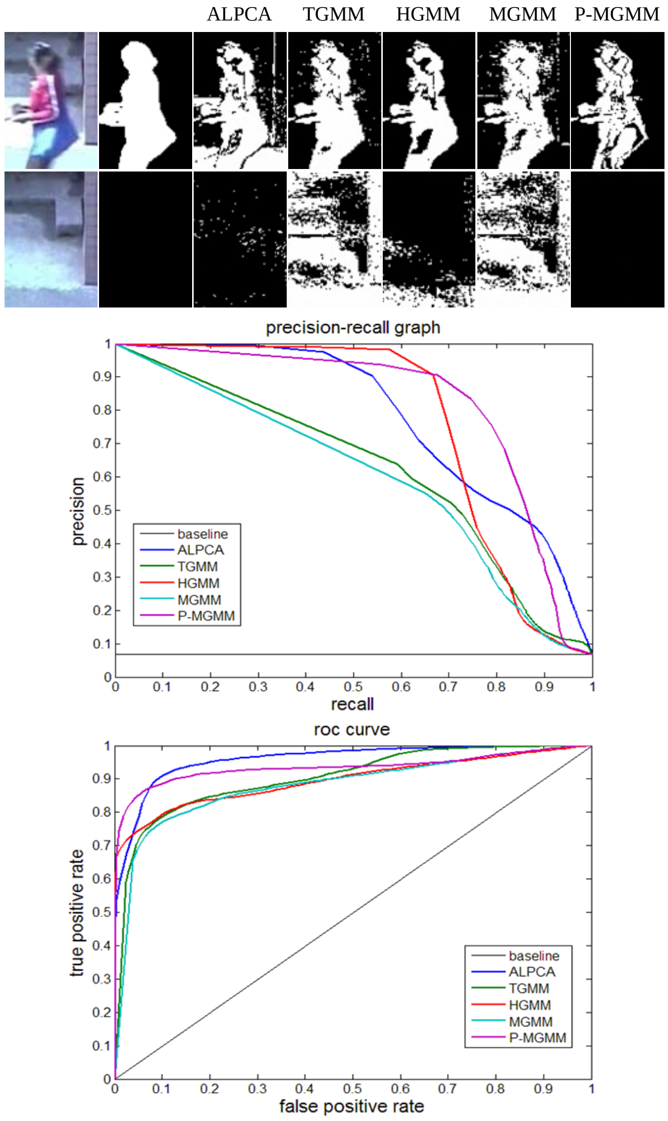


Figure 8. Experimental results on F & GLC containing an extreme fast illumination change accompanied with a pedestrian on high illumination condition ($l = 54$, $M = 118$). The 1st and 2nd column of row 1 and row 2 illustrate the chosen area on different frames and the corresponding ground truths of the 40th video clip. The 1st row shows the detection results of a pedestrian at frame 73. The 2nd row depicts the immunity of the selected methods to fast illumination change at frame 110. The 3rd and 4th row illustrate the precision-recall and ROC curve of the comparative algorithms, respectively. Parameters of P-MGMM for F & GLC are given as follows: $K = 3$, $T = 0.34$, $\sigma_{low} = 2$, $\alpha = 0.005$, and $S = 3$ when the sub-sequence is non-stationary.

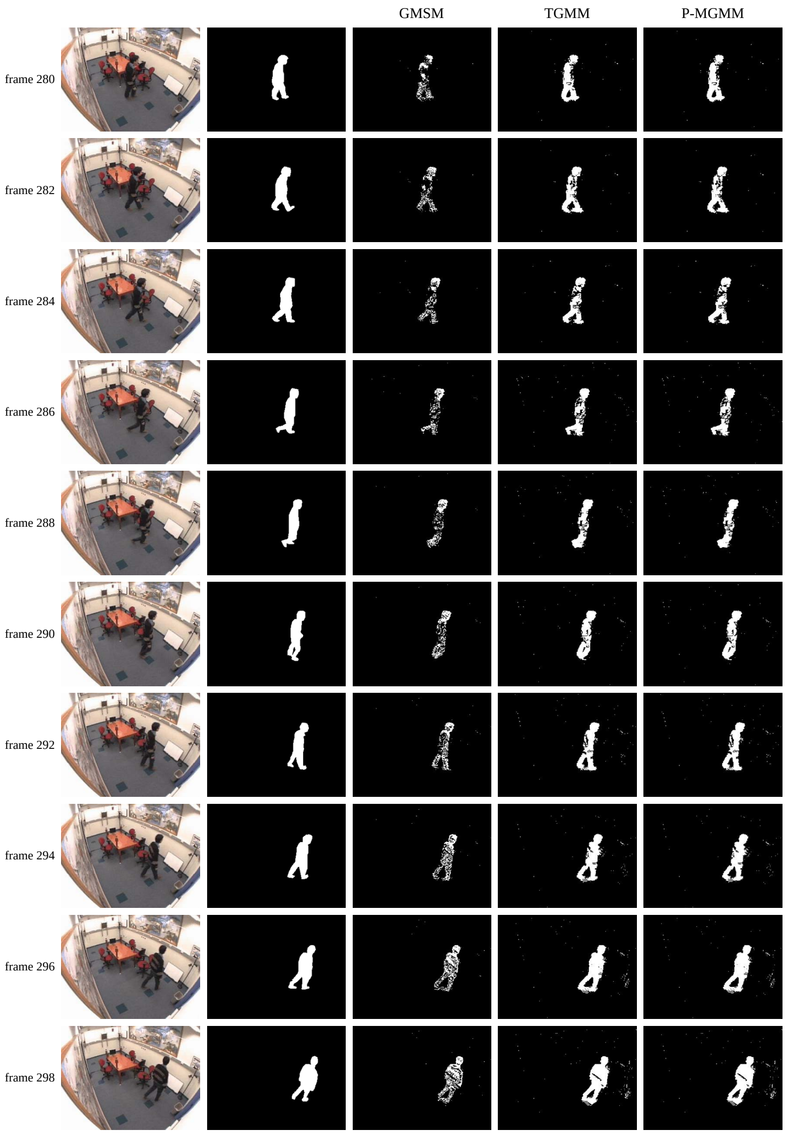


Figure 9. Experimental results on CVRR-S-I containing weak shadows with a pedestrian ($l = 4$, $M = 75$). Rows depict the detection results at a two-frame interval from frame 280 to frame 298. Columns from left to right correspond to original frames, ground truths, detection results of GMSM, TGMM and P-MGMM. Parameters of P-MGMM for CVRR-S-I are same as those for F & GLC.

		ALPCA [16]	LBP [8]	GMM [4]	TGMM [17]	HGMM [18]	GMSM [19]	MGMM [20]	P-MGMM
LS	Precision	0.2111	0.1763	0.1503	0.1504	0.1080	0.1359	0.1549	0.6914
	Recall	0.8884	0.6186	0.8589	0.8589	0.5513	0.6263	0.8927	0.7071
TD	Precision	0.7092	0.9781	0.4558	0.3740	0.4393	0.1643	0.3017	0.9421
	Recall	0.3372	0.2597	0.7261	0.7532	0.7196	0.2435	0.6828	0.5258
WT	Precision	0.5717	0.8663	0.6069	0.5924	0.7394	0.7219	0.6731	0.8198
	Recall	0.8451	0.1707	0.8512	0.8694	0.5480	0.9556	0.8305	0.5406
C	Precision	0.9572	0.8778	0.6789	0.6789	0.7394	0.8182	0.6520	0.8128
	Recall	0.8906	0.7347	0.9835	0.9835	0.0792	0.8769	0.8464	0.8718
B	Precision	0.2964	0.5854	0.4630	0.4747	0.6897	0.6588	0.5521	0.6797
	Recall	0.5925	0.1149	0.5238	0.5579	0.3959	0.5665	0.4315	0.3860
FA	Precision	0.8117	0.8234	0.2478	0.2478	0.2511	0.2745	0.2018	0.6680
	Recall	0.5115	0.5990	0.8033	0.8033	0.6916	0.3969	0.6215	0.5476
MO	Precision	0	0	0	0	0	0	0	0
	Recall	–	–	–	–	–	–	–	–

Table 2. Quantitative comparison on Wallflower

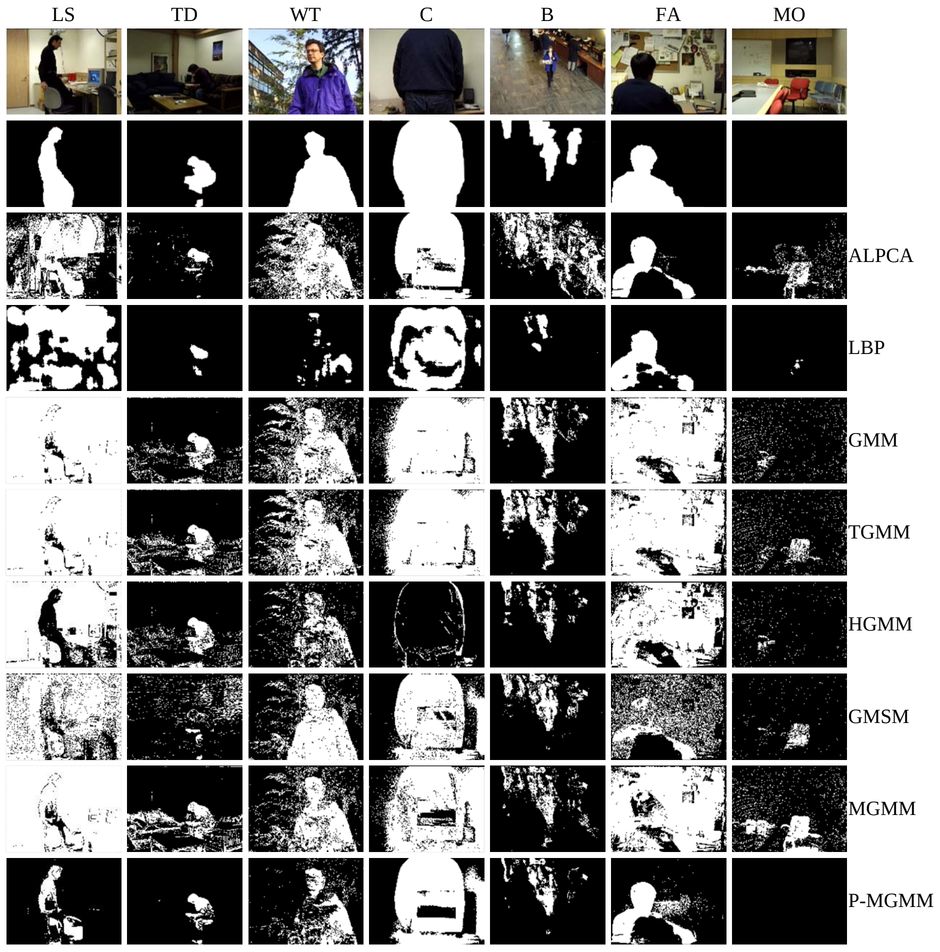


Figure 10. Experimental results on Wallflower with $M = 100$. The 1st and 2nd row illustrate the original frame and corresponding ground truth. Detection results including ALPCA, LBP, GMM, TGMM, HGMM, GMSM, MGMM and P-MGMM are illustrated from the 3rd to the 10th row.

Of course, the problems of *camouflage* and *foreground aperture* are left to be solved using erosion and dilation operations.

6 CONCLUSION

A piecewise memorizing framework, which is capable of solving most typical problems on background subtraction, especially forgetting of long-term memory background, is applied to GMM in this paper. Inspired by the metaphors of psychological selective attention theory, a prior perception-concerned recognition for stationary intensity test is presented, followed by a hierarchical memorizing mechanism containing real-time state record, spatial background memory and global difference memory. Real-time state record duplicates establishment and maintenance of prevailing exponential forgetting models such as GMM and LBP for short-term background memory. Enlightened from spatial information for fast adaptation to background variations (e.g. ALPCA and HGMM) and hierarchical memory strategy for enlarging memory capacity (e.g. MGMM), spatial background memory is developed. In order to ensure the robustness of the framework, global difference memory is designed and can be partially viewed as the first order difference of spatial background memory. Experimental results with various benchmark sequences have quantitatively and qualitatively demonstrated the effectiveness of our P-MGMM compared with many other statistical and structural background models on foreground detection with various disturbances of complex environments. Next, we will concentrate on applying the proposed framework to other background models in the literature.

Acknowledgements

This work was partially supported by the Specialized Personnel Start-up Grant (No. 41112419) and Fundamental Research Funds for the Central Universities (No. 2572017CY08) from Northeast Forestry University.

REFERENCES

[1] Zhou, X.—Yang, C.—Yu, W.: Moving Object Detection by Detecting Contiguous Outliers in the Low-Rank Representation. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 35, 2013, No. 3, pp. 597–610.

[2] Srinivasan, R.—Rudolph, C.—Roy-Chowdhury, A. K.: Computerized Face Recognition in Renaissance Portrait Art: A Quantitative Measure for Identifying Uncertain Subjects in Ancient Portraits. IEEE Signal Processing Magazine, Vol. 32, 2015, No. 4, pp. 85–94, doi: 10.1109/MSP.2015.2410783.

[3] Chen, L. L.—Wang, W.—Panin, G.—Knoll, A.: Hierarchical Grid-Based Multi-People Tracking-by-Detection with Global Optimization. IEEE Transactions on Image Processing, Vol. 24, 2015, No. 11, pp. 4197–4212, doi: 10.1109/TIP.2015.2451013.

[4] Stauffer, C.—Grimson, W. E. L.: Learning Patterns of Activity Using Real-Time Tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, 2000, No. 8, pp. 747–757, doi: 10.1109/34.868677.

[5] Elgammal, A.—Harwood, D.—Davis, L.: Non-Parametric Model for Background Subtraction. In: Vernon, D. (Ed.): Computer Vision – ECCV 2000. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 1843, 2000, pp. 751–767.

[6] Sheikh, Y.—Shah, M.: Bayesian Modeling of Dynamic Scenes for Object Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 27, 2005, No. 11, pp. 1778–1792, doi: 10.1109/TPAMI.2005.213.

[7] Monnet, A.—Mittal, A.—Paragios, N.—Ramesh, V.: Background Modeling and Subtraction of Dynamic Scenes. Proceedings of the 9th IEEE International Conference on Computer Vision (ICCV '03), Nice, October 2003, pp. 1305–1312, doi: 10.1109/ICCV.2003.1238641.

[8] Heikkilä, M.—Pietikäinen, M.: A Texture-Based Method for Modeling the Background and Detecting Moving Objects. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, 2006, No. 4, pp. 657–662, doi: 10.1109/TPAMI.2006.68.

[9] Maddalena, L.—Petrosino, A.: A Self-Organizing Approach to Background Subtraction for Visual Surveillance Applications. IEEE Transactions on Image Processing, Vol. 17, 2008, No. 7, pp. 1168–1177, doi: 10.1109/TIP.2008.924285.

[10] Kim, K.—Chalidabhongse, T. H.—Harwood, D.—Davis, L.: Real-Time Foreground-Background Segmentation Using Codebook Model. Real-Time Imaging, Vol. 11, 2005, No. 3, pp. 172–185.

[11] Seki, M.—Wada, T.—Fujiwara, H.—Sumi, K.: Background Subtraction Based on Cooccurrence of Image Variations. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '03), Madison, June 2003, pp. 65–72, doi: 10.1109/CVPR.2003.1211453.

[12] Toyama, K.—Krumm, J.—Brumitt, B.—Meyers, B.: Wallflower: Principles and Practice of Background Maintenance. Proceedings of the 7th IEEE International Conference on Computer Vision (ICCV '99), Corfu, September 1999, pp. 255–261, doi: 10.1109/ICCV.1999.791228.

[13] Liu, C.—Yuen, P.—Qiu, G.: Object Motion Detection Using Information Theoretic Spatio-Temporal Saliency. Pattern Recognition, Vol. 42, 2009, No. 42, pp. 2897–2906.

[14] Zhao, X.—Satoh, Y.—Takauji, H.—Kaneko, S.—Iwata, K.—Ozaki, R.: Object Detection Based on a Robust and Accurate Statistical Multi-Point-Pair Model. Pattern Recognition, Vol. 44, 2011, No. 6, pp. 1296–1311.

[15] Chen, Y.—Chen, C.—Huang, C.—Hung, Y.: Efficient Hierarchical Method for Background Subtraction. Pattern Recognition, Vol. 40, 2007, No. 10, pp. 2706–2715.

[16] Dong, Y.—DeSouza, G. N.: Adaptive Learning of Multi-Subspace for Foreground Detection under Illumination Changes. Computer Vision and Image Understanding, Vol. 115, 2011, No. 1, pp. 31–49.

[17] Tian, Y.—Senior, A.—Lu, M.: Robust and Efficient Foreground Analysis in Complex Surveillance Videos. Machine Vision and Applications, Vol. 23, 2012, No. 5, pp. 967–983.

[18] Sun, Y.—Yuan, B.: Hierarchical GMM to Handle Sharp Changes in Moving Object Detection. Electronic Letters, Vol. 40, 2004, No. 13, pp. 801–802.

[19] Martel-Brisson, N.—Zaccarin, A.: Learning and Removing Cast Shadows Through a Multidistribution Approach. IEEE Transactions on Pattern Anal-

ysis and Machine Intelligence, Vol. 29, 2007, No. 7, pp. 1133–1146, doi: 10.1109/TPAMI.2007.1039.

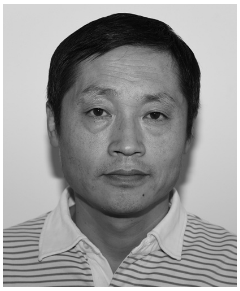
[20] Qi, Y.—Wang, Y.: Memory-Based Gaussian Mixture Modeling for Moving Object Detection in Indoor Scene with Sudden Partial Changes. Proceedings of the 10th International Conference on Signal Processing (ICSP '10), Beijing, October 2010, pp. 752–755, doi: 10.1109/ICOSP.2010.5655913.

[21] Friedman, N.—Russell, S.: Image Segmentation in Video Sequences: A Probabilistic Approach. Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI '97), Paperback, August 1997, pp. 175–181.

[22] Zhao, W.—Zhao, X. D.—Liu, W. M.—Tang X. L.: Long-Term Background Memory Based on Gaussian Mixture Model. Proceedings of Visual Communications and Image Processing (VCIP 2013), 2013, Kuching, November 2013, pp. 1–5.

[23] Galotti, K. M.: Cognitive Psychology in and out of the Laboratory. 4th ed. Thomson Wadsworth Press, Belmont, CA, 2008.

[24] Priestley, M.: Non-Linear and Non-Stationary Time Series Analysis. Academic Press, London, UK, 1988.



Songbo Liu works in the School of Computer Science, Harbin Institute of Technology, Harbin, China. He is now Assistant Professor in Harbin Institute of Technology. His research interests focus on pattern recognition, embedded systems, and intelligent control.

Xudong Zhao works in the College of Information and Computer Engineering, Northeast Forestry University, Harbin, China. He received his Bachelor, Master and Ph.D. degrees from Harbin Institute of Techology in 2003, 2007 and 2013, respectively. After that, he pursued his postdoctoral fellowship in Chinese University of Hong Kong, and now he is Assistant Professor in Northeast Forestry University. His research interests focus on pattern recognition, machine learning, and biostatistics.

Xianglong Tang works in the School of Computer Science, Harbin Institute of Technology, Harbin, China. He is now Professor in Harbin Institute of Technology. His research interests focus on pattern recognition, machine learning, and image processing.

Computing and Informatics, Vol. 37, 2018, 894–914, doi: 10.4149/cai_2018_4_894

AN EFFICIENT ITEMSET REPRESENTATION FOR MINING FREQUENT PATTERNS IN TRANSACTIONAL DATABASES

Savo Tomović, Predrag Stanišić

University of Montenegro
Faculty of Mathematics and Natural Sciences
Džordža Vašingtona bb
81 000 Podgorica, Montenegro
e-mail: {savot, pedjas}@ac.me

Abstract. In this paper we propose very efficient itemset representation for frequent itemset mining from transactional databases. The combinatorial number system is used to uniquely represent frequent k-itemset with just one integer value, for any $k \geq 2$. Experiments show that memory requirements can be reduced up to 300 %, especially for very low minimal support thresholds. Further, we exploit combinatorial number schema for representing candidate itemsets during iterative join-based approach. The novel algorithm maintains one-dimensional array *rank*, starting from $k = 2^{\text{nd}}$ iteration. At the index r of the array, the proposed algorithm stores unique integer representation of the r^{th} candidate in lexicographic order. The *rank* array provides joining of two candidate k-itemsets to be $O(1)$ instead of $O(k)$ operation. Additionally, the *rank* array provides faster determination which candidates are contained in the given transaction during the support count and test phase. Finally, we believe that itemset ranking by combinatorial number system can be effectively integrated into pattern-growth algorithms, that are state-of-the-art in frequent itemset mining, and additionally improve their performances.

Keywords: Frequent itemset mining, Apriori algorithm, combinatorial number system

Mathematics Subject Classification 2010: 68P20, 68P30, 68T30

1 INTRODUCTION

The discovery of frequent itemsets (FI) was introduced in [4] as the first and the most time-consuming phase of the process of finding association rules. The association rule problem is a very important problem in data mining that occupies researchers' attention for decades, with numerous practical applications including market basket analysis, network intrusion detection and pattern discovery in biological applications [8, 9].

An example of association rule from transactional database might be that "85 % of customers who bought *Product X* also bought *Product Y*". Discovering all such rules is important for planning marketing campaigns, designing catalogues, managing prices and stocks, customer relationships management, etc. For example, a shop may decide to place *Product A* close to *Product B* because they are often bought together, to help shoppers finish their task faster. Or the shop may place them at opposite ends of a row, and place other associated items in between to tempt people to buy those items as well, as shoppers walk from one end of the row to the other.

Overall performances of mining association rules are determined by the frequent itemset discovery [5]. Efficient algorithms for generating rules from frequent itemsets can be found in [7]. We do not consider the rule extraction sub-problem in this paper.

Although the pattern growth family of algorithms is considered as state-of-the-art technique in frequent pattern mining, its run-time performance depends on the compaction factor of the data set. For large and sparse databases the performance of the algorithm degrades significantly because it has to generate a very big number of sub-problems and merge the results returned by each sub-problem [7]. In general, the join-based methods are slower but require less memory, while the memory needs of the pattern growth or database projection based algorithms are usually very high, providing that their execution time is smaller [6].

In this paper we introduce the combinatorial number system in context of the frequent itemset mining. With the procedure based on combinatorial numbering schema it is possible to efficiently represent any k-itemset with single integer value. This is "lossless compression", meaning that it is possible to reconstruct the original itemset from the assigned value. Experiments show that memory requirements with such approach can be reduced up to 300 %. It is especially observable for the very low minimal support thresholds when the number of itemsets is the biggest. Usual k-itemset representation requires $O(k)$ integers.

Additionally, we propose Rank Join algorithm which makes several improvements on the Apriori algorithm, that is a classic join-based approach. Although, a hundreds of algorithms have been proposed on various kinds of extensions and applications, the Apriori is still the most commonly recognized reference to evaluate FIM algorithm performances. The Rank Join improves both major phases in Apriori based algorithms: *candidate generation* step and *support count and test* step. Experiments showed that, depending on particular instance of the problem, these two steps can contribute to even more than 90 % of the total execution time [8].

The paper is organized as follows. Section 2 introduces the problem of frequent itemset mining. Related work is reviewed in Section 3. Section 4 introduces combinatorial number system that is base for the new procedure for efficient representation of frequent itemsets. The procedure is named RankItemset and it is presented in Section 5. Section 6 describes a new algorithm named Rank Join, that exploits efficient itemset representation in order to improve both major phases in the Apriori based algorithms. Illustrative example is given in Section 7. Conclusion can be found in Section 8.

2 PRELIMINARIES

This section contains definitions that are necessary for further text. We primarily use notions from [1].

Suppose that $I = \{i_1, i_2, \dots, i_n\}$ is a finite set; we refer to the elements of I as *items.*

A transaction data set on I is a function $T : \{1, \dots, |T|\} \to P(I)$. The set $T(k)$ is the k^{th} transaction of T. The numbers $1, \dots, |T|$ are the transaction identifiers (TIDs).

Given a transaction data set T on the set I, we would like to determine those subsets of I that occur often enough as values of T.

Let $T : \{1, \dots, |T|\} \to P(I)$ be a transaction data set on a set of items I. The support count of a subset K of the set of items I in T is the number $\mathit{suppcount}_T(K)$ given by: $\mathit{suppcount}_T(K) = |\,\{t | 1 \le t \le |T| \wedge K \subset T(t)\}\,|$. The support of an itemset K is the number: $\mathit{support}_T(K) = \frac{\mathit{suppcount}_T(K)}{|T|}$.

An itemset K is μ-frequent relative to the transaction data set T if $\mathit{support}_T(K) \ge \mu$. We denote by F_T^μ the collection of all μ-frequent itemsets relative to the transaction data set T and by $F_{T,r}^\mu$ the collection of all μ-frequent itemsets that contain r items for $r \ge 1$.

Note that $F_T^\mu = \bigcup_{r \ge 1} F_{T,r}^\mu$.

Frequent itemset mining problem consists of finding the set F_T^μ for given minimal support μ and transaction data set T.

The following rather straightforward statement is fundamental for the study of frequent itemsets. It is known as Apriori principle.

Let $T : \{1, \dots, |T|\} \to P(I)$ be a transaction data set on a set of items I. If K and K_1 are two itemsets, then $K_1 \subset K$ implies $\mathit{support}_T(K_1) \ge \mathit{support}_T(K)$.

3 RELATED WORK

During the last two decades numerous algorithms have been proposed to mine frequent itemsets. The interest in this problem still persists [9], mainly due to high computational complexity of the task as well as because it plays an important role in many data mining applications [8].

There are several classifications and consequently several classes of frequent itemsets mining methods [9, 5, 7]. The main challenge of all these approaches is that the number of candidates and frequent itemsets is exponentially large, especially for the lower minimal support thresholds. Although the exponential search space is the fundamental problem of frequent itemset mining, it is possible to significantly speed up the support counting procedure with the use of special memory resident data structures to represent database of transactions [8].

In this section we follow the classification presented in [8, 9]. These papers are the most novel and ones of the most detailed surveys of frequent itemset mining algorithms proposed in the literature.

There are two basic classes of frequent itemset mining algorithms: join-based and tree-based (projection-based) algorithms. The common property of all these algorithms is that they extend prefixes or suffixes of itemsets to generate frequent patterns [8]. Variants of frequent itemset mining, such as maximal [15, 16, 17, 18, 19] and close [20, 21, 22, 23, 24, 25, 26, 27, 28, 29] frequent itemset mining are not considered in this paper.

The join-based algorithms generate candidate itemsets in increasing order of the itemset size, which is usually referred to as level-wise exploration [8, 5]. The candidate generation process of the earliest algorithms used joins. The original Apriori algorithm belongs to this category [8, 9]. It generates $(k+1)$-candidates from previously generated frequent k-itemsets with the use of joins. Then the algorithm computes candidate's support in the database. If the support is equal or higher than the minimum support threshold the itemset is stored as frequent $(k+1)$-itemset. It is important to emphasize that candidate itemsets are generated lexicographically, that is along with pruning irrelevant and duplicate candidates, the key issue of computational efficiency [8, 9].

Although the Apriori is presented as a join-based algorithm, it can be shown that the algorithm is a breadth first exploration of a structured arrangement of the itemsets, known as a set enumeration tree [8]. The set enumeration tree is used to model exponential search space as follows [11]. Given set of items $I = i_1, i_2, \dots, i_m$ where $i_1 \prec i_2 \prec \dots \prec i_m$, a set-enumeration is constructed from the root of the tree that represents the empty set. Then the m child nodes of the root representing m 1-itemsets are created. Additionally, for a node representing itemset $i_{j1}i_{j2}\dots i_{jk}$ and registering i_{jk}, the $(m - jk)$ child nodes representing itemsets $i_{j1}i_{j2}\dots i_{jk}i_{jk+1}$, $i_{j1}i_{j2}\dots i_{jk}i_{jk+2}$, $\dots$, $i_{j1}i_{j2}\dots i_{jk}i_{jm}$ are created.

Later classes of algorithms explicitly discuss tree-based enumeration [8, 9]. The algorithms from this class can explore the tree using breadth-first, depth-first or hybrid strategies. The breadth-first strategy allows level-wise pruning according to the Apriori principle, that is not possible in the second strategy. The depth-first version allows better memory management because one only needs to maintain a small number of projected transaction sets along the depth of the tree. It is especially desirable to efficiently discover maximal patterns.

The enumeration tree originally is defined on the prefixes of the itemsets. Later algorithms such as FP-Growth find all frequent itemsets ending with a particular suffix by recursively employing a divide-and-conquer strategy to split the problem into smaller sub-problems [5, 7]. It uses the compact data structure called an FP-Tree to build a compressed representation of the input data. The tree is constructed by mapping each transaction onto a path in the FP-Tree. As different transactions can have numerous items in common, their paths may overlap. The compression achieved by using the FP-Tree structure is based on paths' overlaps. If the FP-Tree can be stored in main memory, this will allow to extract frequent itemsets directly from the tree, without additionally passes over the database [7].

Frequent itemsets are generated from the FP-Tree by exploring the tree in a bottom-up fashion. This strategy finds frequent itemsets ending with a particular frequent item and is usually referred to as suffix-based approach [8, 9, 5, 7]. Since every transaction is mapped onto a path in the tree, we can derive the frequent itemsets ending with a particular item, by examining only the paths containing that item – prefix paths. These paths can be accessed efficiently using pointers associated with each node in the tree. Prefix paths are converted into a conditional FP-Tree, which is structurally similar to an FP-Tree, except it is used to find frequent itemsets ending with a particular suffix. FP-Growth uses the conditional FP-Tree for frequent item i to solve the sub-problems of finding frequent itemsets ending in xi where $x \in I \setminus i$.

The size of an FP-Tree is typically smaller than the size of the uncompressed data because it is expected that many transactions often share several items. The best-case scenario is when all transactions have the same set of items and the corresponding FP-Tree contains only a single branch. But, when datasets are sparse, the pattern growth algorithms are inefficient [7, 10]. The worst-case scenario happens when every transaction has a unique set of items. In that case, the size of FP-Tree is higher than the size of the original data [7].

The size of an FP-Tree also depends on how the items are ordered. The usual heuristic is to sort frequent items in decreasing support counts. Nevertheless, ordering by decreasing support counts does not always lead to the smallest tree [7].

Although, the pattern growth family of algorithms is considered as state-of-the-art technique in frequent pattern mining [8], construction of the FP-Tree becomes challenging both from runtime and space complexity as the database grows larger [7, 10]. It is mainly due to the fact that the algorithm has to generate a large number of sub-problems and merge the results returned by each sub-problem. There have been many works that deal with the previous challenges.

In recent years, several data structures, named Node-list [13], N-list [12], Nodeset [11] and DiffNodeset [10] have been presented to enhance the efficiency of mining frequent itemset. They are all based on node sets originated from a prefix tree with encoded nodes. The prefix tree employed by Node-list and N-list uses pre-order number and post-order number to encode each node. The only difference between Node-list and N-list is that Node-list uses descendant nodes to represent an itemset while N-list represents an itemset by ancestor nodes. The Nodeset requires only the pre-order (or post-order) number of a node to encode the node. DiffNodeset only keeps track of differences in the Nodesets of a candidate itemset from its generating frequent itemsets. Compared with the Nodeset, the cardinality of the DiffNodeset is much smaller. Inspired by these data structures several algorithms have been designed. Extensive experimental studies have shown that these algorithms are very effective and usually outperform previous algorithms (FP-Growth∗, Eclat_g) [10].

4 COMBINATORIAL NUMBER SYSTEM

In this section we establish a novel theoretical framework for the frequent itemset mining problem.

Let us review the combinatorial number system and introduce one-to-one correspondence between the integers $1, 2, \ldots, \binom{n}{m}$ and the set of m-combinations of $\{1, 2, \ldots, n\}$ listed in lexicographic order. We primarily use results from [3].

Let $(c_1, c_2, \ldots, c_m)$ represent one such combination where $c_1 < c_2 < \ldots < c_m$. We define

$$complement(n, c_1, c_2, \ldots, c_m) = (d_1, d_2, \ldots, d_m) \tag{1}$$

as the *complement* of $(c_1, c_2, \ldots, c_m)$ with respect to $\{1, 2, \ldots, n\}$, where

$$d_i \leftarrow (n + 1) - c_{m-i+1}. \tag{2}$$

The following function takes n and $(c_1, c_2, \ldots, c_m)$ as input and returns $(d_1, d_2, \ldots, d_m)$ as output in $O(m)$ time [3].

```
**function** COMPLEMENT($n, c_1, c_2, \ldots, c_m$)
Step 1: **for** $i = 1$ **to** $m$ **do**
    $d_i \leftarrow (n + 1) - c_{m-i+1}$
**end for**.
Step 2: COMPLEMENT $\leftarrow (d_1, d_2, \ldots, d_m)$
```

Now, let the *reverse* of $(c_1, c_2, \ldots, c_m)$ be given by $(c_m, c_{m-1}, \ldots, c_1)$. The mapping

$$order(c_1, c_2, \ldots, c_m) = \Sigma_{i=1}^{m} C_i^{c_i - 1} \tag{3}$$

has the following properties [3]:

1. if $(c_1, c_2, \ldots, c_m)$ and $(c_1', c_2', \ldots, c_m')$ are two m-combinations and the reverse of $(c_1, c_2, \ldots, c_m)$ precedes the reverse of $(c_1', c_2', \ldots, c_m')$ in lexicographic order, then
$$order(c_1, c_2, \ldots, c_m) < order(c_1', c_2', \ldots, c_m'); \tag{4}$$

2. $order(1, 2, \ldots, m) = 0$ and $order(((n - m + 1)(n - m + 2) \ldots n)) = \binom{n}{m} - 1$ implying that the transformation *order* maps the $\binom{n}{m}$ different m-combinations onto $\{0, 1, \ldots, \binom{n}{m} - 1\}$ while preserving reverse lexicographic order.

The following function takes $(c_1, c_2, \ldots, c_m)$ as input and returns $order(c_1, c_2, \ldots, c_m)$ as output in $O(m^2)$ time [3].

```
function ORDER($c_1, c_2, \ldots, c_m$)
Step 1: $sum \leftarrow 0$
Step 2: for $i = 1$ to $m$ do
            $sum \leftarrow sum + \binom{c_i - 1}{i}$
        end for.
Step 3: ORDER $\leftarrow sum$
```

Using order and complement, we can define the following one-to-one mapping of the set of $\binom{n}{m}$ possible combinations onto $\{1, 2, \ldots, \binom{n}{m}\}$ which preserves lexicographic ordering:

$$rankc(n, c_1, c_2, \ldots, c_m) = \binom{n}{m} - order(complement(n, c_1, c_2, \ldots, c_m)).$$

Thus $rankc(n, 1, 2, \ldots, m) = 1$, $rankc(n, 1, 2, \ldots, m, m+1) = 2$, $\ldots$, $rankc(n, (n-m+1), (n-m+2), \ldots n) = \binom{n}{m}$. The following procedure is an implementation of the preceding mapping: it takes n and the combinations $(c_1, c_2, \ldots, c_m)$ as input and returns the ordinal position h of the later in $O(m^2)$ time [3].

```
procedure RANKC($n, c_1, c_2, \ldots, c_m, h$)
Step 1: $h \leftarrow C_m^n$
Step 2: $(d_1, d_2, \ldots, d_m) \leftarrow COMPLEMENT(n, c_1, c_2, \ldots, c_m)$
Step 3: $h \leftarrow h - ORDER(d_1, d_2, \ldots, d_m)$
```

Let us explain the complexity of the RANKC in more details. In step 2 it calls function COMPLEMENT with $O(m)$ complexity. But, the dominant step is 3rd in which function ORDER is called. Complexity for ORDER is $O(m^2)$ as we stated earlier. Actually, in the second step in ORDER function there is loop m iterations long, where in each iteration $\binom{c_i - 1}{i}$ is calculated, $1 \le i \le m$. Upper bound complexity for each iteration is $O(m)$, so overall complexity for ORDER is $O(m^2)$.

Here we will just mention the question of inverting the RANKC mapping. Specifically, given an integer h, where $1 \le h \le \binom{n}{k}$, it is required to determine the combination $(c_1 c_2 \ldots c_k)$ such that $rankc(n, c_1, c_2, \ldots, c_k) = h$. The function ORDERINV$(n, k, g)$ [3] is an implementation of the mentioned mapping. It takes n, k and h as input and returns a combination $(c_1 c_2 \ldots c_k)$ as output in $O(nk)$ time.

In the end of this section we propose hypothetical algorithm – Apriori Combinatorial for mining frequent itemsets in previously defined framework.

We use two dimensional array *frequentItemsets* to store support counts for k-combinations that appear in database. In the iteration k all frequent k-combinations will be generated. Instead of explicit candidate generation, our algorithm uses combinatorial number schema to map k-combinations to indexes of the array *frequentItemsest*$[k-1]$. This mapping preserves lexicographical ordering, so combination $(1, 2, \ldots, k)$ is mapped to 0 and appropriate support is stored in *frequentItemsets*$[k-1][0]$. Similarly, combination $c_1c_2, \ldots, c_k$ is mapped to $r = RANKC(n, c_1, c_2, \ldots, c_k)$ while corresponding support count is stored in *frequentItemsets*$[k-1][r]$.

In the support count phase in the iteration k, each transaction is read, and the support for all k-combinations contained in the transaction is incremented.

Let us compare complexity of the Apriori Combinatorial to the original Apriori algorithm. We have to compare complexities of the two main steps: candidate generation step and support count step.

In the Apriori Combinatorial algorithm, there is no candidate generation step. Actually, in the iteration k, it is sufficient to allocate $\binom{n}{k}$ integers to the array *frequentItemsets*$[k-1]$ and initialize them to 0.

For support counting, in each iteration the algorithm reads all transactions from the database. The algorithm computes $RANKC(s), s \subset t, |s| = k$ and increments *frequentItemsets*$[k-1][RANKC(s)]$. As it will be explained in the next section, by using Pascal's triangle it is possible to reduce complexity for $RANKC$ to $O(k)$. Consequently, the complexity for support counting phase in the Apriori Combinatorial is $O\left(|T|\Sigma_k k\binom{t_{max}}{k}\right)$, where t_{max} is the size of the longest transaction and $|T|$ is the number of transactions in the database. Notice that factor α_k that appears in complexity estimation for the original Apriori is eliminated.

The previous considerations are summarized in Table 1.

	Apriori-Like Algorithms	Apriori Combinatorial
candidate generation step	$\Sigma_{k=2}^{w}(k-1)\lvert F_{k-1}\rvert^2$	$O(1)$
creating hash tree	$\Sigma_{k=2}^{w}k\lvert C_k\rvert$	0
candidate prunning	$\Sigma_{k=2}^{w}k^2\lvert C_k\rvert$	0
support counting	$O\left(\lvert T\rvert\Sigma_k\alpha_k\binom{t_{max}}{k}\right)$	$O\left(\lvert T\rvert\Sigma_k k\binom{t_{max}}{k}\right)$

Table 1.

As we mentioned earlier, the Apriori Combinatorial is a hypothetical algorithm, because it is impossible to manage the array *frequentItemsets* in cases where memory capacity is limited, as it is expected to be. But still, it is possible to implement the ideas from the Apriori Combinatorial, and significantly improve both major steps in a join-based algorithms. In the next section we propose novel approach for efficient itemset representation. In Section 6 we present Rank Join algorithm that outperforms other Apriori-like algorithms by efficient implementation of the candidate generation and support count steps.

5 EFFICIENT ITEMSET REPRESENTATION

In this section we give a short introduction to the join-based approach and review the Apriori algorithm from [4]. The Apriori is famous and widely-used algorithm for the frequent itemset mining.

The Apriori algorithm iteratively generates frequent itemsets starting from frequent 1-itemsets to frequent itemsets of the maximal size. Each iteration of the Apriori consists of two phases: *candidate generation* and *support count and test.*

In the candidate generation phase potentially frequent k-itemsets or candidate k-itemsets are generated. The anti-monotone property of the itemset support is used in this phase and it provides elimination or pruning of some candidates without calculating its actual support (candidate containing at least one not frequent subset is pruned immediately, before support counting phase according to the Apriori principle).

The set C_k, that contains candidate k-itemsets, is generated from F_{k-1}. The set F_{k-1} is known from the previous iteration and contains frequent $(k-1)$-itemsets. The two $(k-1)$-frequent itemsets $c_1c_2\dots c_{k-1}$ and $d_1d_2\dots d_{k-1}$ give candidate k-itemset e if and only if $c_1 = d_1 \wedge c_2 = d_2 \wedge \dots \wedge c_{k-2} = d_{k-2}$ and $c_{k-1} < d_{k-1}$. In that case, we have $e = c_1c_2\dots c_{k-1}d_{k-1}$.

The support count and test phase consists of calculating support counts for all previously generated candidates (the set C_k) and tests the counts to the minimal support threshold. In this phase, it is essential to efficiently determine if the candidates are contained in a particular transaction in order to increment their support. Candidates that have sufficiently large support count, are termed as frequent itemsets and they are stored as elements of the F_k.

The Apriori algorithm terminates when none of the frequent itemsets can be generated.

We will use $C_{T,k}$ to denote candidate k-itemsets which are generated in the iteration k over the transactional database T. By $F^{\mu}_{T,k}$ we denote k-frequent itemsets in the database T, having support count $\geq \mu$, where μ is minimal support threshold. If T and μ are known from the context, we will omit them. Pseudo-code for the Apriori algorithm comes next.

```
Algorithm:    Apriori
Input:  A transaction dataset $T$;
Input:  Minimal support threshold $\mu$
Output:  $F^{\mu}_{T}$
Method:
    1.  $C_{T,1} = \{\{i\} | i \in I\}$
    2.  $F^{\mu}_{T,1} = \{K \in C_{T,1} | supp_T(K) \geq \mu\}$
    3.  for (k=2; $F^{\mu}_{T,k-1} \neq \emptyset$; k++)
           $C_{T,k} = \text{apriori\_gen}(F^{\mu}_{T,k-1})$
           $F^{\mu}_{T,k} = \{K \in C_{T,k} | supp_T \geq \mu\}$
    end for
4.  $F^{\mu}_{T} = \bigcup_{r\geq 1} F^{\mu}_{T,r}$
```

Let us now introduce the procedure that will reduce memory requirements for storing the set of all μ-frequent k-itemsets $F^{\mu}_{T,k}$ in each iteration. Consequently, the size of the final set $F^{\mu}_{T} = \bigcup_{r\geq 1} F^{\mu}_{T,r}$ that contains all μ-frequent itemsets in the given database is reduced.

The idea is to represent each k-frequent itemset with just one integer. The following procedure named *RankItemset* is an implementation of the mapping: it takes k-itemset $(c_1, c_2, \ldots, c_k)$ as an input and returns the ordinal position h of the later. The procedure is based on results from [3], here reviewed in the previous section.

```
**procedure** RankItemset$(c_1, c_2, \ldots, c_k, h)$
Step 1: $sum \leftarrow 0$
Step 2: **for** $i = 1$ **to** $k$ **do**
            $sum \leftarrow sum + \binom{n-c_{k-i+1}}{i}$
        **end for**
Step 3: $h \leftarrow C^n_k - sum$
```

The *RankItemset* preserves lexicographic ordering. Thus *RankItemset*$(1, 2, \ldots, k) = 1$, *RankItemset*$(1, 2, \ldots, k+1) = 2, \ldots,$ *RankItemset*$((n-k+1), (n-k+2), \ldots n) = \binom{n}{k}$, where n is the number of items.

Let us estimate the complexity of the *RankItemset*. The dominant step is the second one in which *sum* is calculated. There is a loop k iterations long, where in each iteration $\binom{n-c_{k-i+1}}{i}$ is calculated, $1 \leq i \leq k$. Upper bound complexity for each iteration is $O(k)$, so the overall complexity is $O(k^2)$.

It is possible to improve the *RankItemset* if we compute all $\binom{n-c_{k-i+1}}{i}$ in advance. We have $1 \leq c_i \leq n$ and $1 \leq i \leq k$, so we can generate Pascal's Triangle with n rows and k columns. The Pascal's triangle can be stored as lower triangle matrix, where element $(i, j), j < i$ is $\binom{i}{j}$.

Using the Pascal's Triangle we reduce complexity to $O(k)$; there is still a loop k iterations long, where in each iteration $\binom{n-c_{k-i+1}}{i}$ is calculated, $1 \leq i \leq k$, but now in $O(1)$ time. The previous implies that the overall complexity of the *RankItemset* is reduced to $O(k)$. Similarly, the complexity of the *RANKC* from the previous section can be reduced to $O(k)$, as we stated earlier.

The RankItemset procedure allows to change the step $F^{\mu}_{T,k} = \{K \in C_{T,k} | supp_T \geq \mu\}$ with the following $F^{\mu}_{T,k} = \{RankItemset(K, h_K) | K \in C_{T,k} \wedge supp_T(K) \geq \mu\}$. In other words, instead of storing each frequent itemset explicitly, we store corresponding *RankItemset* value. In that way we reduce space requirements for k-itemset storing to $O(1)$ instead of $O(k)$. At the same time, the previous procedure does not degrade time complexity because the *RankItemset* is $O(k)$ as well as making a copy of k-itemset and storing it in $F^{\mu}_{T,k}$.

Here we will just mention that the function ORDERINV(n, k, g) [3] is an implementation of inverting the *RankItemset*. It takes n, k and h as an input and returns an itemset $(c_1 c_2 \ldots c_k)$ as an output in $O(nk)$ time. In other words, the proposed representation of the itemsets does not influence the final result, i.e., we always obtain the same set of the frequent patterns.

It is possible to integrate the *RankItemset* procedure into the most efficient pattern-growth algorithm [14, 13] and improve their space efficiency. Additionally, candidate ranking can be used to encode nodes in the set enumeration tree instead of pre-order or/and post-order number. Node encoding can be done along with the tree construction procedure, providing that several pre-order or/and post-order tree traversals are not necessary any more.

6 RANK JOIN ALGORITHM

In this section we propose the Rank Join algorithm for frequent itemset mining. It implements ideas from the previous sections and improves, at the first place, the *candidate generation* procedure, but also the *support count and test* step.

The Rank Join maintains one-dimensional array *rank*, starting from $k = 2^{\text{nd}}$ iteration. At the index r of the array, the algorithm stores the *RankItemset* of the r^{th} candidate in the lexicographic order, so $\textit{rank}[r] = \textit{RankItemset}(c_1^r, c_2^r, \ldots, c_k^r)$, where $c_1^r c_2^r \ldots c_k^r$ is the r^{th} candidate in lexicographic order in the iteration k.

The array *rank* is used for efficient implementation of "joining" procedure in the candidate generation phase as follows.

As it is described in the previous section, in the iteration k, a join-based algorithm generates candidate k-itemsets from frequent $(k-1)$-itemsets (known from the previous iteration). More precisely, two $(k-1)$-frequent itemsets $c_1 c_2 \ldots c_{k-1}$ and $d_1 d_2 \ldots d_{k-1}$ give candidate k-itemset e if and only if $c_1 = d_1 \wedge c_2 = d_2 \wedge \ldots \wedge c_{k-2} = d_{k-2}$ and $c_{k-1} < d_{k-1}$. In that case, we have $e = c_1 c_2 \ldots c_{k-1} d_{k-1}$. So, each "joining" is of $O(k)$ complexity.

Having in hand the *rank* array, the Rank Join will join $c_1 c_2 \ldots c_{k-1}$ and $d_1 d_2 \ldots d_{k-1}$ in $O(1)$. Actually, let $c_1 c_2 \ldots c_{k-1}$ be the r^{th} and let $d_1 d_2 \ldots d_{k-1}$ be the p^{th} frequent $(k-1)$-itemset in lexicographic order. So, in order to check joining condition it is sufficient to test if $\textit{rank}[r] = \textit{rank}[p]$ and $c_{k-1} < d_{k-1}$, which is $O(1)$ operation.

We emphasise here, that the Rank Join requires $O(k)$ time to calculate a rank for one candidate k-itemset. It is compensated in the general join-base algorithm by the step where candidate k-itemset is inserted in a tree structure in $O(k)$ time. The Rank Join does not create tree for storing candidates. The role of the tree in the support counting phase is given to the *rank* array, as will be described later.

Finally, we can conclude that the candidate generation step in the iteration k in a general join-base algorithm is $O((k-1)|F_{k-1}|^2)$, while in the Rank Join it is $O(|F_{k-1}|^2)$.

The *rank* array is used in the support count phase in order to reduce number of candidate itemsets that are compared against transactions from the database. Let us explain the idea in case of the iteration k. All candidates are lexicographically generated and stored in the array C_k. Also, for each candidate $C_k[i]$ holds $\mathit{rank}[i] = \mathit{RankItemset}(C_k[i])$. In order to find support counts for candidate itemsets it is necessary, for each transaction $t \in T$, to enumerate those candidates that are contained in t and consequently increment their support.

Consider transaction $t = t_1 t_2 \dots t_l$. If $l < k$ there is no k-candidate that is contained in t. If $l \geq k$, the lexicographically smallest k-itemset contained in t is $t_1 t_2 \dots t_k$. At the same time, the lexicographically largest k-itemset contained in t is $t_{l-k+1} \dots t_{l-1} t_l$. So, candidates possibly contained in t are $C_k[i]$, where

$\mathit{RankItemset}(t_1, t_2, \dots, t_k) \leq \mathit{rank}[i] \leq \mathit{RankItemset}(t_{l-k+1}, \dots, t_{l-1}, t_l)$.

Because of lexicographical ordering they are placed consecutively. The idea is illustrated in Figure 1. Notice that the *rank* array does not contain NULL pointers.

The previous procedure is similar but more efficient than the one implemented in algorithms from [15] and [14], that is the most efficient join-based algorithms [8]. The later exploits an array $\mathit{PREFIX}_k[]$ according to the following consideration. The various k-itemsets of C_k are stored in a vector lexicographically, and all of them sharing a common 2-item prefix are stored in a continuous section of this vector. Each entry in $\mathit{PREFIX}_k[]$ represents different 2-item prefix, and contains the pointer to the first candidate in C_k characterized by that prefix. When transaction t is processed, the algorithm generates all the possible prefixes of length 2. Once a prefix $\{t_{i1}, t_{i2}\}$ is selected, the possible completions of this prefix needed to build k-subsets of t can only be found in $\{t_{i2+1}, \dots, t_{|t|}\}$. The previous is illustrated in Figure 1.

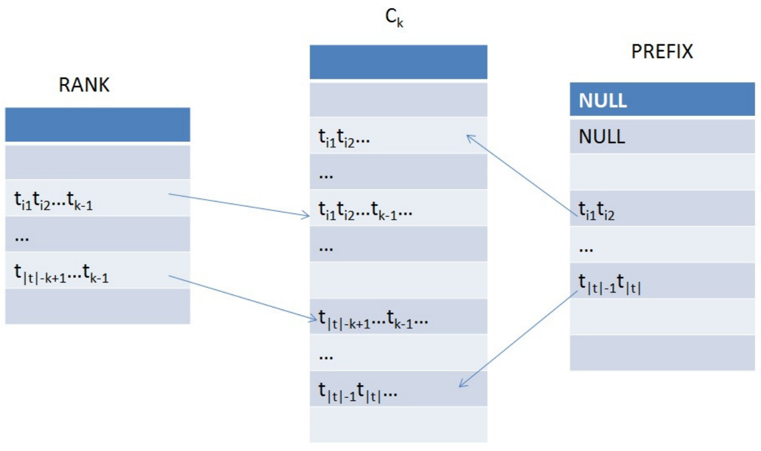


Figure 1. Locating candidates using RANK and PREFIX [15, 14] arrays

The Rank Join is formalized in the following listing.

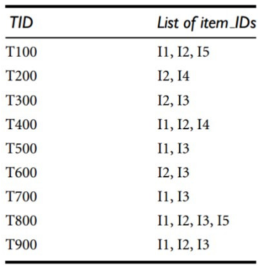
```
Algorithm:    Rank Join
Input:  A transaction dataset $T$; Minimal support threshold $\mu$
Output:  $F_T^\mu$
Method:
    1.  $C_{T,1} = \{\{i\} | i \in I\}$
    2.  $F_{T,1}^\mu = \{K \in C_{T,1} | supp_T(K) \geq \mu\}$
        $rank[i] = i, 1 \leq i \leq |F_{T,1}^\mu|$
    3.  for (k=2; $F_{T,k-1}^\mu \neq \emptyset$; k++)
           $C_{T,k} = \{e_1, e_2, \dots, e_k | \exists c^r, d^p \in F_{T,k-1}^\mu,$
           $c^r = e_1 e_2 \dots e_{k-2} e_{k-1} \wedge d^p = e_1 e_2 \dots e_{k-2} e_k$
           $\wedge\, rank[r] = rank[p]\}$
           for each $t \in T$ do
                $start = RankItemset(t_1, t_2, \dots, t_k)$
                $end = RankItemset(t_{l-k+1}, , \dots, t_{k-1}, t_k)$
                for each $C_k[i], start \leq i \leq end$ do
                         if $C_k \subset t$ $supp_T(C_k[i]) + +$
                     end for each
           end for each
           $F_{T,k}^\mu = \{K \in C_{T,k} | supp_T(K) \geq \mu\}$
        end for
    4.  $F_T^\mu = \bigcup_{r \geq 1} F_{T,r}^\mu$
```

7 EXAMPLE

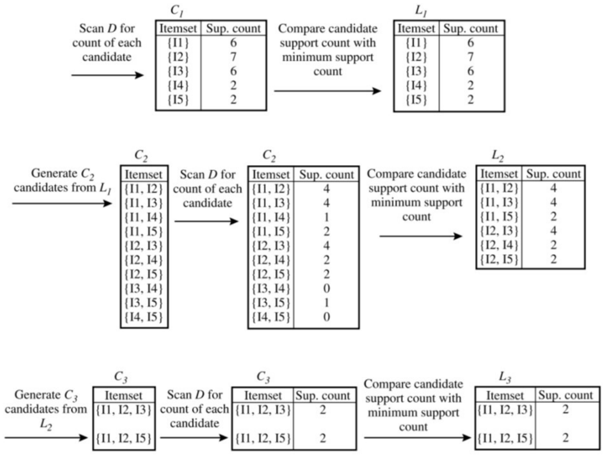
In this section, we will illustrate main benefits of the itemset representation with combinatorial number system. Consider transaction dataset T from Figure 2. It contains nine transactions. Generation of the candidate itemsets and frequent itemsets in the original Apriori algorithm, where the minimum support count is 2 transactions, is also illustrated in Figure 2 [5].

There are $6 + 6 + 2$ frequent itemsets. In the original Apriori method, k-itemset is represented with k integers. It means, that the Apriori needs $6*1+6*2+2*3 = 24$ integers to store all frequent itemsets.

With the *RankItemset* procedure from Section 5, we can uniquely represent each frequent itemset with just one integer. It takes k-itemset $(c_1, c_2, \dots, c_k)$ as an input and returns the ordinal position h of the later. For example, the itemset $(1, 2)$ is represented with 1, because it is lexicographically the first 2-combination of $\{1, 2, 3, 4, 5\}$. Similarly, the itemset $(1, 5)$ is represented with 4, while 3 corresponds to the itemset $(1, 2, 5)$ as it is the third 3-combination of $\{1, 2, 3, 4, 5\}$. In the end, we have $6 + 6 + 2 = 14$ frequent itemsets and we use 14 integers to store them. Again, we emphasize that the proposed representation of the frequent itemsets does not influence the final result, i.e., we always obtain the same set of frequent patterns but in the most compact form. It is guaranteed by the combinatorial number system



TID	*List of item_IDs*
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3


Scan *D* for count of each candidate
Compare candidate support count with minimum support count
Generate C_2 candidates from L_1
Scan *D* for count of each candidate
Compare candidate support count with minimum support count
Generate C_3 candidates from L_2
Scan *D* for count of each candidate
Compare candidate support count with minimum support count

C_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

L_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

C_2

Itemset
{I1, I2}
{I1, I3}
{I1, I4}
{I1, I5}
{I2, I3}
{I2, I4}
{I2, I5}
{I3, I4}
{I3, I5}
{I4, I5}

C_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I4}	1
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2
{I3, I4}	0
{I3, I5}	1
{I4, I5}	0

L_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2

C_3

Itemset
{I1, I2, I3}
{I1, I2, I5}

C_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

L_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Figure 2. Generation of candidate itemsets and frequent itemsets

that introduces one-to-one correspondence between the integers $1, 2, \dots, \binom{n}{m}$ and the set of m-combinations of $\{1, 2, \dots, n\}$ listed in lexicographic order.

Finally, we explain how two candidates are joined in the Rank Join in $O(1)$ having in hand the *rank* array. Consider the itemset $(1, 2, 3, 5)$, that will be pruned by the Apriori principle because it contains the subsets that are not frequent (because of that it does not appear in Figure 2). It is generated by joining $(1, 2, 3)$ and $(1, 2, 5)$. In the original Apriori this joining requires three comparisons: $1 = 1$, $2 = 2$ and $3 < 5$. The Rank Join from Section 6, needs just two comparisons $rank(1, 2) = rank(1, 2)$ and $3 < 5$. In general, in the original Apriori algorithm joining of two k-itemsets requests k comparisons, comparing to just 2 in the Rank Join.

8 PERFORMANCE EVALUATION

In order to prove theoretical considerations from Sections 5 and 6 we performed a number of experiments.

Datasets used in experiments are synthetic datasets generated with IBM Quest Data Generator. At the command-line we run *seq_data_generator lit -ascii -ntrans XX -tlen YY -nitems ZZ -fname TXXLYYNZZ*. It will produce file TXXLYYNZZ with XX $\times$ 1 000 transactions involving YY average number of items per transaction, drawn from ZZ $\times$ 1 000 total number of items. Each line of the file is a transaction. The items in each transaction are represented by item numbers and are separated by spaces. For example, file T1000L10N1 contains 1 000 000 transactions made from 1 000 items, with average length of 10.

We measured memory needed for storing frequent itemsets in KB with respect to *minsup* parameter. Achieved improvements are up to 300 %. We emphasize the fact that the most significant enhancements are achieved for the smallest values of *minsup*, when the number of frequent itemsets is the biggest. Results are presented in Figures 3 and 4.

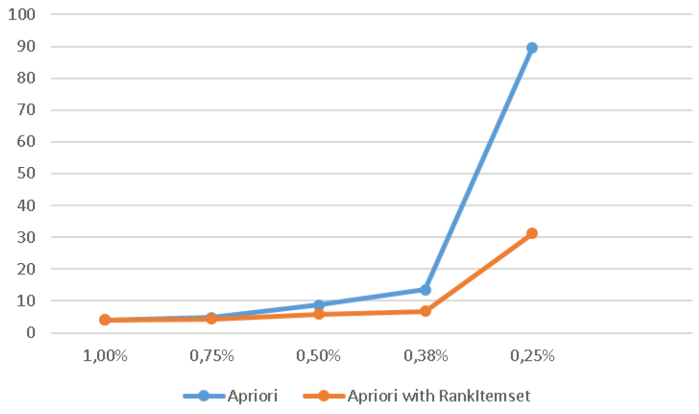


Figure 3. Experiment 1, datasets T200L10N1

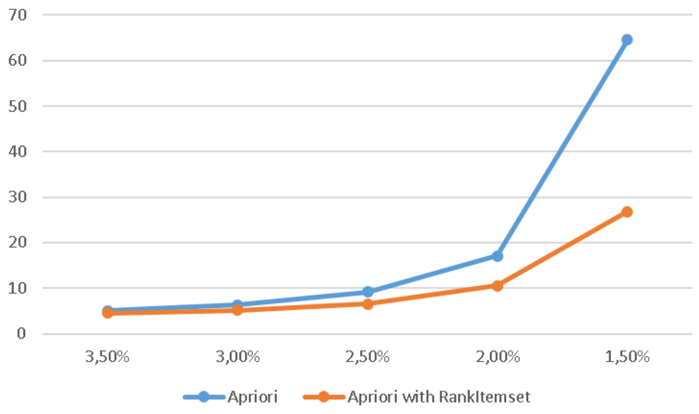


Figure 4. Experiment 1, datasets T100L40N1

We implemented the Rank Join and the original Apriori algorithm with direct count procedure [9], that is the most efficient join-based algorithm [8]. We used programming language C and machine with Intel Celeron 2 GHz and 2 GB of RAM. Datasets used in experiments are synthetic datasets generated with IBM Quest Data Generator as it is already explained.

We measured execution time in seconds with respect to *minsup* parameter. Achieved improvements are between 3 % and 15 %. We emphasize the fact that

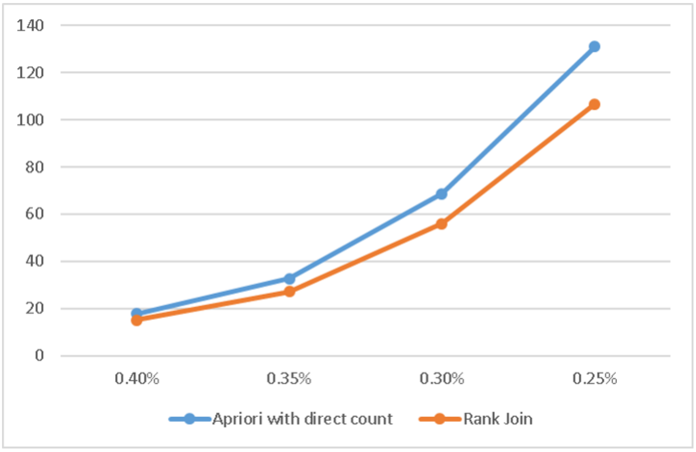


Figure 5. Experiment 2, datasets T100L10N1

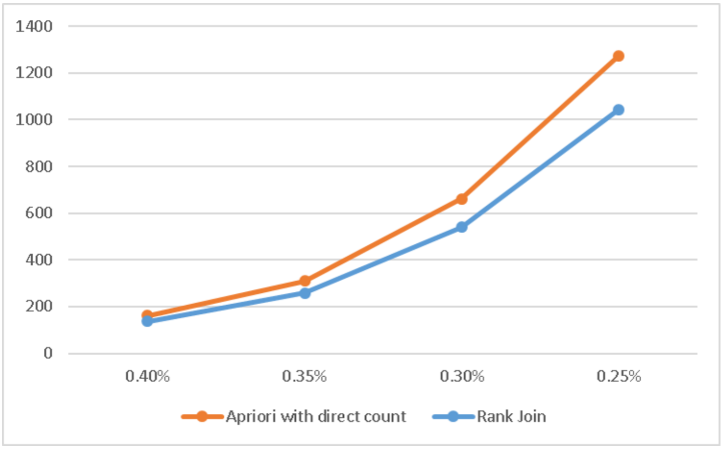


Figure 6. Experiment 2, dataset T1000L10N1

the most significant enhancements are achieved for the smallest values of *minsup*, when the number of candidate itemsets is the biggest. Results of the experiments are presented in Figures 5 and 6 for synthetic datasets and in Figure 7 for Extended BAKERY dataset. The dataset contains information about one year worth of sales information for a couple of small bakery shops. The sales are made by employees. The dataset contains information about the different store locations in West Coast states (California, Oregon, Arizona, Nevada), the assortments of baked goods offered for sale and the purchases made [30]. The experiment shows that the proposed algorithm outperforms the original Apriori on the real dataset, too.

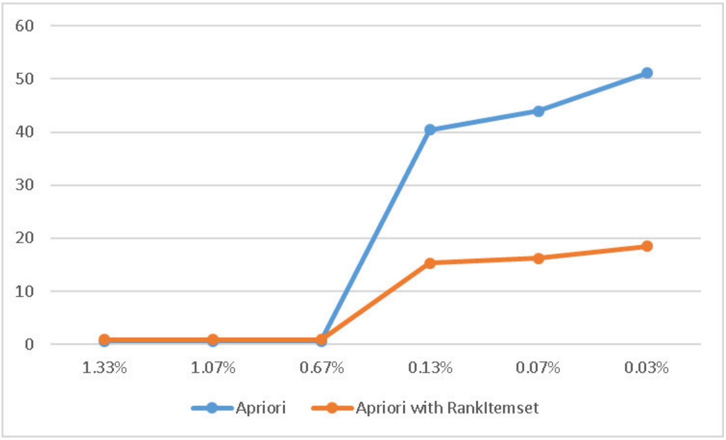


Figure 7. Experiment 2, BAKERY dataset

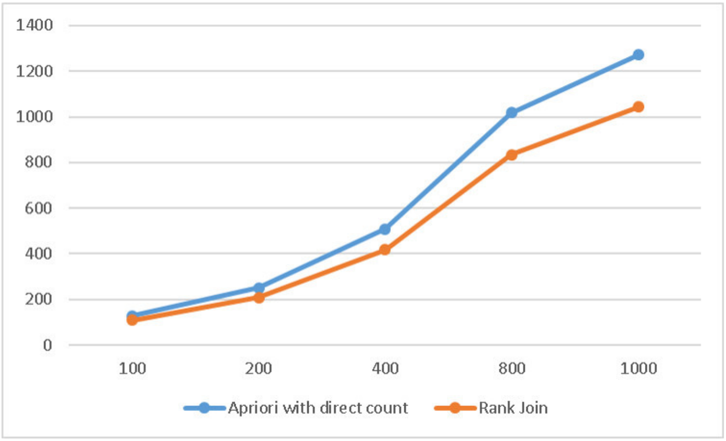


Figure 8. Experiment 3, scalability

In all experiments, the parameter *minsup* is changed in a way that indicates performance differences the best. We started with the greatest *minsup* value for which there was a significant difference in the algorithms' performances and reduced it to the smallest value for which the original Apriori method can finish without "out of memory error".

In Figure 8 we present results of experiment in which Rank Join shows better scalability. We set $minsup = 0.25\,\%$ and change number of transactions from 100 K to 1 000 K.

9 CONCLUSIONS AND FUTURE WORK

In this paper we have proposed a novel procedure for efficient representation of k-itemset with just one integer value. In this segment it is superior by comparison with any other approach. An itemset c is represented with the $RankItemset(c)$ value.

Additionally, we presented new join-based approach called Rank Join. The Rank Join significantly improves both major steps in join-base algorithms. We performed a series of experiments and measured execution times with respect to *minsup* parameter. In all test cases Rank Join is more efficient than any other join-base algorithm. This is especially case for very small *minsup* values when these algorithms generate the biggest number of candidate itemsets.

We believe that candidate ranking by combinatorial number system can be effectively integrated into pattern-growth algorithms, that are state-of-the-art in frequent itemset mining, and additionally improve their performances. As future work, we plan to integrate candidate ranking into the most efficient pattern-growth algorithm [14, 13]. It can improve their efficiency, because candidate ranking can be used to encode nodes in the set enumeration tree instead of pre-order or/and post-order number. Node encoding can be done along with the tree construction procedure, providing that several pre-order or/and post-order tree traversals needed in [10, 11, 12, 13], are not necessary.

REFERENCES

[1] Simovici, D. A.—Djeraba, C.: Mathematical Tools for Data Mining – Set Theory, Partial Orders, Combinatorics. Springer, London, 2008.

[2] Maurer, B. S.—Ralston, A.: Discrete Algorithmic Mathematics. A. K. Peters, Massachusetts, 1998.

[3] Akl, G. S.: The Design and Analysis of Parallel Algorithms. Prentice Hall, New Jersey, 1989.

[4] Agrawal, R.—Imielinski, T.—Swami, A. N.: Mining Association Rules Between Sets of Items in Large Databases. Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data (SIGMOD '93), Washington DC, USA, 1993, pp. 207–216, doi: 10.1145/170035.170072.

[5] Han, J.—Cheng, H.—Xin, D.—Yan, X.: Frequent Pattern Mining: Current Status and Future Directions. Data Mining and Knowledge Discovery, Springer, Vol. 15, 2007, No. 1, pp. 55–86, doi: 10.1007/s10618-006-0059-1.

[6] Ivancsy, R.—Vajk, I.: Automata Theory Approach for Solving Frequent Pattern Discovery Problem. International Journal of Computer, Control, Quantum and Information Engineering, 2007, pp. 203–208.

[7] Tan, P. N.—Steinbach, M.—Kumar, V.: Introduction to Data Mining. Springer, London, 2006.

[8] Aggarwal, C. C.—Bhuiayn, M. A.—Mohammad, H. A.: Frequent Pattern Mining Algorithms: A Survey. In: Aggarwal, C., Han, J. (Eds.): Frequent Pattern Mining. Springer International Publishing Switzerland, 2014, pp. 19–64.

[9] Aggarwal, C. C.: An Introduction to Frequent Pattern Mining. In: Aggarwal, C., Han, J. (Eds.): Frequent Pattern Mining. Springer International Publishing Switzerland, 2014, pp. 1–17, doi: 10.1007/978-3-319-07821-2_1.

[10] Deng, Z. H.: DiffNodesets: An Efficient Structure for Fast Mining Frequent Itemsets. Applied Soft Computing, Vol. 41, 2016, pp. 214–223, doi: 10.1016/j.asoc.2016.01.010.

[11] Deng, Z. H.—Lv, S. H.: PrePost+: An Efficient N-Lists-Based Algorithm for Mining Frequent Itemsets via Children-Parent Equivalence Pruning. Expert Systems with Applications, Vol. 42, 2015, No. 13, pp. 5424–5432.

[12] Deng, Z. H.—Wang, Z. H.—Jiang, J. J.: A New Algorithm for Fast Mining Frequent Itemsets Using N-Lists. Science China Information Sciences, Vol. 55, 2012, No. 9, pp. 2008–2030.

[13] Deng, Z. H.—Wang, Z. H.: PrePost+: A New Fast Vertical Method for Mining Frequent Itemsets. International Journal of Computational Intelligence Systems, Vol. 3, 2010, No. 6, pp. 733–744.

[14] Orlando, S.—Palmerini, P.—Perego, R.: Enhancing the Apriori Algorithm for Frequent Set Counting. In: Kambayashi, Y., Winiwarter, W., Arikawa, M. (Eds.): Data Warehousing and Knowledge Discovery (DaWaK 2001). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2114, 2001, pp. 71–82.

[15] Orlando, S.—Palmerini, P.—Perego, R.—Silvestri, F.: Adaptive and Resource-Aware Mining of Frequent Sets. Proceedings of the ACM ICDM Conference on Data Mining, 2002, doi: 10.1109/ICDM.2002.1183921.

[16] Bayardo Jr., R. J.: Efficiently Mining Long Patterns from Databases. ACM SIGMOD Conference, 1998, doi: 10.1145/276304.276313.

[17] Agarwal, R.—Aggarwal, C. C.—Prasad, V. V. V.: Depth First Generation of Long Patterns. Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '00), 2000, pp. 108–118, doi: 10.1145/347090.347114.

[18] Agarwal, R.—Aggarwal, C. C.—Prasad, V. V. V.: A Tree Projection Algorithm for Generation of Frequent Itemsets. Journal of Parallel and Distributed Computing, Vol. 61, 2001, No. 3, pp. 350–371.

[19] Burdick, D.—Calimlim, M.—Gehrke, J.: MAFIA: A Maximal Frequent Itemset Algorithm for Transactional Databases. Proceedings of the 17th International Conference on Data Engineering (ICDE), 2001, pp. 443–452, doi: 10.1109/ICDE.2001.914857.

[20] Lucchesse, C.—Orlando, S.—Perego, R.: DCI_Closed:: A Fast and Memory Efficient Algorithm to Mine Frequent Closed Itemsets. ICDM Workshop on Frequent Itemset Mining Implementations (FIMI '04), 2004.

[21] Lucchesse, C.—Orlando, S.—Perego, R.: Fast and Memory Efficient Mining of Frequent Closed Itemsets. IEEE Transactions on Knowledge and Data Engineering, Vol. 18, 2006, No. 1, pp. 21–36, doi: 10.1109/TKDE.2006.10.

[22] Pasquier, N.—Bastide, Y.—Taouil, R.—Lakhal, L.: Discovering Frequent Closed Itemsets for Association Rules. In: Beeri, C., Buneman, P. (Eds.): Database Theory – ICDT '99. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 1540, 1999, pp. 398–416.

[23] Pasquier, N.—Bastide, Y.—Taouil, R.—Lakhal, L.: Efficient Mining of Association Rules Using Closed Itemset Lattices. Information Systems, Vol. 24, 1999, No. 1, pp. 25–46, doi: 10.1016/S0306-4379(99)00003-4.

[24] Pei, J.—Han, J.—Mao, R.: CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. DMKD Workshop, 2000.

[25] Uno, T.—Kiyomi, M.—Arimura, H.: LCM Ver. 2: Efficient Mining Algorithms for Frequent/Closed/Maximal Itemsets. ICDM Workshop on Frequent Itemset Mining Implementations (FIMI '04), 2004.

[26] Wang, J.—Han, J.: BIDE: Efficient Mining of Frequent Closed Sequences. Proceedings of the 20th International Conference on Data Engineering (ICDE), 2004, doi: 10.1109/ICDE.2004.1319986.

[27] Wang, J.—Han, J.—Lu, Y.—Tzvetkov, P.: TFP: An Efficient Algorithm for Mining Top-k Frequent Closed Itemsets. IEEE Transactions on Knowledge and Data Engineering, Vol. 17, 2005, No. 5, pp. 652–664.

[28] Wang, J.—Han, J.—Pei, J.: CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets. Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03), 2003, pp. 236–245, doi: 10.1145/956750.956779.

[29] Zaki, M. J.—Hsiao, C.: ChARM: An Efficient Algorithm for Closed Association Rule Mining. SDM Conference, 2002.

[30] Extended BAKERY Dataset, `http://wiki.csc.calpoly.edu/datasets/wiki/ExtendedBakery`.

Savo Tomović received his Ph.D. in computer science from the University of Montenegro in 2011. He is currently Associated Professor in the Faculty of Science – Department of Mathematics and Computer Science at the University of Montenegro and Head of the Centre of the University Information System. He teaches a wide variety of undergraduate and graduate courses in several computer science disciplines, especially database systems, operating systems and programming. In addition, he is currently engaged as an adviser in Crnogorski Telekom on the project for data warehouse design and implementation. His primary research interest is in the area of data mining and artificial intelligence. During his Ph.D. studies he was involved in the project Linear Collider Flavour Identification (LCFI) with the aim to compare different data mining and classification algorithms as well as to understand the relative importance of the various input variables for the resulting tagging performance.

Predrag Stanišić is Full Professor in the Faculty of Science – Department of Mathematics and Computer Science at the University of Montenegro and Vice Chancellor of the University of Montenegro. He received his B.Sc. degree in mathematics and computer science from the University of Montenegro in 1996, his M.Sc. degree in computer science from the University of Belgrade, Serbia in 1998 and his Ph.D. degree in computer science from Moscow State University M. V. Lomonosov in 1999. He teaches a wide variety of undergraduate and graduate courses in several computer science disciplines, especially database systems, operating systems and programming.

Computing and Informatics, Vol. 37, 2018, 915–945, doi: 10.4149/cai_2018_4_915

MAXPART: AN EFFICIENT SEARCH-SPACE PRUNING APPROACH TO VERTICAL PARTITIONING

Benameur Ziani, Youcef Ouinten, Mustapha Bouakkaz

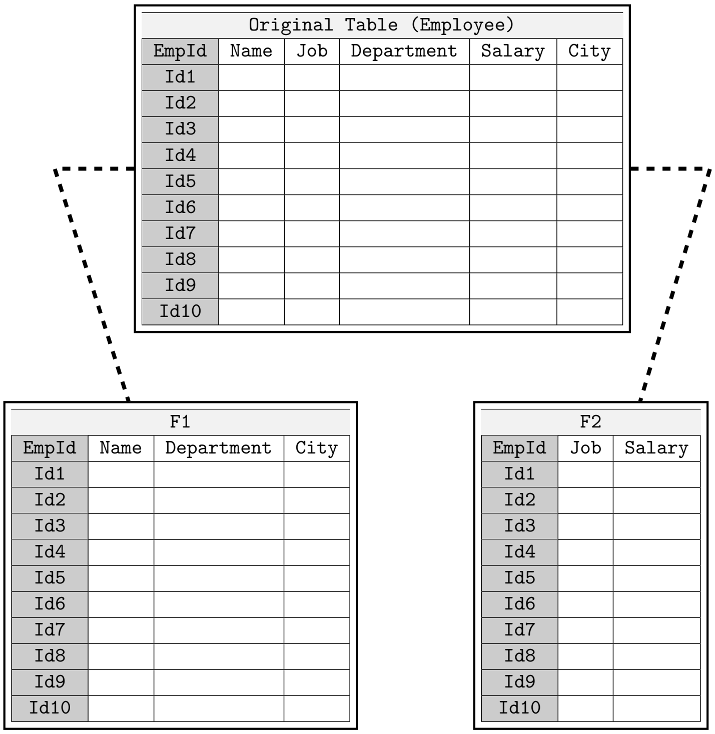
LIM – Department of Informatics
University of Laghouat, BP 37G M'kam 03000, Algeria
e-mail: {bziani, ouinteny, m.bouakkaz}@lagh-univ.dz

Abstract. Vertical partitioning is the process of subdividing the attributes of a relation into groups, creating fragments. It represents an effective way of improving performance in the database systems where a significant percentage of query processing time is spent on the full scans of tables. Most of proposed approaches for vertical partitioning in databases use a pairwise affinity to cluster the attributes of a given relation. The affinity measures the frequency of accessing simultaneously a pair of attributes. The attributes having high affinity are clustered together so as to create fragments containing a maximum of attributes with a strong connectivity. However, such fragments can directly and efficiently be achieved by the use of maximal frequent itemsets. This technique of knowledge engineering reflects better the closeness or affinity when more than two attributes are involved. The partitioning process can be done faster and more accurately with the help of such knowledge discovery technique of data mining. In this paper, an approach based on maximal frequent itemsets to vertical partitioning is proposed to efficiently search for an optimized solution by judiciously pruning the potential search space. Moreover, we propose an analytical cost model to evaluate the produced partitions. Experimental studies show that the cost of the partitioning process can be substantially reduced using only a limited set of potential fragments. They also demonstrate the effectiveness of our approach in partitioning small and large tables.

Keywords: Information systems, knowledge extraction, data mining, maximal frequent itemsets, database design, vertical partitioning

1 INTRODUCTION

Database applications are often characterized by a large volume of data and high demands with regard to query response time and transaction throughput. Vertical partitioning is an effective way of improving performance in the database systems where a significant percentage of query processing time is spent on the full scans of tables. It represents an important aspect of physical database design that have significant impact on performance and manageability [1]. Figure 1 illustrates the principle of vertical partitioning. Consider a classical example of the employee table (original table).



Original Table (Employee)

EmpId	Name	Job	Department	Salary	City
Id1					
Id2					
Id3					
Id4					
Id5					
Id6					
Id7					
Id8					
Id9					
Id10					

F1

EmpId	Name	Department	City
Id1			
Id2			
Id3			
Id4			
Id5			
Id6			
Id7			
Id8			
Id9			
Id10			

F2

EmpId	Job	Salary
Id1		
Id2		
Id3		
Id4		
Id5		
Id6		
Id7		
Id8		
Id9		
Id10		

Figure 1. Principle of vertical partitioning (`EmpId` is the key)

Assume that some employees' information, such as `Name`, `Department` and `City` is frequently required together. When scanning the employee table to fetch this information, other non-relevant information on employee such as `Job` and `Salary` will also be loaded. In such a situation and according to performance expectations, the administrator can split the original table into two fragments $F1$ (`Name`, `Department`, `City`) and $F2$ (`Job`, `Salary`). Such a partitioning is beneficial since it avoids access to non-relevant information and thus significantly reduces the I/O requirements during query processing. In order to minimize the costs of accessing the required data, the relation is partitioned in such a way that each query uses as few fragments as possible. The ideal partitioning is obtained if each query would have to access a single fragment containing exactly the attributes it references, resulting in minimal I/O requirements. Realistically, however, overlapping queries reduce the efficiency of the partitioning. In such a case additional joins between two or more fragments are required to fetch the desired data.

In vertical partitioning, the attributes of a relation $\mathcal{R}$ are clustered into non-overlapping groups and the original relation $\mathcal{R}$ is projected into fragment relations according to these attribute groups. The result of the fragmentation process is a set of fragments defined by a partitioning scheme. The aim is to find a partitioning scheme which minimizes the cost of accessing data during query processing.

However, selecting a suitable partitioning scheme is a difficult problem to solve, since a large space of alternatives must be considered. A table can be vertically partitioned in many different ways. The vertical partitioning problem is computationally intractable. Indeed, if a relation $\mathcal{R}$ has m non-primary key attributes, the possible fragments are given by the Bell number [2] which is approximately $B(m) \approx m^m$. Hence, on the one hand, the complexity of the partitioning problem increases exponentially with the number of attributes. Vertical partitioning, on the other hand, basically stores attributes that are frequently accessed together based on a given workload. But the latter may change over time which implies that the partitioning process may need to be performed very often. Accordingly, finding suitable vertical partitions is a daunting task even for skilled database administrators (DBAs). Thus, database administrators (DBAs) are faced with the challenging task of determining the appropriate choice of partitioned tables and, therefore, there is a practical need for strategies that assist the DBAs in this process.

Studies dealing with the vertical partitioning problem have focused on reducing its complexity and finding approximate solutions using heuristics-based approaches. A basic question related to vertical partitioning is how the attributes are referenced in a given workload? Therefore, most of the proposed algorithms cluster the attributes of a relation according to their *affinity*. Attribute affinity expresses a bond between a pair of attributes. The affinity between two attributes A_i and A_j measures the total number of accesses of queries referencing both attributes A_i and A_j. The core idea of affinity based partitioning is to compute affinities between every pair of attributes and then to cluster them such that high affinity pairs are as close in neighbourhood as possible.

The drawback of affinity based approaches is that the used measure does not reflect the closeness or affinity when more than two attributes are involved. In such approaches, all possible grouping of couples of attributes having high affinity should be examined. During regrouping the attributes are moved between fragments to achieve any possible improvement. This task requires a large number of comparison operations between affinity values of more than two attributes which can be very costly for a large representative workload.

The natural way to reflect the closeness of k attributes of a given relation $\mathcal{R}$ is to measure the accessing frequency of sets of attributes with different size s ($1 \leq s \leq k$). This measure can be achieved by the means of mining frequent itemsets. This technique helps to discover important associations among attributes such that the presence of some attributes in a query will imply the presence of some other attributes. Thus, the partitioning process can be done faster and more accurately with the help of such knowledge discovery technique of data mining. However, the use of all frequent itemsets is limited by the high computational cost as well as the large number of resulting outputs. Such approach generates an enormous number of candidate fragments which leads to high computational overheads.

In this paper we propose `MaxPart` – an approach for finding an optimized solution to vertical partitioning using maximal frequent itemsets. The proposed approach exploits the input workload information to intelligently prune the search space of the optimal solutions. It measures the correlation of attribute sets as naturally expected by the partitioning process. Taking a representative workload as an input, `MaxPart` goes through two major steps

1. enumerating potential fragments, that we call candidate fragments, according to the workload characteristics using maximal frequent itemsets technique;
2. generating possible partitioning schemes exploiting the candidate fragments and selecting the best one according to the workload cost.

We address the complexity of vertical partitioning problem from the perspective of counting a reduced number of optimized possible solutions as efficiently as possible. We are motivated by the desire to achieve computationally simpler, but at least equivalent, solutions for the studied problem. The counting aspect reveals the inherent computational complexity of the partitioning process. We believe that maximal frequent itemsets offer an attractive alternative to achieve this goal.

The reminder of the paper is as follows. We present in Section 2 an example to motivate our proposal. Section 3 provides background information on the studied problem. In Section 4, we investigate related work on vertical partitioning in relational databases and maximal frequent itemsets mining. Section 5 presents the proposed analytical cost model for evaluating generated partitioning schemes. We present the proposed approach to solve the vertical partitioning problem in Section 6. Section 7 deals with the experimental study of the proposed approach. We conclude the paper and present future directions in Section 8.

2 MOTIVATING EXAMPLE

We will first introduce the principle of MaxPart approach through a simple motivating example. An obvious, but important observation, is that the efficiency of the partitioning process depends on the interestingness of the generated fragments in terms of size (length) and frequency. Based on a given workload, expected fragments must, intuitively, store the maximum of attributes that are frequently accessed together.

	Queries	**Freq.**
q_1	Select A, B, E	1
q_2	Select B, E	3
q_3	Select A, D, F	3
q_4	Insert SQL	2
q_5	Insert SQL	1

a)

Extraction Context					
A	B	E			
B	E				
B	E				
B	E				
A	D	F			
A	D	F			
A	D	F			
A	B	C	D	E	F
A	B	C	D	E	F
A	B	C	D	E	F

b)

	AllPart	**MaxPart**
1-Fragment	A, B, D, E, F	–
2-Fragment	AB, AD, AE DF, AF, BE	–
3-Fragment	ADF, ABE	ADF, ABE

c)

Figure 2. Example of a) input workload, b) corresponding extraction context and c) candidate fragments

Example 1. The following example is extracted from [3]. It illustrates the efficiency of using maximal frequent itemsets for the partitioning process. Instead of using all frequent itemsets, our approach, called MaxPart, performs the vertical partitioning exploiting the interesting properties of maximal frequent itemsets. Consider a set of 5 queries $q_1, \ldots, q_5$ referencing the attributes A, B, C, D, E, and F. The queries and their frequencies are illustrated in Figure 2 a). Figure 2 b) shows the corresponding extraction context. For a threshold value % of 40 %, the set of candidate fragments generated using all frequent itemsets, which we call AllPart, and by the means of MaxPart approach are shown in Figure 2 c). AllPart generates 13 candidate fragments, while MaxPart generates only two candidate fragments.

According to the algorithm proposed in [3], the process of generating the possible partitioning schemes is done as follows. AllPart starts by considering the fragment with the maximal length $\{A, D, F\}$ as the first fragment of the partition. It scans successively the 2-itemsets and find that $\{B, E\}$ does not overlap with the existing partition. It forms the second fragment. The remaining attribute $\{C\}$, that is non frequent, forms the third fragment. The result of this iteration is the partitioning scheme $[\{A,D,F\},\{B,E\},\{C\}]$. Similarly, the second large fragment $\{A,B,E\}$ leads to the scheme $[\{A,B,E\},\{D,F\},\{C\}]$. The MaxPart approach performs only two comparisons between the fragments $\{A,D,F\}$ and $\{A,B,E\}$ to generate the same partitioning schemes. Indeed, the maximal fragment $\{A,D,F\}$ is firstly compared to the fragment $\{A,B,E\}$ to deduce the second fragment $\{B,E\}$. Similarly, the comparison between the fragments $\{A,B,E\}$ and $\{A,D,F\}$ generates the second fragment $\{D,F\}$. In both cases, the remaining attribute $\{C\}$, which is non frequent, forms the third fragment.

The above example clearly shows that our approach prunes significantly the search space for the partitioning process. When processing a potential fragment F_k, which has a maximal length, AllPart approach needs to compare it to the fragments $F_{k-1}, F_{k-2}, \dots, F_1$ resulting in a larger total computation cost. However, most of the fragments $F_{k-1}, \dots, F_1$ will not be used any more since they are included in the maximal ones. As the number of attributes increases using large representative workloads, the computational cost grows exponentially. Obviously, the efficiency of the partitioning process depends on the number of enumerated fragments. Indeed, the selection of an optimized partitioning scheme becomes computationally more complex when there is a huge number of candidate fragments to choose from. As the number of the candidate fragments becomes longer, the number of possible comparisons becomes larger, thus the pruning effect of our approach is sharper. Hence, our proposal seems a good approach to cope with scalability issues. We believe that the way partitioning schemes are achieved defines the approach's ability to scale. The objective of our approach is to reduce the computational cost of the partitioning process without compromising its correctness.

3 BACKGROUND

3.1 Workload-Based Vertical Partitioning

Fragmentation is a design technique to divide a single database into two or more partitions such that the combination of the partitions yields the original database without any loss or addition of information [4]. The result of the fragmentation process is a set of fragments defined by a partitioning scheme. The objective is to create vertical fragments of a relation so as to minimize the cost of accessing data during transaction processing. In vertical partitioning, attributes of a relation $\mathcal{R}$ are clustered into groups and the relation $\mathcal{R}$ is projected into fragment relations according to these attribute groups.

A general formulation of the vertical partitioning problem is as follows: Given a relation $\mathcal{R}$ of k attributes $\mathcal{R} = \{A_1, A_2, \dots, A_k\}$ and a representative workload $\mathcal{W}$ of n queries $\{q_1^{f_1}, q_2^{f_2}, \dots, q_n^{f_n}\}$, where each query $q_i (1 \leq i \leq n)$ has an access frequency f_i, the vertical partitioning problem involves selecting a partitioning scheme, $\mathcal{F} = \{F_1, \dots, F_m\}$ among all possible partitioning schemes such that:

1. Every fragment $F_i \subseteq \mathcal{F}$ is composed of a subset of the attributes of $\mathcal{R}$ plus the identifier column (primary key) which is used for join operations to reconstruct the original data.
2. $\forall F_i \in \mathcal{F}, \forall F_j \in \mathcal{F}, i \neq j : F_i \cap F_j = \emptyset$ (except for the primary key).
3. $\mathcal{R} = \bowtie_{i=1}^{m} F_i$.
4. The cost of processing the workload $\mathcal{W}$ using the partitioning scheme $\mathcal{F}$ is minimum.

In order to minimize the costs of accessing the required data, the relation is partitioned in a way that each query uses as few fragments as possible. The ideal partitioning scheme is obtained if each query q_i in the workload $\mathcal{W}$ would have to access a single fragment containing exactly the attributes it references, resulting in minimal I/O requirements. Realistically, however, the workload will contain overlapping between queries which reduces the efficiency of vertical partitioning. In such a case additional joins between generated fragments are required to answer some of the queries in the workload.

3.2 Basic Concepts on Frequent Itemsets Mining

To facilitate the understanding of our approach, we briefly sketch, in this section, the key notions on frequent itemsets mining.

Definition 1 (Extraction context)**.** An extraction context (or a formal context) is a triplet $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$, where $\mathcal{O}$ represents a finite set of objects (or transactions), $\mathcal{I}$ is a finite set of attributes (or items) and $\mathcal{R}$ is a binary relation (i.e., $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{I}$). Each pair $(o, i) \in \mathcal{R}$ expresses that the object $o \in \mathcal{O}$ contains the item $i \in \mathcal{I}$.

	A	B	C	D	E
1			x	x	
2		x	x		
3	x				x
4		x			x
5		x	x		

Table 1. Example of extraction context

An example is illustrated in Table 1. Transactions are denoted by numbers and items by letters. We have

$$\mathcal{I} = \{A, B, C, D, E\}, \quad \mathcal{O} = \{1, 2, 3, 4, 5\}$$

and

$$\mathcal{R} = \{(1, C), (1, D), (2, B), (2, C), (3, A), (3, E), (4, B), (4, E), (5, B), (5, C)\}.$$

Definition 2 (Itemset)**.** An itemset (or k-itemset) is a subset $I \subseteq \mathcal{I}$ that contains k items ($|I| = k$).

Definition 3 (Support of an itemset)**.** Let $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ be an extraction context and $I \subseteq \mathcal{I}$ be an itemset. The support of I, denoted Support(I), is the number of transactions containing all the items of I, divided by the total number of transactions:

$$\text{Support}(I) = \frac{|\{o \in \mathcal{O}/(\forall i \in I, (o,i) \in \mathcal{R}|}{|\mathcal{O}|}.$$

Definition 4 (Frequent itemset)**.** Let $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ be an extraction context and $I \subseteq \mathcal{I}$ be an itemset. The itemset I is said to be frequent if Support(I) $\geq$ minsup where minsup is a user-defined support threshold.

Definition 5 (Maximal frequent itemset)**.** Let $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ be an extraction context and $I \subseteq \mathcal{I}$ be an itemset. The itemset I is said to be maximal frequent itemset if I is frequent and no super-set of I is frequent:

$$\{(\text{Support}(I) \geq \text{minsup}) \wedge (\forall J \subseteq \mathcal{I} : I \subset J \Rightarrow \text{Support}(J) < \text{minsup})\}.$$

Definition 6 (Frequent itemset mining problem)**.** Let $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ be an extraction context. The frequent itemsets mining problem requires to discover all frequent itemsets given a user-defined minimum support minsup.

4 RELATED WORK

4.1 Vertical Partitioning in Relational Databases

The idea of data partitioning was proposed in the early days of databases as a means to increase I/O performance and it has been studied in different contexts. It is worth pointing that the first works on the vertical partitioning have been proposed in the context of centralized relational databases. With the emergence of new models of data, the existing approaches have been widely adopted by taking into account the characteristics of each of the emerging models. For example, in a centralized context, the fragmentation aims at reducing the query processing costs whereas in a distributed context it aims also at achieving a better distribution of data over distant sites so as to achieve a parallelized treatment of queries. Thus, several vertical partitioning approaches have been proposed for centralized databases [3, 5, 6, 7, 8, 9, 10, 11], distributed databases [12, 13, 14, 15, 16, 17, 18], object-oriented databases [19, 20, 21, 22], data warehouse design [23, 24, 25] and XML database systems [26, 27, 28, 29]. Due to space constraints, we cannot possibly list all the related references here. Instead, we will restrict our discussion to related work on vertical partitioning in centralized relational databases and to used techniques that are directly related to our work.

The NP-hard nature of the vertical partitioning was pointed pretty early [2]. Therefore, studies dealing with this problem have focused on reducing its complexity and finding approximate solutions using heuristics-based approaches. In the literature, most of the proposed algorithms cluster attributes of a relation according to their affinity [5, 6, 7, 8, 9]. The core idea of affinity based partitioning is to compute affinities between every pair of attributes and then to cluster them such that high affinity pairs are as close in neighbourhood as possible. The proposed approaches measure the affinity between pairs of attributes and try to cluster attributes according to their pairwise affinity by using the bond energy algorithm(BEA) [30]. They started from constructing an attribute usage matrix (AUM) to construct the attribute affinity matrix (AAM) on which clustering is performed. The affinity matrix is an $n \times n$ matrix for the n-attribute problem where the (i, j) element equals the *between-attributes* affinity. Affinity between attributes measures the total number of accesses of queries referencing both attributes i and j. The attribute affinity matrix helps to perform a first clustering of attributes. A binary partitioning is repetitively applied in a second step. The authors in [9] developed an algorithm based on graphical technique. It considers the attribute affinity matrix as a complete graph called the *affinity graph*. Each edge value in the affinity graph represents the affinity between two attributes. A linearly connected spanning tree is constructed from the affinity graph and all cycles of the spanning tree form fragments of the relation.

Despite their simplicity, affinity based approaches, however, have the following shortcomings.

1. The metric used for clustering the attributes does not reflect the closeness or affinity when more than two attributes are involved. Thus, it complicates the comparison process between affinity values of more than two attributes which can be very costly for a large workload. In such approaches, all possible grouping of couples of attributes having high affinity should be examined. During regrouping, attributes are moved between fragments to achieve any possible improvement.
2. The proposed approaches performs a binary partitioning. Realistically, however, this is not always an optimal solution. In real cases, the attributes can be grouped into more than two fragments especially for a table having a very large number of attributes.
3. Most of the proposed vertical partitioning algorithms do not have an objective function to evaluate the *goodness* of partitions that they produce.

Some approaches use Genetic Algorithms to perform the partitioning [10, 11]. A binary string is used as the genetic representation of the partitioning. As an example, suppose a relation to be partitioned consists of ten attributes. A binary string (solution or chromosome) representing a binary fragmentation scheme is 1110001010. This solution specifies that attributes 1, 2, 3, 7 and 9 constitute one fragment, while attributes 4, 5, 6, 8 and 10 constitute the other fragment. The genetic algorithm starts with an initial population P_0 that is usually chosen at random. A process of selection-crossover-mutation is repeated to form a final optimized solution. The binary partitioning performed, however, is not always an optimal solution since the attributes can be grouped into more than two fragments as mentioned above. Furthermore, despite their theoretical success, Genetic Algorithms still suffer from their parameters setting challenge. Indeed, finding the best parameter values is not a trivial task for the DBAs and it is difficult to understand the effect of every parameter. Many parameters have effects on other parameters which makes the problem even more complex.

As vertical partitioning aims at grouping the attributes that are frequently referenced together, frequent itemsets mining appears as a natural solution to this problem. In [3] a vertical partitioning approach using this data mining technique was proposed. The proposed approach exploits the Apriori algorithm [31, 32] to generate all possible partitioning schema (called candidate fragments). Although the proposed approach is suitable for the vertical partitioning, the use of all frequent itemsets is limited by the high computational cost as well as the large number of resulting fragments. It is well known that frequent itemsets often generate a huge number of fragments that is very costly to handle. Furthermore, most of generated fragments are redundant resulting in a larger total computation cost. This cost increases exponentially with the number of the attributes in the workload. Thus, frequent itemsets is not the best choice for solving the partitioning problem. However, as we have been motivated in Section 2, the partitioning process can be greatly improved by the use of maximal frequent itemsets. Alluding to the closure principle of frequent itemsets which means that every subset of a frequent itemset is also frequent, the maximal frequent itemsets can imply and include all information of the frequent itemsets. Because of this, we believe that mining maximal frequent itemsets provides an interesting alternative since it generates the largest and the most frequent fragments which match the partitioning problem requirements.

4.2 Maximal Frequent Itemsets Mining

The Knowledge Discovery from Database (KDD) means the non-trivial process of trawling through data to find previously unknown relationships among the data that are interesting to the user of the data [33]. Data mining is the core step of the KDD process. Since its conception in the late 1980s, data mining has achieved tremendous success. Many new problems have emerged and have been solved by data mining researchers [34]. Due to its rich variety of important and challenging problems, data mining is proving to be a fruitful research arena for knowledge and information engineering [35]. Data mining is defined as the set of intelligent, complex, and highly sophisticated data processing techniques used to extract knowledge. Knowledge can take several forms depending on the purpose of the user and the data mining algorithm. Mining Frequent itemsets is a data mining task which consists of finding meaningful relationships between objects (items) of a database. It leads to the discovery of associations and correlations among objects in data sets which can help in many decision-making processes. It is one of the most widely used techniques in data mining and knowledge discovery. It was originally proposed in [31] with the Apriori algorithm. The drawback of mining all frequent itemsets is that if there is a large frequent itemset with size s, then almost all 2^s candidate subsets of the itemset might be generated and tested. Furthermore, the number of frequent itemsets grows very quickly when the minimum support threshold is decreased. Consequently, the complexity of the mining task becomes rapidly intractable. Moreover, the huge size of the output complicates the task of the analyst (final user), who has to extract useful knowledge from a very large amount of results. This drawback is known as pattern explosion. Often an unwieldy number of results is produced, comprising strongly redundant information.

Maximal frequent itemsets are a well known solution to the shortcomings described above. A frequent itemset is called maximal if it has no superset that is frequent. Maximal frequent itemsets are a small subset of frequent itemsets, but they represent exactly the same knowledge in a more succinct way. Using this condensed representation, it is straightforward to derive the set of all frequent itemsets. Hence, the problem of mining frequent itemsets can be reduced to mining maximal frequent itemsets. Moreover, many practical data mining applications only require mining maximal frequent itemsets rather than mining all frequent itemsets [36]. Interestingly, this problem has a strong connection to Formal Concept Analysis (FCA) [37]. FCA and frequent itemsets mining are two research fields that are closely related to each other [38, 39] and several research works showed that FCA provides a strong theory for improving both performance and results of frequent itemsets mining algorithms [40, 41, 42, 43, 44, 45].

A workshop dedicated to different implementation methods of frequent itemsets mining (FIMI) is reported in [46]. Several algorithms [47, 48, 49, 50, 51] have been tested and an analysis of each algorithm is performed, highlighting its performance. More information on the implemented methods and experimental data can be found in [46]. Within the category of mining maximal frequent itemsets, FPMAX [51] is stated to be the best algorithm presented at the cited Workshop. Thus, we have used this algorithm to validate our approach. Of course, this does not guarantee that a more efficient implementation cannot be found. Our choice is motivated by the results published in [46]. FPMAX is as an extension of the FPGrowth method [52]. It builds a special data structure called MFI-Tree (Maximal Frequent Itemsets Tree) to store all maximal frequent itemsets discovered. The MFI-Tree is similar to an FP-Tree used by the FPGrowth method. An FP-Tree is a compact representation of all relevant frequency information in the original database. Every branch of the FP-Tree represents a frequent itemset. The nodes along the branches store, in decreasing order, the frequencies of the corresponding items. In [53] we have presented the principle and a java implementation of FPMAX algorithm.

5 ANALYTICAL COST MODEL

The number of disk accesses, and thus the amount of the data transferred, have been the most commonly used parameters to evaluate the cost of query processing on a given database. While the table is the basic unit in the relational databases, we firstly provide an analytical cost model to evaluate the workload cost on a single partitioned table. Then we broaden our view to the data warehouse context where the queries involve several tables.

In the context of a single partitioned table, we assume that the cost of processing a query q on a partitioning scheme is the sum of

1. the cost of joins between fragments needed to answer the query q and
2. the cost of processing q on the resulted portion of the original data.

Notations	**Meaning**
$\|\|X\|\|$	Total number of tuples in a table X
$\|X\|$	Total size, in bytes, of the attributes of a table (or a fragment) X
DBS	Database bloc size in bytes
B_X	Number of blocks needed to store a table (or a fragment) X
B_q^F	Number of blocks to be accessed by the query q in a fragment F
t_q	Number of tuples satisfying a query q

Table 2. Cost model notations.

Based on the number of I/Os needed for joining two tables X and Y given in [54] and the number of blocks accessed for processing a query q given in [55], we propose an analytical cost model to evaluate the workload cost on a partitioning scheme. Table 2 summarizes the notations used in our cost model. For the join operations between two tables X and Y, we assume that all joins are achieved by the hash-join method. The join attributes are used as hash keys in both tables X and Y. The join operation can be viewed as consisting of two phases:

1. Hash phase: Each table is read/written once. The number of I/Os needed is then: $2 \times (B_X + B_Y)$, where B_X and B_Y are the number of disk blocks needed to store the tables X and Y, respectively.
2. Merge phase: Each table is read once. Consequently the cost of this phase is $B_X + B_Y$. The total cost for joining X and Y is then [54]:

$$C_{\bowtie} = 3 \times (B_X + B_Y). \tag{1}$$

In the partitioning scheme, we apply this formula for each join performed between the fragments needed to answer a given query q.

As in [3], we use the number of blocks estimate as the cost of the second step of query processing. Let R be a relation of n tuples and m the number of blocks needed to store R. Assume that k tuples satisfy a given query q and are distributed uniformly among the m blocks. Then the number of blocks accessed to process the query q is given by [55]:

$$B_q^R = m \times \left(1 - \left(1 - \frac{1}{m}\right)^k\right). \tag{2}$$

To evaluate the workload cost on a partitioning scheme we apply this formula, for each query q, on the portion of data obtained by joining the fragments needed to answer q. Thus, the cost of processing a query q, on a partitioning scheme, is estimated as follows:

1. **Evaluate the cost of joining the fragments required by the query q:** Let R be the original relation. Assume that $F_q = \{F_q^1, F_q^2, \ldots, F_q^N\}$ is the set of fragments required to answer the query q. The portion P_q^{Final} of original data needed to answer q is obtained by:

$$P_q^{Final} = \bowtie_{i=2}^{N} \left(F_q^i, P_q^{i-1}\right) \tag{3}$$

where $P_q^1 = F_q^1$ and P_q^i represents the intermediate portions of data obtained after the i^{th} join. Using Equation (1), the cost of performing the joins is:

$$\text{Cost}_{\bowtie} = \sum_{i=2}^{N-1} 3 \times \left(B_{F_q^i} + B_{P_q^{i-1}}\right) \tag{4}$$

where

$$B_X = \frac{||R|| \times |X|}{DBS}.$$

2. **Estimate the number of blocks to be accessed in the result portion:** Let $B_q^{P_q^{Final}}$ be the number of blocks to be accessed in the portion P_q^{Final} required to answer the query q. Using Equation (2), we have:

$$B_q^{P_q^{Final}} = \left\lceil B_{P_q^{Final}} \left(1 - \left(1 - \frac{1}{B_{P_q^{Final}}}\right)^{t_q}\right)\right\rceil \tag{5}$$

where

$$B_{P_q^{Final}} = \frac{||R|| \times |P_q^{Final}|}{DBS}.$$

Consequently, the cost of processing a given query q is:

$$\text{Cost}(q) = \left[\sum_{i=2}^{N-1} 3 \times \left(B_{F_q^i} + B_{P_q^{i-1}}\right)\right] + B_q^{P_q^{Final}}. \tag{6}$$

Finally, we have:

$$\begin{aligned}\text{Cost}(q) = &\left[\sum_{i=2}^{N-1} 3 \times \left(\frac{||R|| \times \left(|F_q^i| + |P_q^{i-1}|\right)}{DBS}\right)\right] \\ &+ \left\lceil \frac{||R|| \times |P_q^{Final}|}{DBS} \times \left(1 - \left(1 - \frac{1}{\frac{||R|| \times |P_q^{Final}|}{DBS}}\right)^{t_q}\right)\right\rceil,\end{aligned} \tag{7}$$

$$\mathrm{Cost}(q) = \frac{||R||}{DBS}\left[\left(\sum_{i=2}^{N-1} 3 \times \left(|F_q^i| + |P_q^{i-1}|\right)\right) + \left(|P_q^{Final}| \times \left(1 - \left(1 - \frac{1}{\frac{||R|| \times |P_q^{Final}|}{DBS}}\right)^{t_q}\right)\right)\right]. \tag{8}$$

In data warehouse environment, the above cost model has to be changed in order to suit the new context. A data warehouse stores a large volume of data and is usually organized in a star schema. A typical star schema consists of a large central fact table linked to multiple dimension tables through primary-foreign key relationships. Dimensions tables are usually much smaller then the fact table. Processing queries over a star schema is expensive. The major bottleneck in evaluating such queries has been the joins of the central (and usually very large) fact table with the surrounding dimension tables (also known as a star joins). The proposed partitioning reduces the size of the fact table tuples participating in a sequence of joins. This way, the joins are performed on a much smaller size tuples. This will be obviously more efficient than the original tuples with much less I/O cost.

In such a context, we assume that the cost of processing a query q on a partitioning scheme is the sum of the two following costs:

1. $\mathrm{Cost}^1_{\bowtie}$: the cost of joins between the fact table fragments needed to answer the query q. The ideal partitioning is obtained if each query would have to access a single fragment containing exactly the attributes it references, This way, additional joins are avoided.

2. $\mathrm{Cost}^2_{\bowtie}$: the cost of joins between the resulted portion of the original fact table and the dimension tables involved by q.

Thus, the cost of processing a query q, on a partitioning scheme, is estimated as follows:

1. **Evaluation of the cost of joining the fragments of the fact table required by the query** q**:** Let F be the original fact table. Assume that $F_q = \{F_q^1, F_q^2, \dots, F_q^N\}$ is the set of fragments required to answer the query q. The cost of performing the joins is obtained in precisely the same manner as in the preceding cost model (Equation (4)):

$$\mathrm{Cost}^1_{\bowtie} = \sum_{i=2}^{N-1} 3 \times \left(B_{F_q^i} + B_{P_q^{i-1}}\right) \tag{9}$$

where

$$B_X = \frac{||F|| \times |X|}{DBS}, \tag{10}$$

$$\text{Cost}^1_{\bowtie} = \sum_{i=2}^{N-1} 3 \times \left(\frac{||F|| \times |F_q^i|}{DBS} + \frac{||F|| \times |P_q^{i-1}|}{DBS} \right), \tag{11}$$

$$\text{Cost}^1_{\bowtie} = \frac{3 \times ||F||}{DBS} * \left[\sum_{i=2}^{N-1} \left(|F_q^i| + |P_q^{i-1}| \right) \right]. \tag{12}$$

2. **Evaluation of the cost of joining the result portion with the dimension tables involved by q:** Let F_q^{Final} be the portion of the original fact table needed to answer q and $\{D_1, D_2, \dots, D_d\}$ the set of the dimension tables involved by q. We have:

$$\text{Cost}^2_{\bowtie} = 3 \times \left(B_{F_q^{Final}} + B_{D_1} \right) + \dots + 3 \times \left(B_{F_q^{Final}} + B_{D_d} \right), \tag{13}$$

$$\text{Cost}^2_{\bowtie} = 3 \times \left(d * B_{F_q^{Final}} + B_{D_1} + \dots + B_{D_d} \right), \tag{14}$$

$$\text{Cost}^2_{\bowtie} = 3 \times \left(\frac{d * ||F|| * |F_q^{Final}|}{DBS} + \frac{||D_1|| * |D_1|}{DBS} + \dots + \frac{||D_d|| * |D_d|}{DBS} \right), \tag{15}$$

$$\text{Cost}^2_{\bowtie} = \frac{3}{DBS} \times \left(d * ||F|| * |F_q^{Final}| + \sum_{i=1}^{d} \left(||D_i|| * |D_i| \right) \right). \tag{16}$$

Finally, we have:

$$\begin{aligned}
\text{Cost}(q) &= \text{Cost}^1_{\bowtie} + \text{Cost}^2_{\bowtie}, \\
\text{Cost}(q) &= \frac{3 \times ||F||}{DBS} * \left[\sum_{i=2}^{N-1} \left(|F_q^i| + |P_q^{i-1}| \right) \right] \\
&\quad + \frac{3}{DBS} * \left[d \times ||F|| \times |F_q^{Final}| + \sum_{i=1}^{d} \left(||D_i|| * |D_i| \right) \right], \qquad (17) \\
\text{Cost}(q) &= \frac{3}{DBS} * \left[||F|| \times \left(\sum_{i=2}^{N-1} \left(|F_q^i| + |P_q^{i-1}| \right) + d \times |F_q^{Final}| \right) \right. \\
&\quad \left. + \sum_{i=1}^{d} \left(||D_i|| * |D_i| \right) \right]. \qquad (18)
\end{aligned}$$

In both cases, the cost of processing the workload $\mathcal{W}$ is:

$$\text{Cost}(\mathcal{W}) = \sum_{q_i \in \mathcal{W}} (\text{Cost}(q_i) \times f_i).$$

6 THE MAXPART APPROACH

6.1 MaxPart Overview

This section describes the MaxPart approach outlined in Algorithm 1. MaxPart takes as input i) a relation $\mathcal{R}$, ii) a workload $\mathcal{W}$ and iii) a predefined threshold value σ and returns an optimized partitioning scheme $\mathcal{F}_{opt}$ which minimizes the cost of the workload $\mathcal{W}$. The MaxPart approach goes through the following main steps:

1. **Construction of the extraction context:** Given a representative workload $\mathcal{W} = \{q_1^{f_1}, q_2^{f_2}, \dots, q_n^{f_n}\}$, where each query $q_i (1 \leq i \leq n)$ has an access frequency f_i, we build the extraction context for mining maximal frequent itemsets. It expresses the access patterns of queries to attributes. Accesses to attribute by queries are represented by a text file where each row represents a query $q_i (1 \leq i \leq n)$ and each column a non-key attribute A_j involved in the corresponding query. Each row i corresponding to the query q_i is duplicated f_i times which corresponds to the frequency of the query q_i. The extraction context has $\sum_{i=1}^{n} f_i$ rows. The number of columns in each row depends on the number of attributes involved in the considered query. For retrieval transactions, the set of attributes in the `SELECT` clause are considered. For an `INSERT/DELETE` transaction, all the attributes in the relation are used.
2. **Generation of candidate fragments:** For a given value of threshold, we generate the corresponding sets of maximal frequent itemsets using the FPMAX algorithm. Each generated itemset corresponds to a candidate fragment. The generated fragments are then clustered into classes. Each class contains the fragments with the same length (number of attributes). The fragments in each class are sorted in descending order according to their support.
3. **Generation of possible partitioning scheme:** Let $\mathcal{F} = \{\mathcal{F}_k, \dots, \mathcal{F}_1\}$ be the set of classes of fragments generated in the previous step. Each class $\mathcal{F}_i$ corresponds to the fragments having length i. For all $\mathcal{F}_j \in \mathcal{F}_k (1 \leq j \leq size(\mathcal{F}_k))$, we construct a possible partitioning scheme as follows:

   - The itemset $\mathcal{F}_j$ is taken as the first fragment.
   - The classes $\mathcal{F}_k, \mathcal{F}_{k-1}, \dots, \mathcal{F}_1$ are successively examined to construct non-overlapping fragments having the maximal length and the highest support.
   - Finally, each non-frequent single attribute is considered as a fragment.

   The result of this step is a set $\{P_1, P_2, \dots, P_p\}$ of the possible partitioning schemes.

4. **Evaluation of the generated partitioning scheme:** The Partitioning schemes generated in the previous step closely match the requirements of the workload provided. They are evaluated using the cost model proposed in Section 5. The partitioning scheme with the smallest cost is recommended.

6.2 Illustrative Example

The purpose of this section is to illustrate the working of the MaxPart approach in the context of a single table. Consider the workload example given in Section 2. Assume the input considerations summarized in Figures 3 a) and 3 b). Consider that the number of tuples of the relation R to be partitioned is $||R|| = 150$. The database block size is assumed to be 100 bytes. The MaxPart approach goes through the following steps:

Query	Attributes	Frequency	# of Tuples
q_1	A, B, E	1	2
q_2	B, E	3	60
q_3	A, D, F	3	20
q_4	A, B, C, D, E, F	2	10
q_5	A, B, C, D, E, F	1	8

a)

Attribute	Size (byte)
A	1
B	4
C	8
D	2
E	1
F	2

b)

Figure 3. Characteristics of a) queries and b) attributes

1. **Construction of the extraction context:** Figure 4 illustrates the extraction context for our example.
2. **Generation of candidate fragments:** Assuming, for example, a minimum support value of 40 %, generated maximal frequent itemsets are $\{A, D, F\}(6)$ and $\{A, B, E\}(4)$. The number in brackets represents the frequency of the itemset. Each generated itemset corresponds to a candidate fragment.
3. **Generation of possible partitioning scheme:** The maximal itemset $\{A, D, F\}$, having the highest frequency, is taken as the first fragment. Considering the itemset $\{A, B, E\}$, we construct the fragment $\{B, E\}$ that does not overlap with the existing partition. The remaining attribute $\{C\}$, which is infrequent, is taken

Algorithm 1 MaxPart algorithm

Require: Workload $\mathcal{W} = \{q_1^{f_1}, q_2^{f_2}, \ldots, q_n^{f_n}\}$, predefined threshold σ.
Ensure: Optimized partitioning scheme $\mathcal{F}_{opt} = \{F_1, F_2, \ldots, F_k\}$

```
1:  Begin
2:  $\mathcal{EC} \longleftarrow \emptyset$;          ▷ Construction of the extraction context $\mathcal{EC}$
3:  Row ← ””;
4:  for all $(q_i^{f_i} \in \mathcal{W})$ do
5:      Row ← Candidate_Attributes($q_i$);
6:      for $(i \leftarrow 1, f_i)$ do
7:          Writeln($\mathcal{EC}$, Row);
8:      end for
9:  end for
10:         ▷ Mining maximal fragments using the extraction context $\mathcal{EC}$
11: $\mathcal{F}_{\mathcal{W}}[..] \leftarrow FPMAX(\mathcal{EC}, \sigma)$;          ▷ $\mathcal{F}_{\mathcal{W}}$ Set of generated fragments
12: $\mathcal{F}_{\mathcal{W}}[..] \leftarrow Sort(\mathcal{F}_{\mathcal{W}}[..], length, frequency)$;  ▷ Sorting the fragments in $\mathcal{F}_{\mathcal{W}}$
13:                                   ▷ $\mathcal{F}_{\mathcal{W}}[..] = \{\mathcal{F}_k, \mathcal{F}_{k-1}, \ldots, \mathcal{F}_1\}$
14:         ▷ Generating possible partitioning scheme using $\mathcal{F}_{\mathcal{W}}[..]$
15: $\mathcal{PPS} \leftarrow \emptyset$;          ▷ $\mathcal{PPS}$ Set of possible partitioning scheme
16: for $(i \leftarrow 1, size(\mathcal{F}_k))$ do
17:     $P \leftarrow \mathcal{F}_k[i]$;
18:     $P_{temp}[..] \leftarrow \emptyset$;
19:     for $(p \neq P, p \in \mathcal{F}_k, \mathcal{F}_{k-1}, \ldots, \mathcal{F}_1)$ do
20:         $P_{temp}[..] \leftarrow p - (P \cap p)$;      ▷ Construct non-overlapping fragments
21:     end for
22:     while $P_{temp}[..] \neq \emptyset$ do
23:         $P_1 \leftarrow$ Fragment $\in P_{temp}[..]$ with maximal size and highest support;
24:         $P \leftarrow P \cup P_1$;
25:         $P_{temp}[..] \leftarrow P_{temp}[..] - P_1$;
26:     end while
27:     $\mathcal{PPS} \leftarrow \mathcal{PPS} \cup P$;
28: end for
29:                                   ▷ Find optimal partitioning scheme
30: $\mathcal{F}_{opt} \leftarrow \mathcal{PPS}[0]$;
31: for $(i \leftarrow 1, size(\mathcal{PPS}))$ do
32:     if Cost($\mathcal{W}, \mathcal{F}_{opt}$) > Cost($\mathcal{W}, \mathcal{PPS}[i]$) then
33:         $\mathcal{F}_{opt} \leftarrow \mathcal{PPS}[i]$;
34:     end if
35: end for
36: Return $\mathcal{F}_{opt}$;
37: End.
```

A	B	E			
B	E				
B	E				
B	E				
A	D	F			
A	D	F			
A	D	F			
A	B	C	D	E	F
A	B	C	D	E	F
A	B	C	D	E	F

Figure 4. Example of an extraction context

as a fragment, resulting in the partitioning scheme $\{[A, D, F], [B, E], [C]\}$. Similarly, the second maximal fragment $\{A, B, E\}$ leads to the partitioning scheme $\{[A, B, E], [D, F], [C]\}$.

4. **Evaluation of the generated partitioning scheme:** After constructing the possible partitioning schemes, we now apply our cost model to recommend the one having the least cost.

- **Using the partitioning scheme $\{[A, D, F], [B, E], [C]\}$:** Consider the query q_1 which references the attributes A, B and E. To answer q_1, we need a join between the fragments $[A, D, F]$ and $[B, E]$ resulting in the final portion $[A, B, D, E, F]$. We have $|ABDEF| = 10$. Using Equation (8), we obtain:

$$\text{Cost}(q_1) = \frac{150}{100}\left[(3 \times (5+5)) + \left(10 \times \left(1 - \left(1 - \frac{1}{\frac{150\times 10}{100}}\right)^2\right)\right)\right] = 46.95.$$

  The query q_3, which references the attributes A, D and F, is answered using only the fragment $[A, D, F]$. We have:

$$\text{Cost}(q_3) = \frac{150}{100}\left[\left(5 \times \left(1 - \left(1 - \frac{1}{\frac{150\times 5}{100}}\right)^{20}\right)\right)\right] = 7.50.$$

  The query q_2, which does not need joins, is treated in the same manner as q_3 and the queries q_4 and q_5 are treated in the same manner as q_1 because they require additional joins. Table 3 lists the costs for all the considered queries.

- **Using the partitioning scheme $\{[A, B, E], [D, F], [C]\}$:** Similarly, queries costs are summarized in Table 4. Consequently, the partitioning scheme $\{[A, D, F], [B, E], [C]\}$ is recommended.

Query	**Frequency**	**Cost**	**Cost*Frequency**
q_1	1	46.95	46.95
q_2	3	00.46	01.38
q_3	3	07.50	22.50
q_4	2	134.37	268.74
q_5	1	133.02	133.02
Workload cost			**472.59**

Table 3. Queries costs using the first partitioning scheme

Query	**Frequency**	**Cost**	**Cost*Frequency**
q_1	1	01.89	01.89
q_2	3	08.99	26.97
q_3	3	56.25	168.75
q_4	2	134.37	268.74
q_5	1	133.02	133.02
Workload cost			**599.37**

Table 4. Queries costs using the second partitioning scheme

7 EXPERIMENTAL STUDY

7.1 Description of the Experiments

The goal of the experiments is to show the efficiency of the MaxPart approach. Our comparative analysis is quantified in terms of the number of candidate fragments generated and the total number of comparisons needed to generate a possible partitioning scheme. Obviously, the time cost of the partitioning process is proportional to the number of needed comparisons to generate a possible partitioning scheme. We also study the workload cost improvement using the recommended partitioning scheme. The datasets used in our experiments are commonly found in the literature.

The proposed approach improves the one presented in [3]. In order to perform a fair comparison with the cited work, we, naturally, conducted a set of experiments on the same tables. However, given their relatively small size, the used tables are unfortunately only of limited value for the experimental study. We, therefore extend our experiments to another important context: data warehouses. Such decisional databases often manipulate huge amount of data. This extension is used to further demonstrate the capability and effectiveness of MaxPart in partitioning large tables. As in [3], we generate the partitioning schemes using predefined threshold values 20 %, 30 %, 40 %, 50 % and 60 %. To generate the candidate fragments, we have implemented the FPMAX algorithm in Java [53]. All experiments were carried out on a PC with 3.4 GHz Intel® Xenon^{TM} and 1 024 MB of memory running Linux Ubuntu 12.04.

7.2 Experiments on TAE and ADULT Tables

For comparative purposes, the first experiments have been conducted using the same tables, workloads and parameters as in the most closely related work [3].

Table	**# of Attributes**	**# of Tuples**	**Attributes**	
			Attribute	**Size (byte)**
TAE	6	161	A: Speaker	19
			B: Cours_instructor	2
			C: Course	2
			D: Semester	7
			E: Class_size	2
			F: Class_attribute	6
ADULT	15	30 162	A: Age	1
			B: WorkClass	16
			C: Final-weight	4
			D: Education	12
			E: Education-num	1
			F: Marital-status	21
			G: Occupation	17
			H: Relationship	14
			I: Race	18
			J: Sex	6
			K: Capital-gain	3
			L: Capital-loss	2
			M: Hours-per-week	1
			N: Native-country	26
			O: Class	2

Table 5. TAE and ADULT tables characteristics

The proposed approach is tested on two real datasets: Teaching Assistant Evaluation (TAE) and ADULT, which were taken from the UCI Machine Learning Repository [56]. TAE and ADULT are small databases containing 161 and 30 162 tuples respectively. Table 5 lists a summary of the two databases. The used workloads [3] involve 12 and 20 queries for TAE and ADULT datasets respectively. In these experiments, the first cost model, e.g. Equation (8), will be used as a basis to perform our comparisons.

7.2.1 Experiment 1: Computational Study

The first experimental results concerning the number of candidate fragments are shown in Tables 6 and 7. We can note that for most values of threshold, MaxPart significantly reduces the space of candidate fragments. For TAE dataset, the ratio of the number of candidate fragments generated by AllPart to the one generated by MaxPart varies from 1.5 to 17. Using ADULT dataset the improvement is more important. That same ratio varies from 3 to 1 638.

	TAE Dataset		
Threshold (%)	**AllPart**	**MaxPart**	**Reduction Rate**
20	51	3	**94.11 %**
30	29	4	**86.20 %**
40	16	5	**68.75 %**
50	7	5	**28.57 %**
60	2	2	**00.00 %**

Table 6. Number of candidate fragments

Table 8 and Table 9 illustrate the number of comparisons between the fragments to be processed. The performed comparisons are required to generate possible partitioning schemes. As predictable, it could be seen that AllPart needs to perform much more comparisons to achieve this task. In summary, MaxPart shows better performances in terms of computational complexity for low threshold values. The performance rate decreases along with the increase of the threshold value. The reason is that for high values of threshold there are very few or no generated fragments for both AllPart and MaxPart. This leads to almost the same number of candidate fragments. As discussed in Section 2, AllPart is a computationally expensive approach.

	ADULT Dataset		
Threshold (%)	**AllPart**	**MaxPart**	**Reduction Rate**
20	32 767	20	**99.94 %**
30	269	17	**93.68 %**
40	21	7	**66.66 %**
50	5	5	**00.00 %**
60	3	3	**00.00 %**

Table 7. Number of candidate fragments

	TAE Dataset		
Threshold (%)	**AllPart**	**MaxPart**	**Reduction Rate**
20	100	4	**96.00 %**
30	56	6	**89.29 %**
40	30	8	**73.33 %**
50	6	4	**33.33 %**
60	2	2	**00.00 %**

Table 8. Number of comparisons for generating possible partitioning schemes

	ADULT Dataset		
Threshold (%)	**AllPart**	**MaxPart**	**Reduction Rate**
20	622 554	19	**99.99 %**
30	268	16	**94.03 %**
40	40	12	**70.00 %**
50	4	4	**00.00 %**
60	2	2	**00.00 %**

Table 9. Number of comparisons for generating possible partitioning schemes

7.2.2 Experiment 2: Performance Study

The costs of the workload exploiting the produced partitioning schemes compared with the baseline case where no partitioning is performed are shown in Tables 10 and 11, respectively. From Table 10, it could be observed that the best partitioning scheme for TAE dataset is obtained at a threshold of 30 % resulting in an improvement of 11.60 % over unpartitioned scheme. The best partitioning scheme for ADULT dataset is obtained at threshold of 40 % (Table 11) resulting in an improvement of 36.05 % over unpartitioned scheme. It could be seen that, for both datasets, as the threshold value increases, there are many more small fragments. This can be explained by the fact that higher threshold values result in fewer maximal (largest) fragments. In such a case, queries require many fragments, what multiplies join operations resulting in a high workload cost.

Threshold (%)	**Partitioning Scheme**	**Cost**	**Reduction Rate**
20	ABDEF C	2 448	0.32
30	ACDF BE	2 171	11.60
40	ACF BE D	2 420	1.46
50	AF B C D E	2 600	−5.86
60	A B C D E F	2 720	−10.74
Without partitions	ABCDEF	2 456	

Table 10. Best partitioning scheme (TAE dataset)

Threshold (%)	**Partitioning Scheme**	**Cost**	**Reduction Rate**
20	ABCDEFGHIJKLMNO	454 236	0.00
30	ABCEIKM FO DH G J L N	331 520	27.01
40	ABO EFK M G D N C H I J L	290 451	36.05
50	A B C D E F G H I J K L M N O	501 283	−10.35
60	A B C D E F G H I J K L M N O	501 283	−10.35
Without partitions	ABCDEFGHIJKLMNO	454 236	

Table 11. Best partitioning scheme (ADULT dataset)

7.3 Experiments on TPC-H Benchmark

In order to evaluate our approach in data warehouse context, we use as experimental data the TPC-H benchmark with a sequence of 21 queries [57]. The decision-support benchmark TPC-H contains a fact table `Lineitem` (6 000 000 tuples) and 7 dimensions tables: `Orders` (1 500 000 tuples), `Part` (200 000 tuples), `Partsupp` (800 000 tuples) `Supplier` (10 000 tuples), `Customer` (150 000 tuples), `Nation` (25 tuples) and `Region` (5 tuples). Table 12 lists a summary of the fact table attributes.

Table	**# of Attributes**	**# of Tuples**	**Attributes**	
			Attribute	**Size(byte)**
			A: LineNumber	4
			B: Quantity	8
Lineitem	13	6 000 000	C: ExtendedPrice	8
			D: Discount	8
			E: Tax	8
			F: ReturnFlag	1
			G: LineStatus	1
			H: ShipDate	7
			I: CommiDate	7
			J: ReceipDate	7
			K: ShipInStruct	25
			L: ShipMode	10
			M: Comment	44

Table 12. TPC-H fact table characteristics

7.3.1 Experiment 1: Computational Study

Table 13 and Table 14 show respectively the number of candidate fragments and the number of comparisons required to generate possible partitioning schemes for different values of threshold. As it is predictable, we can note that MaxPart significantly reduces the space of candidate fragments. This can be explained by the fact that MaxPart minimizes the cost of enumerating candidate fragments by restricting the output set to only the most relevant fragments. As a consequence (Table 14), AllPart needs to perform much more comparisons to generate possible partitioning schemes.

7.3.2 Experiment 2: Performance Study

In these experiments, the second cost model, e.g. Equation (18), is used as a basis to perform our comparisons. The workload cost using the produced partitioning schemes can be seen in Table 15. The best partitioning scheme is obtained at a threshold value of 30 %. The results clearly show that the performance of queries

	TPC-H		
Threshold (%)	**AllPart**	**MaxPart**	**Reduction Rate**
20	2 621	23	**99.12 %**
30	251	19	**92.43 %**
40	32	6	**81.25 %**
50	7	4	**42.85 %**
60	2	2	**00.00 %**

Table 13. Number of candidate fragments

	TPC-H		
Threshold (%)	**AllPart**	**MaxPart**	**Reduction Rate**
20	100	4	**96.00 %**
30	56	6	**89.29 %**
40	30	8	**73.33 %**
50	6	4	**33.33 %**
60	2	2	**00.00 %**

Table 14. Number of comparisons for generating possible partitioning schemes

is greatly enhanced. The workload cost in approximately half the cost where no partitioning is performed.

Threshold (%)	**Partitioning Scheme**	**Cost**	**Reduction Rate**
20	BCD AEFGHIJKLM	172 849 442.32	39.59
30	BCD FG AEHIJKLM	145 251 636.64	49.23
40	CD ABEFGHIJKLM	306 179 690.01	−6.99
50	CD ABEFGHIJKLM	306 179 690.01	−6.99
60	CD ABEFGHIJKLO	306 179 690.01	−6.99
Without partitions	ABCDEFGHIJKLO	286 149 243.01	

Table 15. Best partitioning scheme (TPC-H)

8 CONCLUSIONS AND PERSPECTIVES

Partitioning is a common method used for improving the performance and the scalability of data bases systems. The database is divided into smaller pieces called partitions. Partitions are then managed independently increasing the system throughput. For a given workload, vertical partitioning in databases is highly dependent on the number of attributes. The number of choices that can be made is very large making the partitioning process quite tedious and difficult even for skilled DBAs. Therefore, there is a practical need for strategies that assist the DBAs in this process.

In this paper we have proposed a maximal frequent itemsets based approach to vertical partitioning. We believe that the input data, available as a workload, can be turned into useful information and knowledge that are previously unknown. In particular, in this work we are dealing with knowledge in the form of maximal frequent itemsets. The information and knowledge gained can be used for identifying an optimized partitioning scheme. We have particularly optimized the partitioning process by the means of the downward-closure property of the set of maximal frequent itemsets which are inherently scalable and far less numerous. We use only an extremely small percentage of the possibly huge search space required by similar approaches.

Experimental study have shown that our approach does not only reduce the search space of the studied problem, but it also improves the system performance. The experiments confirm the theoretical prediction, showing that the performance improvement increases along with the increase of the volume of the data. The workload cost improvement is less noticeable for the smaller dataset (having 161 tuples), but for the other datasets (having 30 162 and 6 000 000 tuples, respectively), the improvement is more important.

The work presented in this paper can be extended in the two following directions. First, the system performance can be improved by coupling vertical partitioning with other optimization techniques such as indexing. It will be interesting to study the impact of combining the two optimization techniques on the system performances. Second, due to the interdependencies between partitioning and distributed query optimization, our approach can easily be extended to distributed, or cloud, context design according to their special requirements. The technique presented in this work would also be beneficial for those systems. Each compute node exploits our technique to split the data that are locally stored. In such cases, our cost model will be updated including network transit fees.

REFERENCES

[1] Agrawal, S.—Narasayya, V.—Yang, B.: Integrating Vertical and Horizontal Partitioning into Automated Physical Database Design. Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, Paris, France, 2004, pp. 359–370, doi: 10.1145/1007568.1007609.

[2] Hammer, M.— Niamir, B.: A Heuristic Approach to Attribute Partitioning. Proceedings of the 1979 ACM SIGMOD International Conference on Management of Data, Boston, Massachusetts, 1979, pp. 93–101, doi: 10.1145/582095.582110.

[3] Gorla, N.—Pang, W. Y. B.: Vertical Fragmentation in Databases Using Data-Mining Technique. In: Taniar, D., Rusu, L. I. (Eds.): Strategic Advancements in Utilizing Data Mining and Warehousing Technologies: New Concepts and Developments. IGI Global, 2010, pp. 178–197.

[4] Ramakrishnan, R.—Gehrke, J.: Database Management Systems. McGraw-Hill, Inc., New York, NY, USA, 2003.

[5] Hoffer, J. A.—Severance, D. G.: The Use of Cluster Analysis in Physical Data Base Design. Proceedings of the 1st International Conference on Very

Large Data Bases (VLDB '75), Framingham, Massachusetts, 1975, pp. 69–86, doi: 10.1145/1282480.1282486.

[6] Navathe, S.—Ceri, S.—Wiederhold, G.—Dou, J.: Vertical Partitioning Algorithms for Database Design. ACM Transactions on Database Systems (TODS), Vol. 9, 1984, No. 4, pp. 680–710, doi: 10.1145/1994.2209.

[7] Cornell, D. W.—Yu, P. S.: A Vertical Partitioning Algorithm for Relational Databases. Proceedings of the Third International Conference on Data Engineering (ICDE), IEEE Computer Society, 1987, pp. 30–35.

[8] Cornell, D. W.—Yu, P. S.: An Effective Approach to Vertical Partitioning for Physical Design of Relational Databases. IEEE Transactions on Software Engineering, Vol. 16, 1990, No. 2, pp. 248–258, doi: 10.1109/32.44388.

[9] Navathe, S. B.—Ra, M.: Vertical Partitioning for Database Design: A Graphical Algorithm. ACM SIGMOD Record, Vol. 18, 1989, No. 2, pp. 440–450, doi: 10.1145/67544.66966.

[10] Song, S. K.—Gorla, N.: A Genetic Algorithm for Vertical Fragmentation and Access Path Selection. The Computer Journal, Vol. 43, 2000, No. 1, pp. 81-93.

[11] Cheng, C. H.—Lee, W. K.—Wong, K. F.: A Genetic Algorithm-Based Clustering Approach for Database Partitioning. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), Vol. 32, 2002, No. 3, pp. 215–230.

[12] Muthuraj, J.—Chakravarthy, S.—Varadarajan, R.—Navathe, S. B.: A Formal Approach to the Vertical Partitioning Problem in Distributed Database Design. Proceedings of the Second International Conference on Parallel and Distributed Information Systems, San Diego, California, USA, 1993, pp. 26–35, doi: 10.1109/PDIS.1993.253076.

[13] March, S. T.—Rho, S.: Allocating Data and Operations to Nodes in Distributed Database Design. IEEE Transactions on Knowledge and Data Engineering, Vol. 7, 1995, No. 2, pp. 305–317.

[14] Bellatreche, L.—Simonet, A.—Simonet, M.: Vertical Fragmentation in Distributed Object Database Systems with Complex Attributes and Methods. Proceedings of 7$^{\text{th}}$ International Conference and Workshop on Database and Expert Systems Applications (DEXA 96), 1996, pp. 15–21, doi: 10.1109/DEXA.1996.558266.

[15] Özsu, M.—Valduriez, P.: Principles of Distributed Database Systems. Prentice-Hall, 1999.

[16] Ezeife, C. I.—Barker, K.: Distributed Object Based Design: Vertical Fragmentation of Classes. Distributed and Parallel Databases, Vol. 6, 1998, No. 4, pp. 317–350.

[17] Barker, K.—Bhar, S.: A Graphical Approach to Allocating Class Fragments in Distributed Object Base Systems. Distributed and Parallel Databases, Vol. 10, 2001, No. 3, pp. 207–239.

[18] Son, J. H.—Kim, M. H.: An Adaptable Vertical Partitioning Method in Distributed Systems. Journal of Systems and Software, Vol. 73, 2004, No. 3, pp. 551–561.

[19] Gorla, N.: An Object-Oriented Database Design for Improved Performance. Data and Knowledge Engineering, Vol. 37, 2001, No. 2, pp. 117–138.

[20] Fung, C. W.—Karlapalem, K.—Li, Q.: An Evaluation of Vertical Class Partitioning for Query Processing in Object-Oriented Databases. IEEE Transactions on Knowledge and Data Engineering, Vol. 14, 2002, No. 5, pp. 1095–1118.

[21] Fung, C. W.—Karlapalem, K.—Li, Q.: Cost-Driven Vertical Class Partitioning for Methods in Object Oriented Databases. The VLDB Journal, Vol. 12, 2003, No. 3, pp. 187–210.

[22] Schewe, K. D.: Fragmentation of Object Oriented and Semistructured Data. Proceedings of the Baltic Conference, BalticDB & IS, 2002, Vol. 1, 2002, pp. 253–266.

[23] Labio, W.—Quass, D.—Adelberg, B.: Physical Database Design for Data Warehouses. Proceedings of the Thirteenth International Conference on Data Engineering, Birmingham, U.K., 1997, pp. 277–288, doi: 10.1109/ICDE.1997.581802.

[24] Golfarelli, M.—Maio, D.—Rizzi, S.: Vertical Fragmentation of Views in Relational Data Warehouses. SEBD, 1999, pp. 19–33.

[25] Furtado, C.—Lima, A. B.—Pacitti, E.—Valduriez, P.—Mattoso, M.: Physical and Virtual Partitioning in OLAP Database Clusters. 17th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD '05), Rio de Janeiro, Brazil, 2005, pp. 143–150, doi: 10.1109/CAHPC.2005.32.

[26] Kling, P.—Özsu, M. T.—Daudjee, K.: Generating Efficient Execution Plans for Vertically Partitioned XML Databases. Proceedings of the VLDB Endowment, Vol. 4, 2010, No. 1, pp. 1–11, doi: 10.14778/1880172.1880173.

[27] Andrade, A.—Ruberg, G.—Baião, F. A.—Braganholo, V. P.—Mattoso, M.: Efficiently Processing XML Queries over Fragmented Repositories with PartiX. Proceedings of the 2006 International Conference on Current Trends in Database Technology (EDBT '06), Munich, Germany, 2006, pp. 150–163, doi: 10.1007/11896548_15.

[28] Mahboubi, H.—Darmont, J.: Data Mining-Based Fragmentation of XML Data Warehouses. Proceedings of the ACM 11th International Workshop on Data Warehousing and OLAP, Napa Valley, California, USA, 2008, pp. 9–16, doi: 10.1145/1458432.1458435.

[29] Bose, S.—Fegaras, L.: XFrag: A Query Processing Framework for Fragmented XML Data. Proceedings of the Eighth International Workshop on the Web and Databases (WebDB 2005), Baltimore, Maryland, USA, collocated with ACM SIGMOD/PODS, 2005, pp. 97–102.

[30] McCormick, W. T.—Schweitzer, P. J.—White, T. W.: Problem Decomposition and Data Reorganisation by a Clustering Technique. Journal of Operations Research, Vol. 20, 1972, No. 5, pp. 993–1009.

[31] Agrawal, R.—Imieliński, T.—Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. ACM SIGMOD Record, Vol. 22, 1993, No. 2, pp. 207–216, doi: 10.1145/170035.170072.

[32] Agrawal, R.—Srikant, R.: Fast Algorithms for Mining Association Rules in Large Databases. Proceedings of the 20th International Conference on Very Large Data Bases (VLDB '94), San Francisco, CA, USA, 1994, pp. 487–499.

[33] Fayyad, U. M.—Piatetsky-Shapiro, G.—Smyth, P.: Knowledge Discovery and Data Mining: Towards a Unifying Framework. Proceedings of the Second Interna-

tional Conference on Knowledge Discovery and Data Mining (KDD '96), AAAI Press, 1996, pp. 82–88.

[34] Yang, Q.—Wu, X.: 10 Challenging Problems in Data Mining Research. International Journal of Information Technology and Decision Making, Vol. 5, 2006, No. 4, pp. 597–604.

[35] Glover, F. W.—Kochnberger, G.: New Optimization Models for Data Mining. International Journal of Information Technology and Decision Making, Vol. 5, 2006, No. 4, pp. 605–609, doi: 10.1142/S0219622006002143.

[36] Han, J.—Cheng, H.—Xin, D.—Yan, X.: Frequent Pattern Mining: Current Status and Future Directions. Data Mining and Knowledge Discovery, Vol. 15, 2007, No. 1, pp. 55–86.

[37] Ganter, B.—Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer-Verlag New York, Inc., 1999.

[38] Martin, B.—Eklund, P. W.: From Concepts to Concept Lattice: A Border Algorithm for Making Covers Explicit. In: Medina, R., Obiedkov, S. (Eds.): Formal Concept Analysis (ICFCA 2008). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 4933, 2008, pp. 78–89.

[39] Pisková, L.—Horváth, T.: Comparing Performance of Formal Concept Analysis and Closed Frequent Itemset Mining Algorithms on Real Data. Proceedings of the Tenth International Conference on Concept Lattices and Their Applications, 2013, pp. 299–304.

[40] Pasquier, N.—Bastide, Y.—Taouil, R.—Lakhal, L.: Efficient Mining of Association Rules Using Closed Itemset Lattices. Information Systems, Vol. 24, 1999, No. 1, pp. 25–46, doi: 10.1016/S0306-4379(99)00003-4.

[41] Zaki, M. J.—Ogihara, M.: Theoretical Foundations of Association Rules. Proceedings of SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 1998, pp. 1–8.

[42] Kuznetsov, S. O.—Obiedkov, S. A.: Comparing Performance of Algorithms for Generating Concept Lattices. Journal of Experimental and Theoretical Artificial Intelligence, Vol. 14, 2002, No. 2-3, pp. 189–216, doi: 10.1080/09528130210164170.

[43] Poelmans, J.—Ignatov, D. I.—Viaene, S.—Dedene, G.—Kuznetsov, S. O.: Text Mining Scientific Papers: A Survey on FCA-Based Information Retrieval Research. In: Perner, P. (Ed.): Advances in Data Mining. Applications and Theoretical Aspects (ICDM 2012). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7377, 2012, pp. 273–287.

[44] Kuznetsov, S. O.—Poelmans, J.: Knowledge Representation and Processing with Formal Concept Analysis. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, Vol. 3, 2013, No. 3, pp. 200–215.

[45] Andrews, S.: A 'Best-of-Breed' Approach for Designing a Fast Algorithm for Computing Fixpoints of Galois Connections. Information Science, Vol. 295, 2015, pp. 633–649, doi: 10.1016/j.ins.2014.10.011.

[46] Goethals, B.—Zaki, M.: An Introduction to Workshop on Frequent Itemset Mining Implementations. Proceeding of the ICDM 03 International Workshop on Frequent Itemset Mining Implementations, 2003, pp. 1–13.

[47] Bayardo Jr., R. J.: Efficiently Mining Long Patterns from Databases. ACM SIGMOD Record, Vol. 27, 1998, No. 2, pp. 85–93.

[48] Agarwal, R. C.—Aggarwal, C. C.—Prasad, V. V. V.: Depth First Generation of Long Patterns. Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '00), 2000, pp. 108–118, doi: 10.1145/347090.347114.

[49] Burdick, D.—Calimlim, M.—Flannick, J.—Gehrke, J.—Yiu, T.: MAFIA: A Maximal Frequent Itemset Algorithm. IEEE Transactions on Knowledge and Data Engineering, Vol. 17, 2005, No. 11, pp. 1490–1504, doi: 10.1109/TKDE.2005.183.

[50] Gouda, K.—Zaki, M. J.: Efficiently Mining Maximal Frequent Itemsets. IEEE International Conference on Data Mining, IEEE Computer Society, 2001.

[51] Grahne, G.—Zhu, J.: High Performance Mining of Maximal Frequent Itemsets. Sixth SIAM International Workshop on High Performance Data Mining, 2003.

[52] Han, J.—Pei, J.—Yin, Y.: Mining Frequent Patterns Without Candidate Generation. ACM SIGMOD Record, Vol. 29, 2000, No. 2, pp. 1–12.

[53] Ziani, B.—Ouinten, Y.: Mining Maximal Frequent Itemsets: A Java Implementation of FPMAX Algorithm. Proceedings of the 6th International Conference on Innovations in Information Technology (IIT), IEEE Press, Piscataway, NJ, USA, 2009, pp. 11–15, doi: 10.1109/IIT.2009.5413790.

[54] Mishra, P.—Eich, M. H.: Join Processing in Relational Databases. ACM Computing Surveys (CSUR), Vol. 24, 1992, No. 1, pp. 63–113, doi: 10.1145/128762.128764.

[55] Yao, S. B.: Approximating Block Accesses in Database Organizations. Communications of the ACM, Vol. 20, 1977, No. 4, pp. 260–261.

[56] UCI (Machine Learning Repository) Web Site. Available at: `http://archive.ics.uci.edu/ml`, University of California, Irvine, School of Information and Computer Sciences.

[57] TPC (Transaction Performance Council) Web Site. Available at: `http://www.tpc.org`.

Benameur Ziani received his Engineer degree in computer science from Sidi Belabbes University (Algeria) and Ph.D. degree in computer science from the University of Laghouat (Algeria). He is currently Associate Professor in computer science at the Department of Computer Science of the University of Laghouat. Prior to joining the Department of Computer Science he served as Engineer in computer science at the computing center of the University of Laghouat during 1992–2012. His current research interests include knowledge discovery, data mining and machine learning with applications in various areas: database and data warehouse design optimisation, big data and data networks analytics.

Youcef Ouinten received his M.Sc. degree and Ph.D. degree in operational research from the University of Southampton (UK), in 1984 and 1988, respectively. He received his graduation degree (Diplôme d'Etudes Superieures) in mathematics, option operational research, from the University of Science and Technology – Houari Boumediene of Algiers, Algeria, in 1981. He served as Head of the Computing Center at the University Amar Telidji of Laghouat, from 1999 to 2012. He is currently Senior Lecturer at the Department of Mathematics and Computer Science of the University Amar Telidji of Laghouat, Algeria. His research interests include data mining, text mining, information retrieval and optimization.

Mustapha Bouakkaz is Associate Professor in computer science at the Department of Computer Science of the University of Laghouat Algeria. He received his Ph.D. degree in computer science from the University of Laghouat in 2017. He carries out research on OLAP and Data Mining. He is more interested about data coming from documents or social networks. His current work focuses on graph OLAP, text mining and social networks analysis.

Computing and Informatics, Vol. 37, 2018, 946–968, doi: 10.4149/cai_2018_4_946

OBJECT MAPPING IN THE OPC-UA PROTOCOL FOR STATICALLY AND DYNAMICALLY TYPED PROGRAMMING LANGUAGES

Piotr P. Nikiel

CERN
CH-1211 Geneva 23
Switzerland
e-mail: `piotr@nikiel.info`

Krzysztof Korcyl

Institute of Nuclear Physics PAN
ul. Radzikowskiego 152
31-342 Kraków, Poland
e-mail: `Krzysztof.Korcyl@ifj.edu.pl`

Abstract. Two or more object-oriented components located in networked computers can form a distributed system to exchange information and execute methods. The most known approaches include object request broker architectures (e.g. CORBA), messaging-service architecture (e.g. based on ZMQ or JMS) or some variant of Service Oriented Architecture (e.g. SOAP). One of new approaches in the field is the OPC-UA protocol. While having common parts with all aforementioned architectures, it brings very rich and extensible information modelling capabilities, versatility and dynamic address space model, among others. This paper proposes a mapping of information model (applicable in the OPC-UA protocol) into class and object structure of an object-oriented programming language. Special attention is paid to whether given programming language is statically or dynamically typed, with examples and applications in C++ for the former case and Python for the latter. The study also covers the cases of using the proposed mapping at both server- and client-side of OPC-UA software.

Keywords: Middle-ware, OPC-UA

Mathematics Subject Classification 2010: 94-A99

1 INTRODUCTION

There are many architectures and software technologies enabling data exchange between software applications. In the software layering stack they are positioned between a software layer devised to send or receive data (e.g. particular end-user software application) and a layer that enables access to the communication medium (e.g. operating system functions providing access to the network protocol stack or a software library implementing HTTP client). Such a glue layer is often termed "middle-ware".

The software technologies, architectures and programming languages evolved over the years and many new concepts became common, for example object orientation, or common usage of programming languages running in virtual machines, to name a few. In parallel there was a shift towards higher-level programming languages, which are less bound to specific execution environment architecture or even completely independent from it. Middle-ware technologies followed the evolution, profiting both from advancements in programming as well as in ubiquity of network-oriented software, thanks to nowadays presence of the Internet almost everywhere.

Today the middle-ware technologies take part in all steps of data exchange, such as:

- supporting varied data types, possibly including nested or otherwise aggregated types, as well as types defined in run-time, and data serialization[1] of these types,
- processing of serialized data such as timestamping, encoding, encryption or checksum verification at the reception,
- ensuring guaranteed, timely and efficient data exchange, possibly notifying when these conditions cannot be met,
- establishing and shutting down communication channels in varied communication patterns (e.g. one-to-one, one-to-many or other).

The consequences of abandoning the middle-ware layer seem to be quite serious. Then some or most of its tasks have to be carried out by the application layer. The obvious consequence is the lack of abstraction which such a layer would provide, so the application layer would have to deal also with e.g. on-the-wire data encoding concerns. Also modularity and portability would be affected, the former one by not being able to rework data exchange mechanisms without touching the application layer while the latter one by potentially being bound to the initially chosen communication medium.

To take a practical example, let us imagine a system of one data publisher and many receivers, for example: a system distributing currency quotes. A simple substitute for middle-ware solution could be e.g. publishing the data in plain

[1] Serialization is a transformation of given data (e.g. a variable in a computer program) to a stream of octets, such that it can be sent over network and restored at a remote program to a representation which is identical to the original data.

text, with currency pairs in the consecutive lines, where each line would have the currency pair identifier and the bid and ask prices, all comma-separated. An interested receiver would connect to the publisher and wait for the delivery of one or more lines of text. This already brings in many open questions starting from the format of numbers (e.g., which decimal separator to use?) and lines (which newline character(s) to use?) up to what happens when a receiver is interested only in one currency pair? Does it have to receive all the pairs and ignore all but the one of the interest?

Even bigger concern arises when the receiver has to automatically process the data. In our example, the plain text is attractive to humans but no so much for automated processing where it is not only redundant but also ambiguous. Assuming some binary format helps to alleviate some issues but not all of them (e.g., when the receiver is interested only in a part of the data). Generally, it is clear that a more generic solution in middle-ware layer would be desired.

An improvement can be achieved when a generic data serialization approach is used. Such an approach typically requires that the data format (often called a protocol) is described beforehand in a supported notation. The description can be then used to govern the behaviour of the data serializer and deserializer (e.g., the serializer knows the offset of a given data field in the serialized message). More importantly, such a description can be used to obtain programming language bindings to data structures which are to be serialized.

There are many examples of such generic data serializers. One of the most known is the Abstract Syntax Notation One (ASN.1) [1], commonly used as a workhorse of many protocols like SNMP [2] and other. The notation used to describe the data format has a well defined syntax covering any data format which bases on primitive data types (Booleans, integers, etc.) or an aggregation of those (sequence, set, etc.). The description written according to the ASN.1 notation can be used to generate bindings (mappings) in many possible programming languages using the ASN.1 compilers, such that a software developer just refers to field names in generated bindings and not to encoding-specific data. Interestingly, the ASN.1 makes a distinction between the notation itself (being a description of exchanged information) and particular encoding types which are specified in different documents [3]. Therefore one notation can be encoded using many possible ways, including binary formats, XML and other.

A second example is a much newer generic data serializer called Google Protocol Buffers [4]. The Google Protocol Buffers defines a message description format called "proto files". The description format lets define message types that an application would use and then generate mappings for a number of supported programming languages. Such a mapping not only provides an entry point for the programming language but also contains everything which is needed to output or input data in a serialized, binary form.

One must mention that some programming languages support serialization as a built-in feature. The Java programming language with its Java Virtual Machine is a notable example [5]. A significant improvement with regard to both examples cited above (the ASN.1 and the Google Protocol Buffers) is that in order to serialize given object, only the information from its class is needed, without any prior preparation of external description of the data format[2]. In practice it has an advantage of being able to exchange information as objects between systems in most pristine way with no additional cost in encoding and serialization. However the downside is that the mechanism is specific to the Java programming language and the usage of such serialized objects from another programming language clearly falls beyond the intended purpose of the mechanism.

All three examples shown above illustrate how to interface an important part of middle-ware layer – data serialization and encoding – to a programming language of choice. However nothing was said yet on factual transport of serialized data from one system to another.

Message oriented middle-ware (often abbreviated as MOM) is one of the most common paradigms used in the data exchange between systems. The bottom line of message orientation is that the visible interface from the application layer is expressed in terms of messages having properties like source identifier, destination identifier(s), validity, priority, persistence settings and among others, payload. The payload is where the actual information is to be placed, and most generally it can be seen as an array of octets(bytes). The payload is where higher level data is supposed to be placed after serialization. Therefore combining the aforementioned data serializers with a message oriented middle-ware form a powerful combination which can transport high level data from one application to another while hiding away details of network technology or data encoding concerns.

A common feature of message oriented middle-ware is that it supports many communication patterns like one-to-one, one-to-many, many-to-many, and others. As a consequence, many data exchange problems can be solved efficiently, e.g. in the aforementioned example of currency quotes distribution, using one-to-many communication pattern, only one message publication would be sufficient to update all interested receivers.

There are many notable examples of message oriented middle-ware. One can imagine using UDP/IP datagrams through the means of socket API (e.g. BSD sockets) with some data serializer as a very basic approach. In the recent years, ZeroMQ [6] gained wide interest as a general purpose distributed messaging library, with a focus on the performance and support for diversified communication patterns. When coupled with the aforementioned Google Protocol Buffers (or another general purpose data serializer), the two offer a powerful message-oriented data exchange solution. In Java-related technologies, the state-of-the-art approach uses the Java Message Service [7], abbreviated as JMS. The JMS itself stays as an abstraction layer on concrete distributed messaging implementation, with support for many open-source and commercial messaging solutions. Taking into account Java's

[2] The class must implement java.io.Serializable interface so that this mechanism could work.

support for built-in serialization, the JMS delivers a straighforward approach for exchanging high level data (Java objects) out-of-the-box.

Message oriented middle-ware is not the only paradigm for data exchange between software systems. Object request broker architectures (abbreviated ORB) take a completely different approach, often termed "distributed object" paradigm. The primary difference is that the ORB approach looks at the problem from the perspective of (remotely placed) objects rather than from the perspective of message circulation. The basic role of an ORB is letting interaction between the objects (e.g. calling their methods) no matter where they are - such a feature is called "location transparency". To illustrate how this paradigm might be used for the data exchange between software systems, we can imagine an object A in a system that wants to share data and an object B in a system that wants to obtain the data. In such a configuration, the object B might ("remotely") execute a method "getData()" on the remote object A, obtaining the data as a value returned from such a call.

One of the most notable examples of ORB architectures is the Common Object Request Broker Architecture [8], often abbreviated "CORBA". CORBA requires that interfaces of the interacting objects are specified in an Interface Definition Language, abbreviated IDL. A source file in IDL language is then passed to IDL compiler which generates stub and skeleton code[3] in a chosen programming language. The generated stub, when called upon, is able to pass the call request to a remote object where the factual implementation gets executed.

One of the primary deficits of CORBA as a data exchange solution is that it is focused around objects interaction (i.e. passing method invocations remotely) and not around data exchange per se. It is easy to notice CORBA advantages when it comes to one-to-one, synchronous communication. However many data exchange problems require one-to-many communication which although possible, does not fit CORBA architecture well. Publish-subscribe communication pattern of course can be implemented in CORBA (e.g. following object-oriented publish-subscribe design pattern, also called observer design pattern [9]), but its asynchronous execution (not to slow down the publisher by clients) requires advanced CORBA mechanisms and complex designs. In these factors it is inferior to message-oriented solutions.

WebServices is a more recent paradigm, covering a big number of specific protocols, technologies and software products, like XML-RPC or SOAP. Their common part is profiting from the design choices that were (and still are) responsible for rapid growth of the internet: HTTP, text-based data encoding like XML or JSON and openness of standards. Using XML-RPC or SOAP with a language binding compiler in fact enables to replicate the distributed-object paradigm, known e.g. from CORBA. The XML and JSON allow data exchange with unlimited structuring features, though the overhead of text based data to its binary counterpart cannot be ignored. It is one of the reasons for which better suited protocols are cho-

[3] In CORBA terminology, stub is the caller's interface while skeleton is the callee's interface.

sen when performance or smaller foot-print (e.g. for embedded applications) matter.

One of the newest additions to the catalogue of the data exchange solutions is the OPC Unified Architecture, abbreviated as OPC-UA [10]. OPC-UA is a standard covering wide span of aspects of data exchange between software applications, including support for extensible and rich information models, notifications, support for rich meta-data and multiple encoding manners (including binary and XML).

Rich information model of OPC-UA, covering well beyond object oriented semantics, is especially attractive for its application in data exchange software written in high-level object-oriented programming language. Therefore it is interesting to study possible mappings and bindings between application's classes and objects and their representations exposed to remote systems communicated using OPC-UA. Some inspiration to this study comes from CORBA architecture as well as from other types of mappings in object-oriented programming languages, e.g. from object-relational mappings for database interfacing or from XSD-CXX project providing XML Schema to C++ data binding compiler [11]. The work presented in this paper has its roots in the previously published research on model-based generation of OPC-UA servers [12, 13].

The primary application of the mappings studied in the paper is for a distributed control system of the ATLAS Experiment, one of experiments at the Large Hadron Collider at CERN in Geneva, Switzerland. The OPC-UA is used in the ATLAS Experiment for the data exchange between software components deployed on nodes of the distributed system. The system is currently composed of about 150 machines running Linux; the majority of them run at least one OPC-UA server delivering data obtained from varied types of hardware. A particular challenge of such a system is that it integrates numerous types of data sources; each data source type has different interface (i.e. schema of published information), different amount of data published per unit of time and might be implemented in a different software technology (many of the integrated components are made by external contributors or external companies). The OPC-UA has been selected as the protocol of choice for component integration because of its high-level object-oriented information model and wide adoption in industry; at the same time it enables to establish much weaker coupling between systems than e.g. CORBA, enabling to easily integrate diversified nodes and providing a lot of flexibility.

The mapping approaches studied in the paper attempt to fill the gap between high-level object-oriented programming languages and middle- ware protocols with rich information modelling capabilities. The study focuses on implementation of mapping known from the ORB-like architecture to the new standard OPC-UA. The additional novelty is that the process of mapping is carried out fully dynamically (if permitted by the programming language), without a priori knowledge of exposed object interfaces and type definitions. Few examples of object-oriented systems are given for considerations of practical aspects of proposed approaches.

2 OPC-UA INFORMATION MODEL AND ITS OBJECT-ORIENTED ASPECTS

Information exposed by an OPC-UA server is organized as a graph [10]. The vertices of the graph are called nodes while the edges are called references. Each node is assigned a type, one of: Object, ObjectType, Method, Variable, VariableType, ReferenceType, View and it also has an address. There are some built-in nodes, e.g. the root node symbolizing an entry point to the OPC-UA address space exposed by the server.

A reference (being graph's edge) may be placed only between two nodes. Each reference has a type, either one of built-in OPC-UA types (e.g. HasComponent, HasTypeDefinition) or a custom reference type.

Let us take a simple example of an OPC-UA address-space exposing just one object called "sensor1" of a class "Sensor", having two fields: "id" and "value" and a method "calibrate". Such an address-space would have graph representation according to the OPC-UA information model as in the Figure 1.

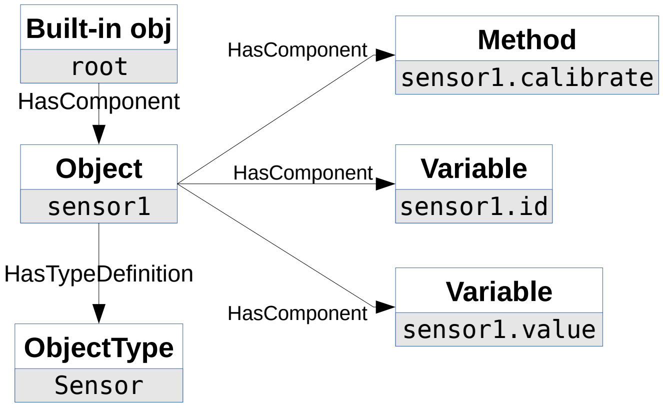


Figure 1. Graph representing OPC-UA address-space described as Example 1

In an OPC-UA server exposing such information model, the graph is stored in the server's OPC-UA address-space. A primary way to obtain the graph by an OPC-UA client is to invoke the "browse request", which takes an address of a node and returns all references originating from the node, including their types and addresses pointed to by them. Therefore, by applying one of the common graph search algorithms (e.g. the breadth-first search or the depth-first search [14]) and starting from the built-in root node, an interested client can discover the whole graph.

However, it is not necessary to know the whole graph to just invoke OPC-UA operations on a remote system: the node address in the address-space is sufficient to uniquely identify any node.

Following operations (also known as transactions) can be invoked on nodes of the address-space[4]:

1. Read and Write of given property of a given node: all nodes support common properties like "description" or "localized name". In addition, variables support "value" property which refers to data stored by a variable.
2. Call: which is a way to invoke methods.
3. Begin, Modify and Stop Monitoring: which implement publish-subscribe functionality in the OPC-UA and let a client be notified about new data.
4. Browse, as explained above.

3 OPC-UA OBJECT MAPPING TO PROGRAMMING LANGUAGE OBJECTS

3.1 OPC-UA Mapping for Distributed-Object Paradigm

The mapping compatible with the distributed-object paradigm is attractive because of its inherent object-oriented properties and proven track record of Object Request Broker architectures (as in e.g. CORBA).

A primary feature of such a mapping should be that an object residing in server application, having a particular interface (methods), could be used through a proxy object residing in client's application. Therefore each invocation of a method should be routed using OPC-UA to a factual remote implementation at server side, as illustrated in Figure 2.

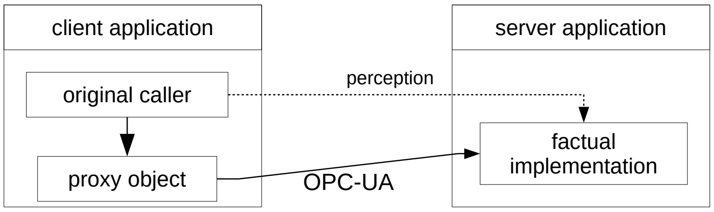


Figure 2. Distributed-object in OPC-UA: factual route of the call operation (solid line) and client-side perception (dotted line)

However, compared to many ORB architectures like CORBA, the OPC-UA information model (including interfaces of objects exposed through the address space) is just a graph stored in the server's memory, and there is no restriction to add or delete new vertices (nodes) or edges (references) in the runtime. The OPC-UA itself does not impose any restriction on requiring the model to be known at compile

[4] The list is not complete, it just enumerates operations attractive in the scope of the study.

time. On the other hand, many commonly used programming languages require type definitions (including interfaces of objects, that is classes) to be known at compile time. Therefore we can speak of many scenarios depending on features of the implementation programming language (separately at client and server side) and chosen mapping approach.

It is worth to emphasize that in this analysis for every high-level class (e.g. "Sensor" from the example in Figure 1) three different type definitions are considered:

- the type definition in the OPC-UA information model, which is stored in the server's address space and can be changed at runtime,
- the type definition for the proxy object at client side, its requirements depend on the chosen programming language,
- the type definition for the factual implementation at server side, its requirements depend on the chosen programming language.

There are three relevant traits to be observed in case of C++ as the implementation language either for client or server side:

- storing a reference to an object (whether achieved by pointers or by C++ reference) does not need a class definition (a forward declaration or opaque-pointer technique can be used),
- calling a method requires the class definition to be known at compile time,
- C++ does not support reflection[5].

Therefore, in C++ implementation at the server side, the type definition for the information model cannot be deduced in run-time (no reflection), and it must be given, along with the source code, with the type definition for factual implementation's class. Similarly, at client side, the proxy object definition must be equivalent to the type definition for the information model at the server side, so both must be given at compile time. C++ implementation brings one more requirement: the type definition in the OPC-UA information model must be constant, otherwise incosistency might arise between all three type definitions.

Java shares many similarities with C++ in the context of the analysis, but it brings an important advantage. Java supports reflection, which in Java not only lets the code to inspect class definitions but also lets invoke the methods discovered using reflection mechanism [15]. Practically it means that a Java object given at runtime could be used as the factual implementation and then mapped as a remote object using OPC-UA. However, the opposite is not straight-forward: creation of a Java class solely on the information from the dynamic information model.[6]

[5] Reflection is a feature of a programming language letting the executed code to inspect itself, e.g. to get a list of methods declared in a given class at run-time.

[6] One could imagine an approach based on bytecode manipulation libraries, however it is not a straight-forward and common solution.

A big difference is achieved when a dynamically-typed programming language is chosen for the implementation. Let us consider Python as an example of such language.

Python enables a number of very attractive features relevant to the study. The first one is the support for a custom definition for accessor methods for object's fields, which enables to intercept get and set operations on the fields ([16] on operator overloading). The methods have standardized names, `__getattr__` and `__setattr__`, respectively. Though having to be defined before the interpreter or compiler is run, they can use run-time data, effectively imitating that the object's list of fields and their types can be altered at runtime. In addition, the act of accessing field's data can be fully customized and refer to the data external to the object, i.e. imitating field access to a remote object. Such application will be further studied in Section 3.2.

The second feature is the support to intercept invocations to call any callable object through a method called `__call__`. When a custom implementation is provided, the call might redirect to a remote invocation of a method through OPC-UA, among other applications.

Let us illustrate the possibilities of combining usage of `__getattr__` and `__call__` in an example as in the UML diagram in Figure 3.

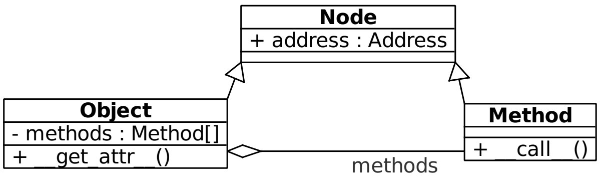


Figure 3. Classes (and their relations) providing remote method invocation capabilities over OPC-UA in Python for distributed-object paradigm

The intention of the example is to illustrate how an object request broking could be facilitated by these features. An example system has an OPC-UA address space with an object called "anObject" with a method called "testMethod". At the client side, in Python program, according to the UML diagram, we construct two objects: "obj" of class "Object" and "method" of class "Method". We insert "method" object into "methods" list in the object. Since both objects are descendants of "Node" class, both have an "address" field. The address of "obj" and "method" point to aforementioned OPC-UA address space nodes "anObject" and "anObject.testMethod".

The aim of the example is to show that invoking "obj.testMethod()" can be given an implementation which performs an OPC-UA transaction invoking respective methods on the server-side.

The invocation of "obj.testMethod()" will first call `__getattr__` method of "obj" object with an argument of "testMethod". A provided custom implementation will intercept the call and identify that requested object's field "testMethod" is in fact a method (it is in "methods" list) and return its reference, turning it into invocation equivalent to: "testMethod()". In the next step, `__call__` method of "Method" class will be invoked, this time its provided implementation will perform an OPC-UA call transaction on the remote server and return its result. As can be seen, the caller might not even know that the call is effectively handled by a remote method, which is the purpose of the location-transparency feature of distributed-object paradigm.

3.2 OPC-UA Mapping for Data Access

In a similar fashion to mapping object's methods remotely, one could imagine mapping object's fields to OPC-UA variables bound to a particular object.

Let us consider the address-space from example in Figure 1, excluding methods, for C++ programming language. In C++ (or, with minimal changes, Java), we could naively depict an equivalent of "Sensor" class with its instance:

```
class Sensor
{
  public:
  string id;
  double value;
};

Sensor sensor1;
```

However in C++ object's fields are just memory locations and their access (either read or write) cannot do anything else than to access the memory locations. Therefore some additional support is required to synchronize object's fields with their OPC-UA counterpart.

The first approach to consider could be called a snapshot approach, where a copy of data stored in remote object's variables is put on object's user request to local object fields. The implementation synchronizing the contents could be provided either as object's method or as an external function. Such implementation would use OPC-UA synchronous read transaction.

Another possible approach is to use publish-subscribe mechanism in OPC-UA where each notification (carrying new updates) takes care of copying the received data to object's fields. The object's user obtains the data without prior request, behind the scenes. Such configuration would have downsides though, especially because of possible race conditions between two threads – the object's user's thread and the updating thread. Addressing the race condition by a mutex would no longer make it look like behind the scenes update.

A different mapping can be studied for C++, which would map OPC-UA object fields into accessor methods (set, get or both, depending on the information model) rather than to C++ object's fields. Such mapping would drift away from the most obvious class declaration shown in the listing above, however it would be free from issues of the previous approach:

```
class Sensor
{
  public:
  string getId ();
  double getValue ();
};

Sensor sensor1;
```

It is worth noting that the approach can be applied to server-side and client-side C++ code. Particular examples are detailed in the Section 4.

Similarly to the conclusions drawn in Section 3.1, moving to a more dynamic programming language like Python has benefits also for data-access mapping. Aforementioned `__getattr__` and `__setattr__` methods can be used to intercept access to fields of a Python object, respectively returning or setting the value from/to a remote OPC-UA address-space. To illustrate this, let us again look at the "Sensor" object example (example in Figure 1). When a suitable implementation is provided for the `__getattr__` method, the following statement can invoke a read operation to fetch the remote value and return it like from a plain local object: `sensor1.value`.

When studying mapping for data access, differences in data types between chosen programming language and the OPC-UA types need to be solved. OPC-UA supports many built-in types which have direct equivalents in common programming languages, like integers of varied bit length, floats, Booleans or strings. N-dimensional arrays of those data types are natively supported without need to create custom types. A special "opaque" data type called variant is also supported which can hold any of the built-in primitive types. Therefore no additional work is expected to profit from built-in types in mapped objects in programming language of choice.

However, OPC-UA supports additional ways to define data types which require the type definition to be first available in the address-space. Among them there are so called "simple types" (effectively one of built-in types with a custom name), "enumerations" (resembling "enums" known from C or C++) and "structured data types". A structured data type resembles structures known from the C programming language. In the context of OPC-UA applications, it enables atomic transfer of the whole structure thus guaranteeing that transport layer events like message fragmentation or buffer size restrictions will not deliver an incomplete snapshot of data. This comes at a price however: a structured data type requires that encoding definition (either for binary or XML serialization) is supplied by the application and stored in the address-space. Thus a mapping for data access involving structures requires that in-application serializer is created (using code generation or dynamic language features or differently).

4 IMPLEMENTATION AND ITS PRACTICAL ASPECTS

So far, the paper explained various conceptual aspects of OPC-UA object mapping for object-request-broker approach and for generic data-access approach, for server side and client side, taking into account relevant features of programming language. This chapter shows how the concepts were implemented.

This paper has roots in the Quasar project, which is a model-based development environment for rapid generation of OPC-UA servers in C++ [12, 13]. A distinctive feature of Quasar relevant to this paper is that Quasar uses a server-side object mapping in the C++ programming language for the data access, e.g. to map read, write operations directly into objects as well as to support the publish-subscribe mechanism in the same approach. As shown in the Section 3.1, for such a configuration, the objects' definition must be known at compile-time. Quasar achieves this by generating classes in C++ from the information model stored in an XML file. The same information model is then stored in the exposed OPC-UA address space and protected from being changed at the runtime, for consistency between C++ classes and the information model.

This study extends Quasar's functionality by adding support for methods. The implementation follows Quasar concepts, that is: methods are declared in the Design file (which is a file primarily storing the information model in the XML format); server build process then generates stubs for implementation and required glue logic. As a result, OPC-UA client request to call a method is mapped in 1:1 relation to a method definition provided in C++ at server side.

To evaluate practical aspects of the mapping for clients in C++, authors have studied a stub generation approach re-using much of Quasar's code generation. As an input, a Design file, in Quasar format is required. For a chosen class (out of all classes defined in the Design file), a proxy class can be generated, wrapping fields access and method invocations into OPC-UA transactions. This effectively creates a companion code for any situation where a client in C++ needs to access data from an OPC-UA server created in Quasar. In this mapping, however, methods in OPC-UA have direct equivalent in the generated stubs while variables have accessor methods instead of fields. In authors opinion it is a most robust approach knowing that C++ does not support overloading field access operator.

For Python programming language at client side, authors have created a library called UaObjects. The purpose of the library is to profit from dynamic nature of Python and let instantiate object mappings to OPC-UA at run-time for clients, both for methods and for data access (read, write, monitored items). The UaObjects extensively uses the approach of overloading `__getattr__`, `__setattr__` and `__call__` methods. PyUaf library [17] is used for OPC-UA interfacing.

The UaObjects library exposes a couple of classes as the API:

- Session – represents an open connection to OPC-UA server

- Node – represents a node in the OPC-UA address-space, stores OPC-UA node address.
- Object – a specialization of Node class representing an object in the address-space.
- Variable – a specialization of Node class representing a variable in the address-space.
- Method – a specialization of Node class representing a method.

The programmer's entry point to the UaObjects library is by instantiation of a Session, supplying PyUaf client handle and server's URI:

```
session = uao.Session(client, server_uri)
```

A primary feature of the Session object is that it can be given an OPC-UA address pointing to an OPC-UA address-space object and perform the object mapping. Such feature is achieved by calling a method `get_object`, e.g.:

```
obj = session.get_object('anObject', 2) # 2 is namespace index
```

The mapping is performed recursively by walking the address-space graph using the Depth-First-Search graph algorithm. As an effect, the whole hierarchy of descendant objects becomes accessible from Python. Thanks to this, a hierarchy of objects can be "walked" like nested classes with no further calls to UaObjects functions.

Let us illustrate the process by an example from a building automation domain. Figure 4 shows a graph of address-space information model.

When `get_object` method is invoked with an address of e.g. "airConditioner" object, the following would happen:

- Browse request is executed on "airConditioner" object, returning three references: one method, one object reference and one variable; they get stored in the object being created.
- The process is recursively repeated for all discovered objects.

Few examples of possible uses of objects mapped by UaObjects library are:

- `airConditioner.turnOn()` will invoke the method over OPC-UA,
- `airConditioner.temperatureSetting = Float(22.5)` will invoke a write operation, any error will be thrown,
- `airConditioner.temperatureSensor.temperature` will invoke a read operation over OPC-UA and return the obtained value or None, or throw for an error.

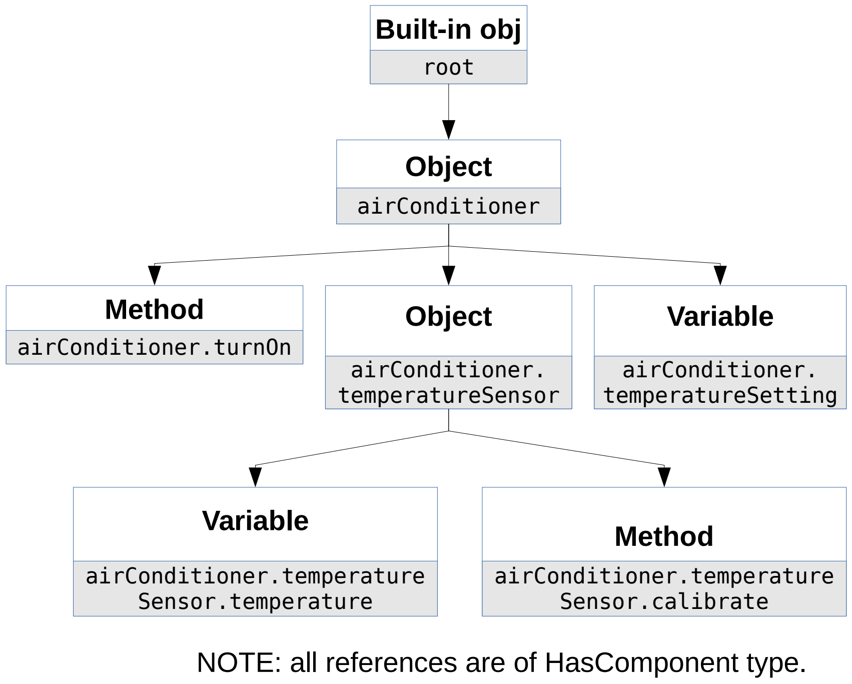


Figure 4. An example OPC-UA information model to illustrate UaObjects library features. The text in greyed fields corresponds to OPC-UA addresses of the nodes (apart from "root" address, which in the OPC-UA is encoded as a numeric identifier, here shown as a string, for simplicity).

5 PERFORMANCE CONSIDERATIONS

Performance is usually an important factor in taking a decision whether to use or to avoid given protocol or solution. Authors have measured the performance of various OPC-UA based approaches studied in this paper and compared it against CORBA and ZMQ/Protocol Buffers under most fair conditions. The primary goal was to measure the time of operation comprising passing payload of given size from either client to server or server to client (both directions were tested separately). Since solely synchronous operations were tested (thus obtained time is the complete duration of the whole operation), obtained operation durations have been used to calculate throughput.

The factual ways of passing data depended on the protocol under test:

- For OPC-UA, write and read operations were under test as well as calling methods either taking arguments (to pass data from a client to a server) or returning values (to pass data from a server to a client). In all situations, a server under test was created with Quasar, with either UASDK or open62541 used as

OPC-UA communication stacks (backends)[7]. As a client, a C++ based client was used (with object mappings generated at compile-time, as described in the paper) or a Python based client with object mappings created in run-time (using UaObjects library, as described in the paper).

- For ZMQ+Google Protocol Buffers, a protocol description file has been prepared and then compiled with the "proto" compiler for C++. The protocol description file containted just one message type consisting of an operation selector (i.e. put data, get data) and the data. For client to server data transfer, a message with "put data" operation with a payload would be sent, and a confirmation message with no payload would be returned. For server to client data transfer, a message with "get data" operation with no payload would be sent, and a confirmation message with payload would be returned. For the transport mechanism ZMQ's request-reply was chosen on top of TCP/IP.
- For CORBA, an interface description file has been prepared and then compiled with the IDL-C++ compiler. OmniORB 4.1.6-2 has been used as the CORBA provider at both client and server side with GIOP as the protocol. Just one interface has been used, taking a string as an argument and returning a string, depending on chosen operation ("get data" or "put data"), a payload would be put in either arguments (for client to server data transfer) or in the return values (for server to client data transfer).

The common configuration of all tests were:

- 1 server and 1 client computer; the server was an 8-core Intel i7 machine at 3.6 GHz clock with 24 GB of RAM, the client was an 4-core Intel i7 machine at 2.4 GHz clock with 8 GB of RAM. Both machines ran Linux operating system;
- physical network connection: Gigabit Ethernet, direct cable connection between both machines. There were no other active network connections;
- request-reply communication pattern was used;
- end-user payload size was considered as the sweeped parameter;
- all test programs were compiled at best optimization levels available;
- all test programs verified size of data at reception with the value configured for given test; an inconsistency would have aborted the test.

The durations of operations are presented in the Figure 5.
The following conclusions could be made from both figures:

- For very small payload sizes (up to 32 B), three clusters of results are visible: ZMQ-based results as clear winners with operations durations of about 100 μs; remaining solutions based on C++ (including CORBA and OPC-UA with C++ clients) with operations durations of about 300 μs; OPC-UA results with client in Python with durations of about 600 μs. Since the only difference between the

[7] At the time of writing, methods support was available only with the UASDK backend.

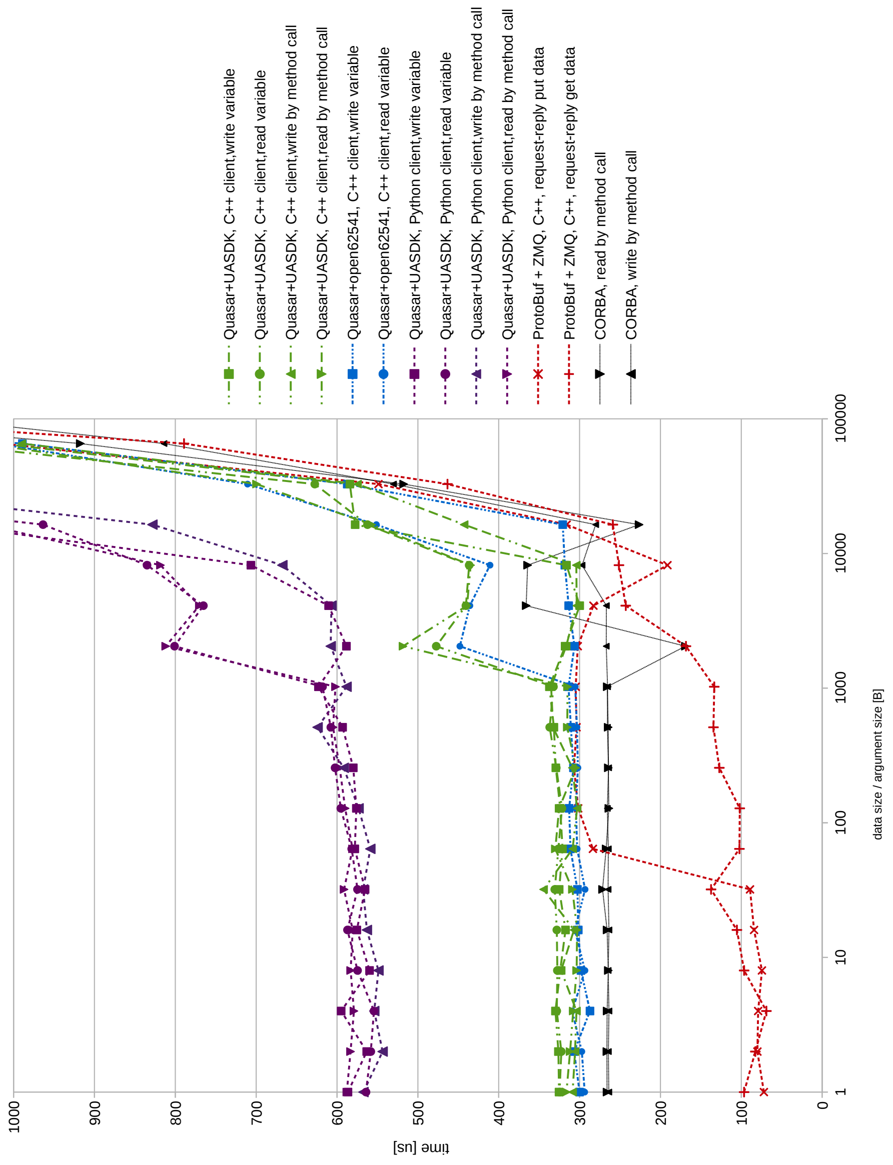


Figure 5. Time required to complete one operation of a given type vs size of payload data. Note X axis is logarithmic and within the range of 1 B–100 kB to emphasize the lower end of the measurements.

second and the third group of results is the programming language (C++ vs. Python), we assume it is the primary factor contributing to the results and not the protocol itself.

- For payload sizes from 64B to 1024B, ZMQ's "put data" implementation joins the second group of results while "get data" implementation still stays ahead by at least twice shorter operation time. The origin of the asymmetry is not clear. Apart from this, observations from the point above still hold true.
- For payload sizes bigger than 1 024 B network related delay becomes the primary contributor. It is worth noticing that apart from Python implementation, all other results converge, however ZMQ-based implementation to transfer data from server to client is still the fastest one.

Based on the results, throughput figures were charted in the Figures 6 and 7. Please note that for chosen communication medium (Gigabit Ethernet) the maximum throughput of the medium itself is about 125 MB/s[8] and in fact, for big payload sizes, the throughput figures converge to about 80–85 % of the medium throughput for all C++ implementations of all 3 chosen protocols.

The following overall conclusions can be made:

- For a data exchange application based on "distributed object" principles (calling remote methods to send/receive data), CORBA and OPC-UA have similar performance. Significant differences apply to both protocols in aspects different from the performance.
- For applications where latency matters most ZMQ surpasses both CORBA and OPC-UA by a very significant factor.

Please note that measurements described in the chapter focused on synchronous request-reply communication patterns. Some of the measured solutions (e.g. ZMQ) offer truly asynchronous data transfers (e.g. ZMQ's Pub-Sub pattern) which, however, have completely different principle of operation and different properties. For example, a subscriber has no control of the rate at which it is receiving data; also error handling is at a different level, e.g., a subscriber might not be getting any data because of either publisher does not have any new data or there is a network issue. Therefore ZMQ's asynchronous communication patterns have not been included in the comparison.

6 CONCLUSIONS

In the paper, authors have studied application of rich information modelling features of OPC-UA to mapping classes and objects defined in OPC-UA address-space into

[8] This, of course, is a rough simplification: on top of the medium, Ethernet frame headers, IP packet headers and TCP headers contribute to what amount of bandwidth is left for payload data.

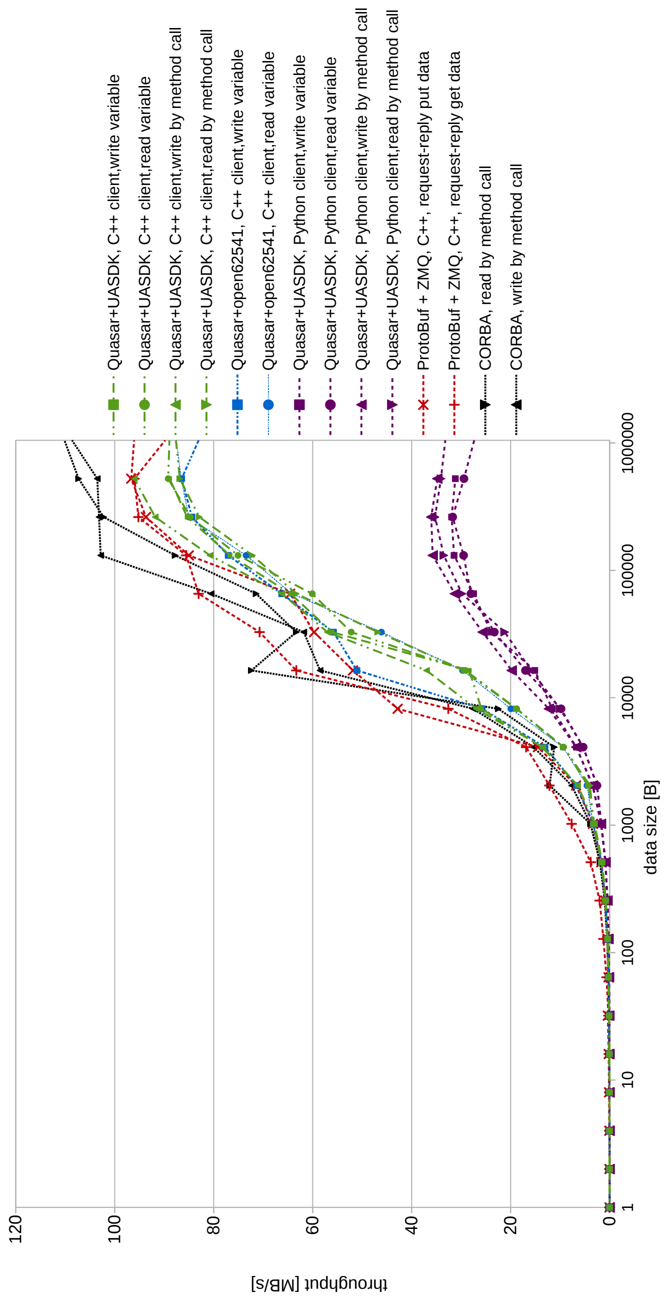


Figure 6. Throughput expressed in payload data size per second as a function of the payload size. The X axis is logarithmic.

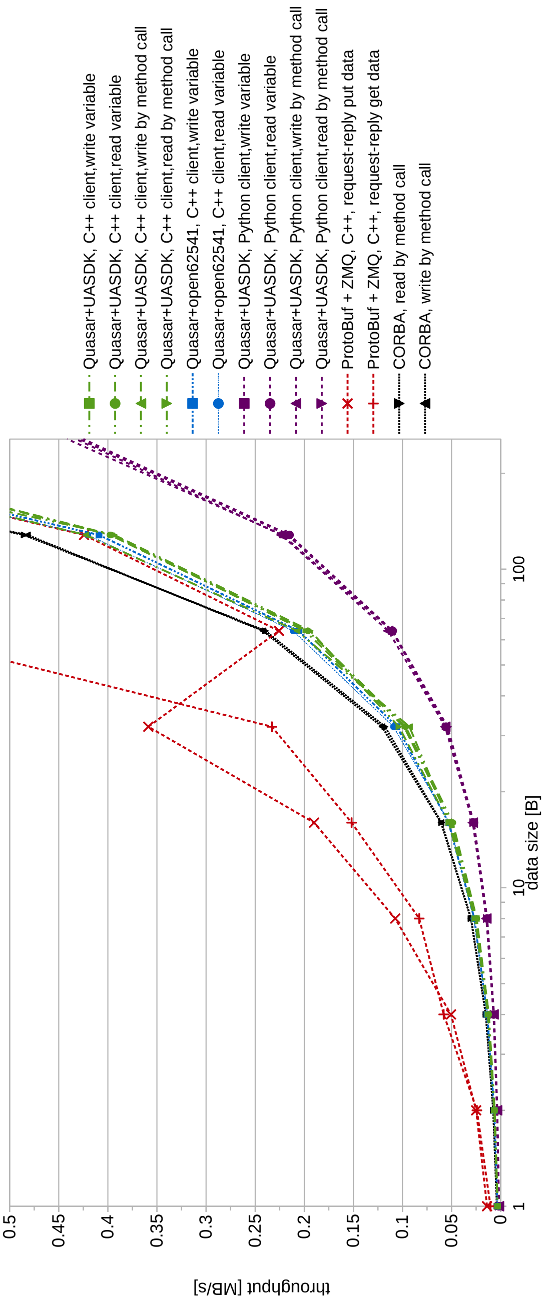


Figure 7. Throughput expressed in payload data size per second as a function of the payload size. The X axis is logarithmic and within range of 1 B–256 B to illustrate the lower range of payloads. The source data is the same as in the Figure 6.

classes and objects of chosen programming language. A number of conclusions have been identified:

- Strongly-typed programming languages (as C++ in the study) enable a direct method mapping to OPC-UA methods, however direct fields mapping has a number of practical down-sides. Authors identify that indirect mapping using accessor methods is a more robust solution.
- Weakly-typed and dynamic programming languages (Python in the study) enable direct mapping of both methods and fields to their OPC-UA counterparts. In addition, a number of "convenience" features is possible like run-time type conversion and possibility to walk the OPC-UA address-space graph by just refering to object fields.
- Languages with support for reflection enable to create the corresponding OPC-UA information model based on programming language class.

The following deliverables are additional effects of the study:

- methods support for Quasar environment,
- UaObjects library for Python,
- code generator enabling client-side mappings to OPC-UA for C++.

7 FURTHER STUDY

Authors would like to point out that the OPC-UA gives a possibility to modify the address-space by client requests[9]. Such feature opens further possible improvements:

- Adding or deleting object instances in run-time by a client might be attractive in many scenarios. For example, it might be more practical to configure the address-space contents by a client than by changing server's configuration. Such scenario easily fits to all cases studied in the paper because it does not change (or add or delete) types definitions.
- Adding, deleting or altering type definitions in run-time by clients would not be compatible at least with C++ mappings studied in the paper, however, there is no obvious obstacle for such application in Python.

One could also imagine an extension of the UaObjects library towards Python's meta-classes concept. In such approach, creation of Python's classes in run-time based solely on information from OPC-UA information model could be attempted.

[9] In a multi-user networked environment, it seems natural that such a feature should be restricted only to the clients having certain (i.e. elevated) privileges. In the OPC-UA this restriction is implemented by requesting authentication when a session (i.e. an OPC-UA connection) is created, for example by using the username-password authentication. Then the server will allow only certain users to modify the address-space, for example by checking whether the username bound to the session through which such request comes is in the set of users with elevated privileges.

REFERENCES

[1] International Telecommunication Union: Information Technology – Abstract Syntax Notation One (ASN.1): Specification of Basic Notation, Recommendation ITU-T X.680, also known as ISO 8824-1.

[2] Douglas, M.—Schmidt, K.: Essential SNMP. 2nd ed., O'Reilly, 2009. ISBN 978-0-596-00840-6.

[3] International Telecommunication Union: Information Technology – ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER), Recommendation ITU-T X.690, also known as ISO 8825-1.

[4] Google: Protocol Buffers, Website, `https://developers.google.com/protocol-buffers/`, accessed 27-Dec-2016.

[5] Java Object Serialization Specification, Java SE v.8, `https://docs.oracle.com/javase/8/docs/technotes/guides/serialization/index.html`.

[6] Hintjens, P.: ZeroMQ: Messaging for Many Applications. O'Reilly, 2013. ISBN 978-1-449-33406-2.

[7] Monson-Haefel, R.—Chappell, D. A.: Java Message Service. O'Reilly, 2001. ISBN 978-0-596-00068-5.

[8] Henning, M.—Vinoski, S.: Advanced CORBA Programming with C++. Addison-Wesley Professional Computing Series, 1999. ISBN 978-0-201-37927-9.

[9] Gamma, E.—Helm, R.—Johnson, R.—Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994. ISBN 978-0-201-63361-2.

[10] Mahnke, W.—Leitner, S.-H.—Damm, M.: OPC Unified Architecture. Springer-Verlag, 2009. ISBN 978-3-540-68898-3, doi: 10.1007/978-3-540-68899-0.

[11] Code Synthesis Tools CC: C++/Tree Mapping Getting Started Guide, available at: `http://www.codesynthesis.com/projects/xsd/documentation/cxx/tree/guide/`.

[12] Nikiel, P. P.—Farnham, B.—Filimonov, V.—Schlenker, S.: Generic OPC-UA Server Framework. Proceedings of 21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015), Okinawa, Japan, 2015. Journal of Physics: Conference Series, Vol. 664, 2015, Art. No. 082039.

[13] Nikiel, P. P.—Farnham, B.—Schlenker, S.—Soare, C.-V.—Filimonov, V.—Abalo Miron, D.: Quasar – A Generic Framework for Rapid Development of OPC-UA Servers. Proceedings of ICALEPCS 2015, Melbourne, Australia, 2015.

[14] Cormen, T. H.—Leiserson, C. E.—Rivest, R. L.—Stein, C.: Introduction to Algorithms. MIT Press, 2009. ISBN 978-0-262-03384-8.

[15] Horstmann, C. S.—Cornell, G.: Core Java^{TM}: Volume I, Fundamentals. 9th ed., Prentice Hall, 2012. ISBN 978-0-13-708234-6.

[16] Lutz, M.: Python: Pocket Reference. 5th ed., O'Reilly, 2014. ISBN 978-1-449-35701-6.

[17] Pessemier, W.—Deconinck, G.—Raskin, G.—Saey, P.—Van Winckel, H.: UAF: A Generic OPC Unified Architecture Framework. Software and Cyber-infrastructure for Astronomy II., Proceedings of the SPIE, Vol. 8451, 2012, Art. No. 84510P, 10 pp.

Krzysztof Korcyl is Adjunct in Institute of Teleinformatics of Cracow University of Technology, Poland and in Institute of Nuclear Physics PAN, Cracow, Poland where he received his habilitation in physics. He worked for the third level trigger of Delphi experiment at LEP and subsequently for TDAQ system of ATLAS experiment at LHC at CERN, Geneva. His research interests include modeling of large scale real time systems and applicability of FPGA and GPGPU technologies for improving performance in data acquisition and filtering systems.

Piotr P. Nikiel is Software Engineer in the Detector Control System of the ATLAS experiment at LHC at CERN, Geneva. He received his M.Sc. in computing from Cracow University of Technology, Poland and his M.Sc. in electronic engineering from AGH University of Science and Technology, Cracow, Poland. He is software architect of the Quasar project and author of many OPC-UA based applications used at CERN. His research interests include model-based software engineering, embedded systems and programming languages.

Computing and Informatics, Vol. 37, 2018, 969–991, doi: 10.4149/cai_2018_4_969

ANNOTATING WEB TABLES WITH THE CROWD

Ning Wang, Huaxi Liu

School of Computer and Information Technology, Beijing Jiaotong University
No. 3 Shangyuancun, Haidian District, 100044 Beijing, China
e-mail: {nwang, 13120407}@bjtu.edu.cn

Abstract. The Web contains a large amount of structured tables, most of which lacks header rows. Algorithmic approaches have been proposed to recover semantics for web tables by annotating column labels and identifying subject columns. However, state-of-the-art technology is not yet able to provide satisfactory accuracy and recall. In this paper, we present a hybrid machine-crowdsourcing framework that leverages human intelligence to improve the performance of web table annotation. In this framework, machine-based algorithms are used to prompt human workers with candidate lists of concepts, while an improved K-means algorithm based on novel integrative distance is proposed to minimize the number of tuples posed to the crowd. In order to recommend the most related tasks for human workers and determine the final answers more accurately, an evaluation mechanism is also implemented based on Answer Credibility which measures the probability of a worker's intuitive answer being the final answer for a task. The results of extensive experiments conducted on real-world datasets show that our framework can significantly improve annotation accuracy and time efficiency for web tables, and our task reduction and answer evaluation mechanism is effective and efficient for improving answer quality.

Keywords: Crowdsourcing, semantic recovery, web tables, information integration

1 INTRODUCTION

Structured information of tables has a great value. In Google's recent research [1], the table schema and subject column are used to find related tables and to integrate multiple tables. Column labels are also used to learn binary relationships between multiple columns and class labels on cells [2]. The World Wide Web consists of a huge number of structured data in the form of HTML tables, most of which lacks header rows [3]. Some researchers tried to recover semantics of web tables by using large knowledge bases [4, 5], but the results are far from being perfect. According to the experimental results of [4], the precision of finding top-k concepts for each column using existing work is often low.

Recent studies have shown that crowdsourcing could be used effectively to solve problems that are difficult for computers, such as travel planning [6], open querying [7] and schema matching [8]. This can be applied for semantic recovery of web tables as well. For example, let us look at a simple table from a web page presented in Table 1 a). Human workers can easily decide that the concept for the third column is *capital* while the algorithm based on Probase [9] will return a candidate concept list *(city, large city, big city, capital, ...)*. For the second column of the table in Table 1 b), since the values are all numeric, it is really hard for the system to decide the concept for this column. But a human worker often can accurately determine that the concept is *price* according to the other columns' values. In addition to column label annotation, a worker is also able to play an important role in subject column identification.

1	2	3
America	English	Washington
China	Chinese	Beijing
Japan	Japanese	Tokyo

a) Example I

1	2	3
iPhone 4S 8 GB	2 217	white
iPhone 5C 8GB	3 159	black
Samsung S7572	818	white

b) Example II

Table 1. Web table

As we can see from the above analysis, it is difficult to provide satisfactory accuracy and recall when annotating web tables only by computer algorithms. To tackle this problem, we propose a hybrid machine-crowdsouring framework, which combines the strength from computer algorithms with human intelligence to find the best answers for web table annotation.

It is a non-trivial task to develop a framework to annotate web tables with crowdsourcing. First, if a table contains too many tuples, it is a really boring task for a worker to browse the whole table and decide its headers and subject column. Second, a worker may wonder how to start if they could not get any prompt of candidate information from the system. Third, when a worker lacks the required knowledge for handling a complex job, the human contributed results can be arbitrarily bad.

In this paper, we present a hybrid machine-crowdsourcing framework that leverages human intelligence to improve the performance of web table annotation. To the best of our knowledge, we are the first to leverage crowdsourcing for recovering semantics of web tables. To achieve that, our framework starts with prompting human workers with candidate lists of concepts provided by machine-based algorithms to help the workers to annotate column labels and identify subject keys. Then the workers are presented with a small number of representative tuples in the table by clustering. If necessary, a worker could see more tuples which are similar with the representative one. Finally, an evaluation mechanism is implemented based on Answer Credibility according to the setting of expertise and practical performance in order to recommend the most related tasks for workers and decide the final answers.

We summarize our contributions below:

1. We propose a hybrid machine-crowdsourcing framework for web table semantic recovery problem.
2. We present crowd with a small number of representative tuples in the table for task reduction by clustering similar tuples based on integrative distance biased towards significant attributes.
3. We propose Answer Credibility to evaluate the probability of a human worker's intuitive answer being the final answer for a task.
4. We establish an evaluation mechanism based on Answer Credibility, which is used to recommend related tasks and determine the final answers for each task.
5. We have conducted extensive experiments based on real web tables and hundreds of crowdsourcing tasks, which demonstrate that our framework can significantly improve annotation accuracy and time efficiency for web tables, and our task reduction and answer evaluation mechanism is effective and efficient for improving answer quality.

2 ANNOTATING WEB TABLES BASED ON PROBASE

Probase uses the world as its model which has an extremely large concept/category space (2.7 million categories) harnessed from billions of web tables and many years worth of searching logs [9]. As those concepts are automatically acquired from web pages authored by millions of users, it is probably true that they cover most concepts in our mental world (about worldly facts). In addition, it has a large data space, a large attribute space and a large relationship space. These characteristics make it perfect for getting candidate labels and entity columns of web tables.

To help human workers to annotate column labels and identify subject keys, it is necessary to prompt them with candidate lists of concepts by machine-based algorithms.

We leverage Probase to find concepts by column in Probase which share at least one cell value with that in the column of the web table, and estimate the probability of a concept c_k given a set of instances $E = \{e_1, \dots, e_n\}$ using a naive Bayes model as follows.

$$P(c_k|E) = \frac{P(E|c_k)P(c_k)}{P(E)} \propto P(c_k) \prod_{j=1}^{N} P(e_j|c_k), \tag{1}$$

$$P(e_i|c_k) = \frac{n(e_i, c_k)}{n(c_k)} \tag{2}$$

where $n(e_i, c_k)$ is the occurrence frequency of pair (e_i, c_k), and $n(c_k)$ is the occurrence frequency of c_k. Then the concepts are sorted by the probability and we choose the top-k concepts as candidate headers of the column.

Finally, we take the evidence-based method introduced in [5] to detect the candidate subject column in a web table.

3 REDUCING UNCERTAINTY OF ANNOTATION BY CROWDSOURCING

3.1 Task Reduction by Clustering Similar Tuples

When human workers are asked to complete tasks by browsing the whole tables, they will not only spend much time on tasks but also generate poor quality answers. Therefore,we propose a novel solution for task reduction, that is to cluster similar tuples and present workers with representative ones.

3.1.1 Integrative Distance

A web table is composed of columns with various data types. In fact, some of columns are more significant for clustering. In rough set, the core attribute is thought to have a greater contribution to classification and decision. To make the calculation of distance between two tuples effective, more weight should be put on significant attributes during clustering which are either core attributes [10] or representative attributes.

For the i^{th} column in a web table T, if we get top-k candidate concept set $CH = \{ch_1, \ldots, ch_k\}$ with the probability set $P = \{p_1, \ldots, p_k\}$ for this column, and top-k candidate concept set $RC = \{rc_1, \ldots, rc_k\}$ for whole table T based on Probase, the representative possibility of the i^{th} column, which describes the relevance degree between this column and table T, is computed as $rp_i = \frac{\sum_{c_j \in CH \cap RC} p_j}{|CH \cap RC|}$, where $p_j \in P$ is the probability of candidate concept c_j for this column. Representative attributes of T are those attributes with representative possibility which are beyond a threshold t_r.

Definition 1 (Significant Attribute)**.** Assume $CA = \{CA_1, \ldots, CA_m\}$ denotes the core attribute set of a web table and $RA = \{RA_1, \ldots, RA_n\}$ denotes its representative attribute set, then each one in their union set $SA = CA \cup RA$ is a significant attribute.

We use integrative distance biased towards significant attributes to evaluate the similarity between tuples in a web table.

Definition 2 (Integrative Distance)**.** For a web table T with attribute set $A = \{a_1, \ldots, a_n\}$, given the significant attribute set $SA = \{sa_1, \ldots, sa_l\}$, the distance set $D_{ij} = \{d_{ij}^1, \ldots, d_{ij}^n\}$ between two tuples t_i and t_j in table T, where d_{ij}^k denotes distance between corresponding $a_k (k = 1, \ldots, n)$ in t_i and t_j, and the weight set

$$w_{ij} = \begin{cases} \left\{ w_{ij}^1, \ldots, w_{ij}^n \middle| w_{ij}^p{}_{1\le p\le n, a_p \in SA} > w_{ij}^q{}_{1\le q\le n, a_q \notin SA} \right\}, & SA \neq \phi, \\ \left\{ w_{ij}^1, \ldots, w_{ij}^n \middle| w_{ij}^p{}_{1\le p\le n} = w_{ij}^q{}_{1\le q\le n} \right\}, & SA = \phi. \end{cases} \quad (3)$$

assigned on D, the integrative distance between t_i and t_j is calculated as follows.

$$D_{ij} = \sum_{k=1}^{n} w_{ij}^k d_{ij}^k. \quad (4)$$

Integrative distance is proposed to combine Euclidean distance with Jaccard similarity on different attributes in a web table. The distance function is biased towards the existing significant attributes.

3.1.2 Clustering Similar Tuples Based on Integrative Distance

In order to reduce the number of tuples posed to the crowd, we design a clustering algorithm named Clustering Algorithm based on Integrative Distance (CAID), which is an improved K-means algorithm, to cluster similar tuples in a web table and prompt workers with representative ones that are nearest to each of cluster centers.

We select K-means instead of K-medoids for computational constraints in crowdsourcing environment. Traditional K-means is improved in CrowdSR to adapt to web tables by using the integrative distance biased towards significant attributes.

As shown in Algorithm 1, CAID is different from naive K-means in step 3 and step 6, which are also our improvements. In step 3, we firstly get the signficant attributes of a table based on Probase. From step 5 to step 8, we get the initial clustering centers by finding the tuples which are farthest to each other. In step 6, we find the next tuple for initial clustering centers by evaluating the integrative distance biased towards significant attributes between tuples. Integrative distance is also used to reassign tuples and update clustering centers in step 10 and step 12.

3.2 Improving Answer Quality Based on Answer Credibility

Quality control is very important for a crowdsourcing platform. Based on the fact that a worker can give high quality answers when he does tasks with expertise, we propose Answer Credibility to evaluate his acquaintance with fields assigned by a task. In order to improve answer quality, we establish an evaluation mechanism based on Answer Credibility to recommend related tasks and decide the final answers.

Algorithm 1 CAID Clustering Algorithm

```
1: procedure GETCLUSERRESULT($T, K$)
2:     $F \leftarrow True$
3:     $SA \leftarrow getSignificantAttributes(T)$
4:     $addOneRandTuple(T, R)$
5:     while $R.size() <= K$ do
6:         $NT \leftarrow getNextFarthestTuple(T)$
7:         $addTuple(R, NT)$
8:     end while
9:     while $F$ do
10:        $reassignTuple(T, R)$
11:        for all $S \in R$ do
12:            $replaceCentroidWithMean(S, R)$
13:        end for
14:        $F \leftarrow centroidChange(R)$
15:    end while
16:    return $R$
17: end procedure
```

3.2.1 Answer Credibility

In our framework, users are divided into requesters who would like to publish tasks, and workers who are willing to accept and complete tasks. We use the set $F = \{f_1, \dots, f_m\}$ to describe the whole set of fields and each task is assigned with several fields in F by the requester when published.

For each worker, his answer credibility for a task is based on the degree of his acquaintance with each related field. Field credibility is therefore proposed to evaluate a worker's knowledge about some field in F, which is evaluated by following aspects.

1. Settings of expertise: At the beginning, each worker is required to choose several fields from F as his expertise. The initial score of each field is set by our system and the score of expertise (HS) is higher than the score of other fields (LS). Let $E = \{e_1, \dots, e_v | e_i \in F, 1 \leq i \leq v\}$ denote the set of expertise, the elementary score set for a worker is

$$ES = \left\{ es_1, \dots, es_m \,\middle|\, \sum_{1 \leq i \leq m} es_i = 1 \right\} \quad (5)$$

where

$$es_i = \begin{cases} HS, & f_i \in E, \\ LS, & f_i \notin E, \end{cases} \quad (0 < LS < HS < 1). \quad (6)$$

2. Brilliance test: Furthermore, a new user is required to join our brilliance test to check his actual field credibility when he registers as a worker. Assume a worker completes M test tasks. Let $F_M = \{tf_1, \ldots, tf_M\}$ denote the field set selected by the worker and corresponding score set of test tasks be $S_M = \{s_1, \ldots, s_M\}$. Brilliance test score set is

$$BS = \{bs_1, \ldots, bs_m\} \tag{7}$$

where

$$bs_i = \begin{cases} \frac{s_j}{\sum_{1\le k\le M} s_k}, & (f_i \in F_M) \wedge (f_i = tf_j), \\ 0, & (f_i \notin F_M) \vee ((f_i \in F_M) \wedge (f_i \neq tf_j)). \end{cases} \tag{8}$$

3. Actual performance of doing tasks: For a task T, let $F_T = \{f_1, \ldots, f_t\}$ denote the field set assigned on T by requester, $IAns = \{Ians_1, \ldots, Ians_n, Ians_{n+1}\}$ denote a worker's intuitive answers for n column labels and one subject column, $FAns = \{Fans_1, \ldots, Fans_n, Fans_{n+1}\}$ denote the final answer summarized from several workers' intuitive answers for T, the incremental performance score set is

$$\Delta PS = \{\Delta ps_1, \ldots, \Delta ps_m\} \tag{9}$$

where

$$\Delta ps_i = \begin{cases} \frac{\sum_{1\le k\le n+1} |IAns(k)=FAns(k)|}{|IAns|}, & f_i \in F_T, \\ 0, & f_i \notin F_T. \end{cases} \tag{10}$$

Then the actual performance score set for this worker is

$$PS = \{ps_1^k, \ldots, ps_m^k\} \tag{11}$$

where

$$ps_i^k = \begin{cases} 0, & k = 0, \\ \frac{ps_i^{k-1} + \Delta ps_i}{\sum_{1\le j\le m} (ps_j^{k-1} + \Delta ps_j)}, & k \ge 1. \end{cases} \tag{12}$$

For a worker U and field $f_i \in F$, if his elementary score, brilliance test score and performance score for field f_i are es_i, bs_i and ps_i, the field credibility of U for f_i could be computed as

$$fc_i = \begin{cases} \lambda es_i + (1-\lambda) bs_i, & ps_i = 0, \\ \lambda_1 ps_i + \lambda_2 bs_i + \lambda_3 es_i, & ps_i > 0 \end{cases} \tag{13}$$

where $(0 < \lambda < 1) \wedge \left(\sum_{1\le j\le 3} \lambda_j = 1\right)$ and they are weights for corresponding possibility. For a new worker, we only take the settings of expertise and brilliance test into consideration, and he could improve his credibility only by doing actual tasks.

Definition 3 (Answer Credibility)**.** For a worker U working on task T, given the field set $F_T = \{f_1, \ldots, f_t\}$ assigned by requester on T, field credibility set $FC = \{fc_1, \ldots, fc_t\}$ and intuitive answers $IAns = \{Ians_1, \ldots, Ians_n, Ians_{n+1}\}$ of U on task T. The answer credibility of U for T is computed as

$$AC = \sum_{i=1}^{t} fc_i. \tag{14}$$

Answer credibility is modeled to evaluate the probability of which a worker's intuitive answer comes to be the final answer for a task, which could be used to recommend tasks for workers who are more competent for and to decide the final answers.

3.2.2 Evaluation Mechanism Based on Answer Credibility

Task Recommendation: Let $T = \{T_1, \ldots, T_n\}$ denote the task list. For a worker U, we get his answer credibility set $AC = \{AC_1, \ldots, AC_n\}$ as described before. Then we recommend $T_{rec} = \{T_1, \ldots, T_k\}$ for him with top-k answer credibility values.

Answer Decision: In order to get final answer from multiple workers, a voting mechanism is built on Answer Credibility. For instance, when u workers working on task T, $AC_T^1, \ldots, AC_T^u$ are their answer credibility values for T, $IAns^i (i = 1, \ldots, n+1)$ is the set of the i^{th} answers for i^{th} column label $(i = 1, \ldots, n)$ or subject column $(i = n + 1)$ gathered from u workers. If we use U^i to denote the set of workers whose candidate answers in $IAns^i$ are the same, the score of this candidate answer in $IAns^i$ is evaluated as $\sum_{u \in U^i} AC_T^u$, and the final answer for $IAns^i$ is the candidate answer with the maximum score.

4 SYSTEM OVERVIEW

We have implemented CrowdSR, a crowd enabled system for semantic recovering of web tables based on a hybrid machine-crowdsourcing framework [11]. CrowdSR is implemented in JSP with SQL Server database as back-end. Figure 1 depicts the system architecture.

User Interface is used to interact with users (both requesting a task and contributing to it). Basically, *DB* stores answers collected from the crowd, targeting information and details about each user and task. *Task builder* receives requests from requesters and builds tasks. *Crowd Manager* constantly receives an updated list of online workers from targeting information in *DB*.

Semantic Recovery Component is responsible for finding candidate column labels and subject columns for each task based on the *Semantic Library* (we now use Probase, which could be replaced by other third-party libraries easily). When a task is published, it is enriched with candidate answers by the *Semantic Recovery Component*.

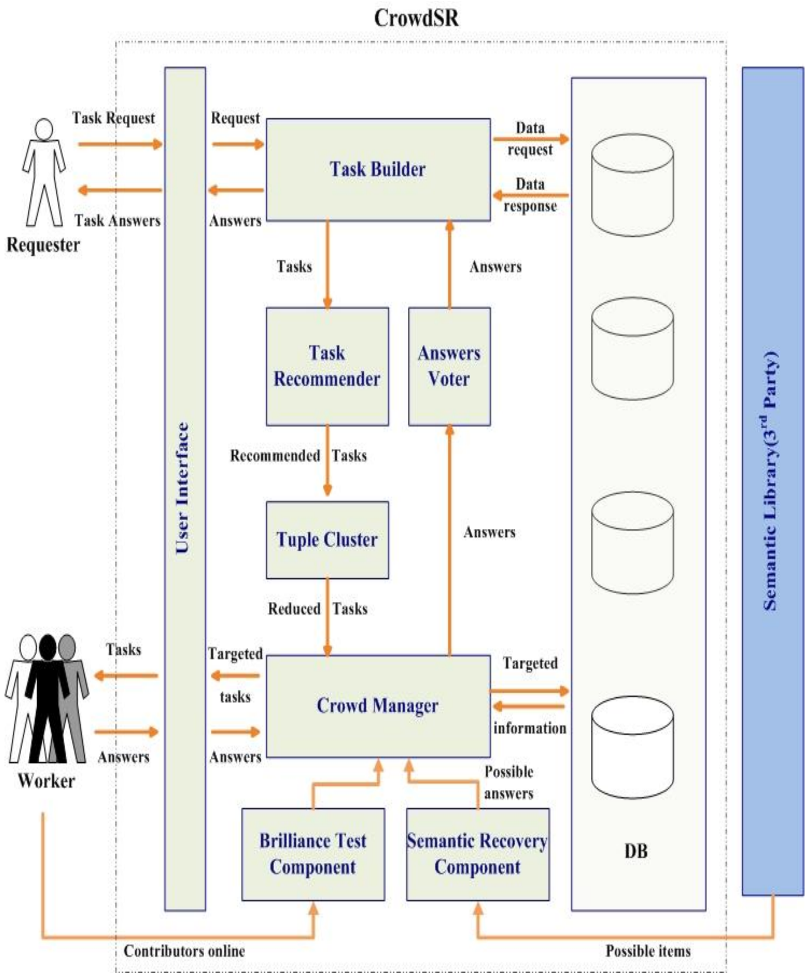


Figure 1. CrowdSR architecture

At the beginning, a worker is required to join our brilliance test to evaluate his understanding of each field through the *Brilliance Test Component*. *Task Recommender* and *Answers Voter* implement our evaluation mechanism based on Answer Credibility. Once a worker logs in, the *Task Recommender* recommends task list for him based on his answer credibility. While the deadline of a task arrives, received answers are passed to *Answers Voter* to decide the final answer.

Tuple Cluster executes our CAID algorithm to cluster similar tuples for each task. Workers complete tasks via a question-choice game where they are shown representative values of each column, along with a set of candidate table headers and subject column, and are asked to select ones most matched. Furthermore, a worker could check the alteration of his answer credibility based on his performance.

5 EXPERIMENTS AND RESULTS

The goals of our experiments are:

1. to evaluate the effectiveness of our hybrid machine-crowdsourcing framework for web table annotation based on Probase knowledge base;
2. to validate the performance of our CAID clustering methods based on integrative distance;
3. to evaluate the effectiveness of our evaluation mechanism based on Answer Credibility for task recommendation and answer decision.

5.1 Experiment Setup

Google Fusion Table (GFT) is a popular web application provided by Google to allow people, including those with no database expertise, to manage their data [13]. We choose GFT as one of our source of data for the reason that GFT tables have a neat, regular structure and the Australian government has already published its database on GFT. Besides, we extract large-scale web tables named WT which covers kinds of fields for crowdsourcing tasks to evaluate our mechanism based on Answer Credibility. Headers of tables in our dataset are known, but we remove them during experiment. Table 2 gives statistics of our data sets.

Dataset	Size (MB)	# Columns	# Rows	# Cells
GFT	4.5	261	15 132	3 949 452
WT	5.6	479	9 290	4 449 910

Table 2. Experimental data set

To conduct the experiments, we have implemented CrowdSR in JSP while using SQL Server as the database engine. The architecture of CrowdSR is presented in Figure 1. Compared with some existing platforms, such as Amazon Mechanical Turk (AMT) [22] and CrowdFlower [23], CrowdSR implemented task reduction and answer quality control. All our experiments were conducted on an Intel® Core™ 2.13 GHz computer with 2 GB of RAM running Windows 7.

5.2 Evaluation on Performance of CAID Clustering Algorithm

We select 10 representative web tables from different fields in GFT, which consist of labels used to get the clustering precision, to evaluate the clustering performance of our CAID clustering algorithm. Table 3 gives statistics of those tables. We compare CAID with naive K-means from two aspects, which are clustering precision and

Table Name	Size (KB)	# Columns	# Rows	# Cells
Agriculture	384	3	2 697	8 091
Architecture	71.5	6	347	2 081
Privacy Survey	1 028	45	1 000	45 000
Disaster	324	9	373	3 357
School	64.5	13	132	1 716
Crime	229	8	565	4 520
Books	303	9	786	7 074
Fish	154	7	452	3 164
Social Statistics	57	5	35	175
Wetland	584	7	1 000	7 000

Table 3. Statistics of 10 representative tables in GFT

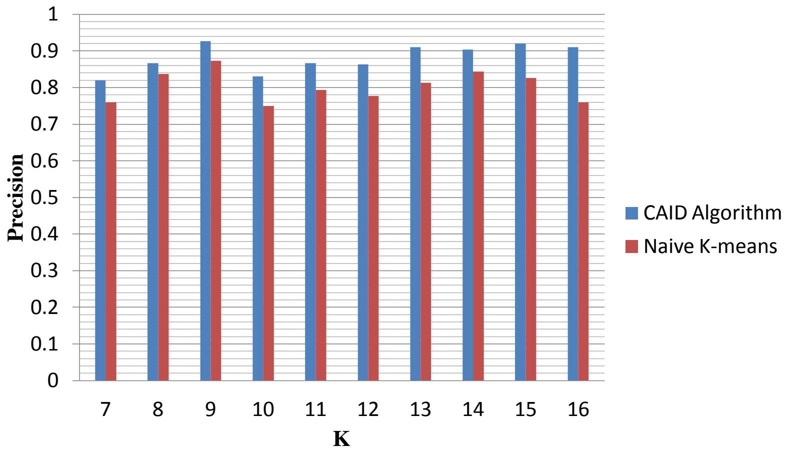
time efficiency. As the precision of naive K-means depends greatly on the selection of initial clustering centers, we ran it for 500 times to get its average precision just for fairness.

Before the experiment, we also took two factors below into consideration.

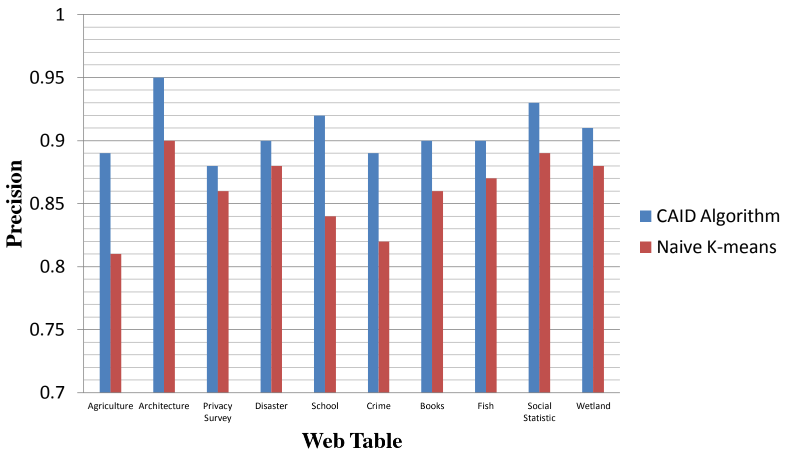
1. The number of clustering centers K: The size of K has influence on clustering algorithm. In order to compare the performance, we apply naive K-means and CAID with different size of K between $[7, 16]$. We select the region for experiments because it is easy for workers to finish tasks by browsing a small number of tuples.
2. Distance between tuples: As described before, CAID uses integrative distance to calculate similarity between tuples, which combines Euclidean distance with Jaccard similarity and gives high weight on significant attributes. Since a web table is usually composed of columns with various data types, we also combine Euclidean distance with Jaccard similarity for naive K-means just for fairness, but the weight of each column is equal.

Figure 2 a) shows the average precision of two algorithms with different k value in $[7, 16]$. The precision and time cost of two algorithms on 10 representative tables with $K = 9$ are displayed in Figures 2 b) and 2 c), respectively. We have following observations:

1. The precision of CAID is obviously higher than that of naive K-means, either average precision on different tables or precision on single table, because of the improvement of initial clustering centers and use of integrative distance. The precision of two methods varies with the change of k value, but both of them achieve the best with $K = 9$.
2. The time cost of naive K-means is much lower than that of CAID. Compared with naive K-means, CAID spends much time getting the initial clustering centers and significant attributes for each web table as described in Algorithm 1. The time for table of privacy survey and wetland are obviously higher as they



a) Average clustering precision with different K

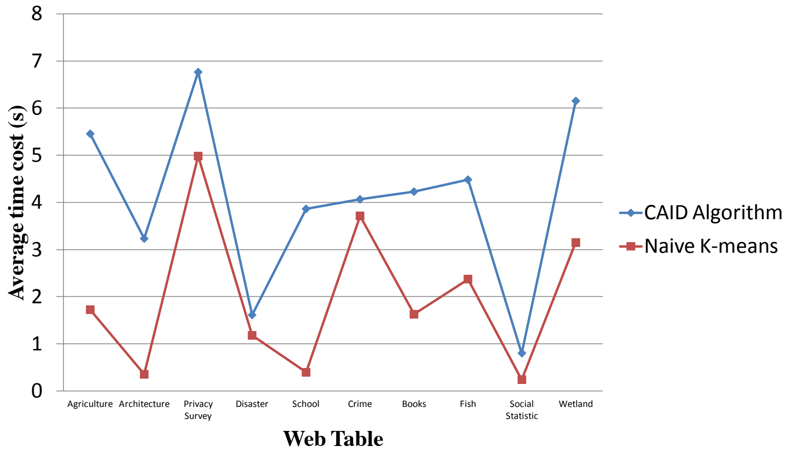


b) Clustering precision with $K = 9$

contain much more tuples. We will build index and improve time efficiency of CAID in future work.

5.3 Effectiveness of Hybrid Framework

We select web tables from WT data set which covers 12 different fields to publish crowdsourcing tasks. Tables from WT have original headers, and we labeled entity columns for each table as the golden standard. We enlisted hundreds of students from our university to join our experiments as workers. Each task is assigned with three fields when published and one task could be completed by several workers.



c) Clustering time cost with $K = 9$

Figure 2. Performance of CAID algorithm

Before doing the tasks, each worker was required to set his expertise and join our brilliance test. When he signed in, he was required to complete four kinds of tasks as following:

1. Recommended task with Representative tuples (RR): The task was recommended by the evaluation mechanism in CrowdSR and the workers were shown representative tuples in this task at first. They could also see more similar tuples by clicking corresponding buttons.
2. Recommended task with Whole tuples (RW): The task was recommended but the workers were shown all tuples in a table even if it contains hundreds of tuples.
3. General task with Representative tuples (GR): The general task that may not be recommended by our system and the workers were shown representative tuples in this task.
4. General task with Whole tuples (GW): The general task that may not be recommended by our system and the workers were shown all tuples in this task.

We use abbreviations RR, RW, GR and GW to refer the tasks in the following section. Finally, we collected the candidate answers and the time cost of each task to evaluate the effectiveness of our hybrid framework.

5.3.1 Evaluation on Hybrid Machine-Crowdsourcing Method

To compare the effectiveness of machine-based method with our hybrid machine-crowdsourcing method, we use precision, recall and F-measure, defined in Equations (15), (16) and (17):

$$Precision = \frac{|CAL|}{|AL|}, \quad (15)$$

$$Recall = \frac{|AL|}{|WL|}, \quad (16)$$

$$F = \frac{(1+\alpha) \times Precision \times Recall}{\alpha \times Precision + Recall}. \quad (17)$$

Precision measures the percentage of annotated headers or entity columns which are correct, and recall measures the percentage of headers or entity columns which could be annotated. F-measure is the weighted harmonic mean of precision and recall in which precision and recall are evenly weighted if $\alpha = 1$, and we set $\alpha = 1$ in this paper.

In Equations (15) and (16), WL is the set of whole headers or entity columns, AL is the set of annotated headers or entity columns and CAL is the set of headers or entity columns which are correctly annotated.

We took following two issues into consideration when collecting experimental data.

1. Answer Decision: Our hybrid machine-crowdsourcing framework can decide the final answers by the voting mechanism based on Answer Credibility. However, we did a favor for machine-based method. We chose the top three candidate concepts and the top one entity column returned by algorithm as the final answer set, and it is thought to be correctly annotated if one of concepts in answer set matched with golden standard.
2. Synonyms Principle: WT data set has original headers and labeled entity columns as the golden standard to judge the accuracy of results. But it is inevitable to face the synonyms problem for table headers. For example, we could get *author*, *writer* as the candidate headers for a column. So we made a principle that the word which is synonymous with the golden standard turns to be right. We use WordNet [24] as the synonymous words library to solve the problem.

According to the original experimental data in Table 4, the annotation precision, recall and F-measure on two approaches for headers and entity columns are calculated and shown in Figure 3. We could have the following observations:

1. For machine-based method, the precision of headers and entity columns are only 41.1 % and 41.38 %. But the average precision of our hybrid method are 57.29 % and 81.23 %. Our hybrid method could obviously improve the precision of web table annotation, especially for entity columns. The growth rate of the average precision on our hybrid approach for column headers (16.19 %) is much lower than that for entity columns (39.83 %) because the machine-based method uses the top three candidates rather than just the single top one candidate as final answers for headers.

Method	Machine-based	Hybrid Machine-crowdsourcing			
		RR	RW	GR	GW
# Total tables	130	41	51	66	55
# Total columns	479	249	246	275	301
# Total rows	9 290	1 746	2 048	3 997	4 810
# Annotated columns	163	155	179	186	182
# Correctly annotated columns	67	108	118	79	93
# Annotated entity columns	29	36	48	49	50
# Correctly annotated entity columns	12	31	40	37	40

Table 4. Original experimental data

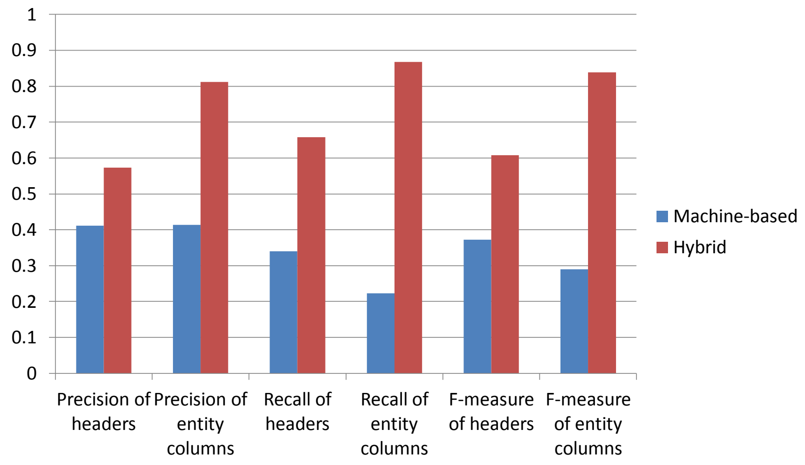


Figure 3. Comparison of annotation performance on two approaches

2. The machine-based algorithm totally processed 130 web tables with 479 columns and returned 29 candidate entity columns and 163 table headers. The recall of table headers and entity columns are only 163/479 = 34.0 % and 29/130 = 22.3 %. We gathered results from over five hundred crowdsourcing tasks. As one task could be completed by several workers, our hybrid method processed more than 1 000 columns. The average percentage of annotated columns and entity column are 74.88 % and 85.9 %, which are significantly higher than a machine-based method.
3. As the precision and recall are higher, the F-measure of our hybrid method is also better than that of a machine-based method.

It is difficult for computer algorithm to annotate web tables mainly due to two reasons. At first, the data format of web tables is irregular and there is some noisy data. Secondly, the web tables usually consist of long text and it is really hard for computer algorithm to analyze their semantics. Our hybrid method could significantly improve the performance of web table annotation by using crowd wisdom.

5.3.2 Benefits from Task Reduction

As described before, human workers are required to complete four kinds of tasks, which are Recommended task with Representative tuples (RR), Recommended task with Whole tuples (RW), General task with Representative tuples (GR) and General task with Whole tuples (GW). Task reduction means providing reduced tasks with representative tuples, such as RR or GR tasks, in which workers were firstly shown representative tuples and they could also see more similar data with current tuples when necessary. On the contrary, a table is completely shown in a page when workers are doing RW and GW tasks which are named full tasks. Although most of web tables consists of hundred of tuples and even some of tables contain thousands of tuples, the workers doing full tasks need to browse the whole table to decide their answers. We evaluate the benefits from task reduction by comparing the performance of reduced tasks with that of full tasks.

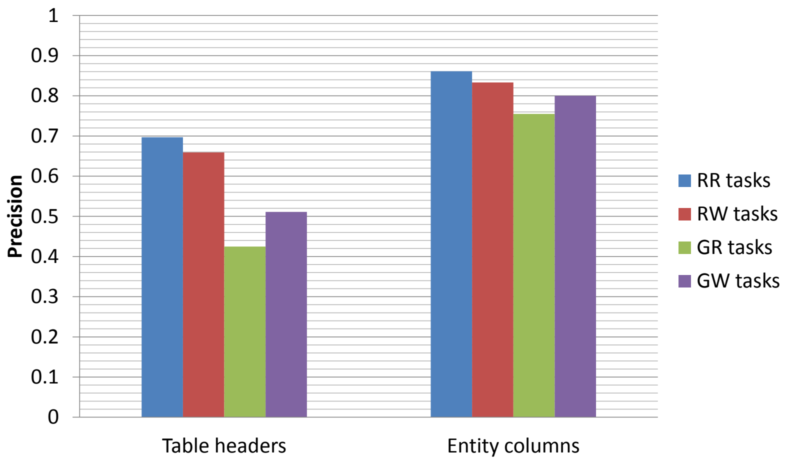
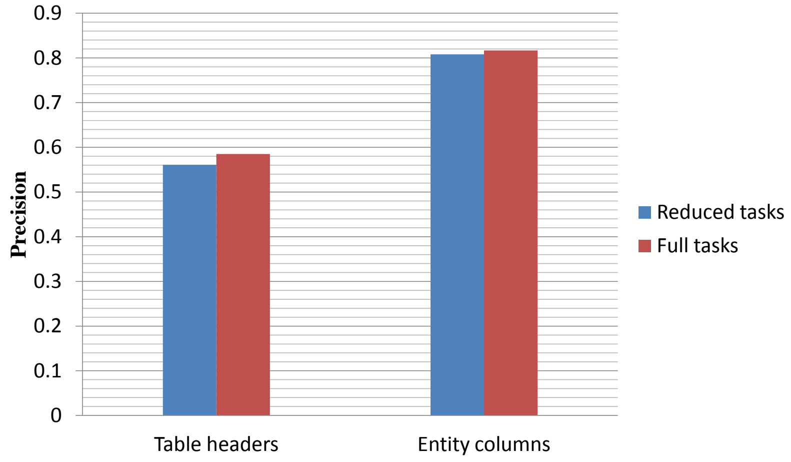


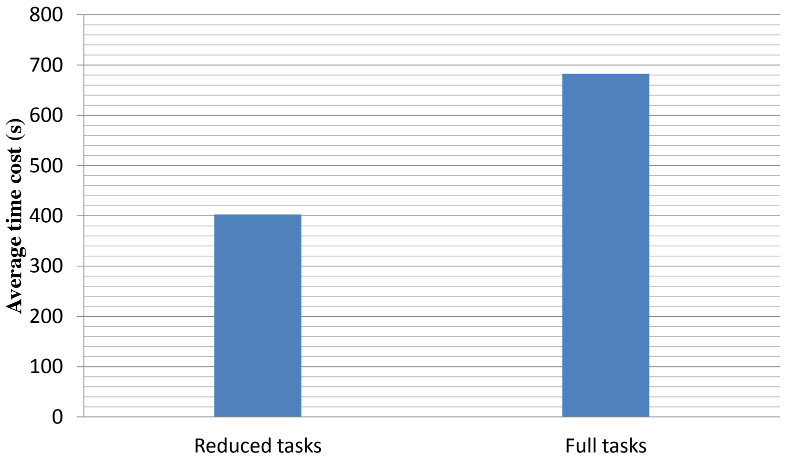
Figure 4. Annotation precision of RR, RW, GR and GW tasks

Figure 4 gives the annotation precision of RR, RW, GR and GW tasks. The average precision of reduced tasks and full tasks, and the corresponding time cost are shown in Figures 5 a) and 5 b), respectively. We have the observations as follows:

1. As shown in Figures 4 and 5 a), although the precision of RR tasks is better than that of RW tasks, the average precision of full tasks is yet a little higher than that of reduced tasks mainly because the precision of GR tasks is much lower than that of GW tasks. For general tasks with the fields workers may not be familiar with, it may be better to browse whole tuples to make decision. In



a) Comparison of annotation precision



b) Comparison of time cost

Figure 5. Comparison between reduced and full tasks

CrowdSR, we have an effective evaluation mechanism based on Answer Credibility to recommend tasks for workers who are more competent for, which is beneficial for improving annotation precision.

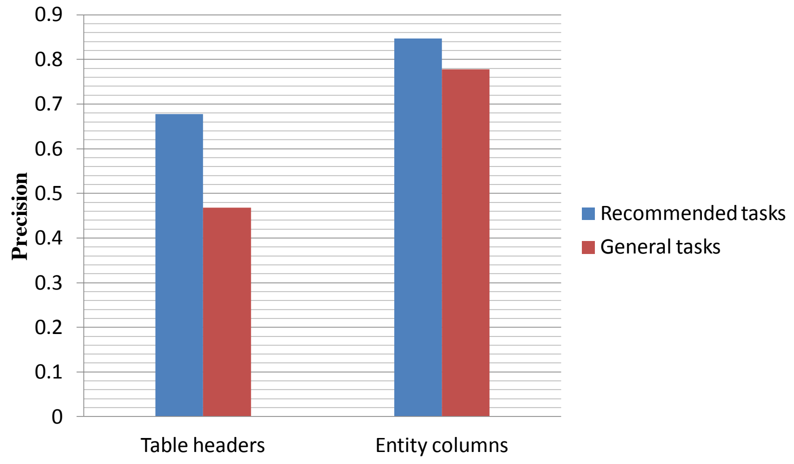
2. As in Figure 5 b), the average time cost of reduced tasks is only 402.62 seconds, which is nearly 70.0 % lower than that of full tasks. The result demonstrates that task reduction could obviously reduce the time cost.

Furthermore, in order to investigate the workers' attitude towards different kinds of tasks, we conducted a questionnaire survey after they complete the tasks. 83.3 %

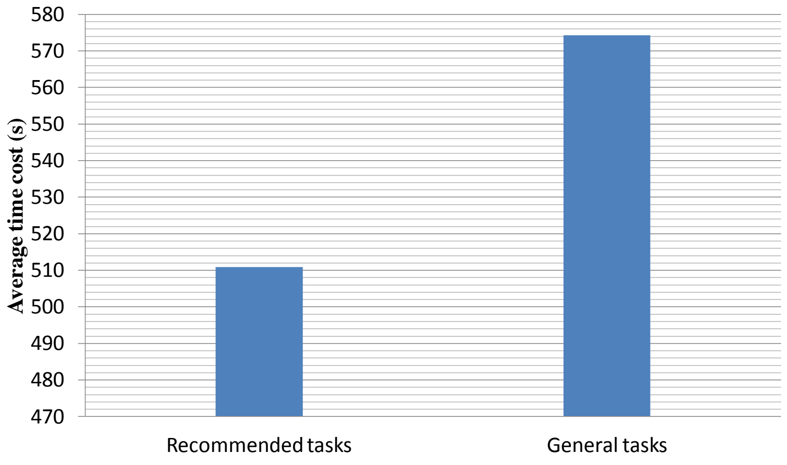
of them thought the reduced tasks are enjoyable, while 66.7 % of them considered the full tasks are really boring. The results prove that human workers prefer to do reduced tasks rather than full tasks.

5.3.3 Effectiveness of Answer Evaluation Mechanism

In this section, we will validate the effectiveness of answer evaluation mechanism by comparing the performance of recommended tasks and general tasks.



a) Comparison of Annotation Precision



b) Comparison of Time Cost

Figure 6. Comparison between Recommended and General Tasks

CrowdSR recommends tasks for each worker according to his answer credibility. At the beginning, we get the initial answer credibility of each worker according to his expertise settings and performance of brilliance test. Their answer credibility values are dynamically changed when they are doing actual tasks.

As shown in Figures 6 a) and 6 b), the performance of recommended tasks is obviously better than that of general tasks. The average annotation precision of recommended tasks increased about 14.0 % over that of general tasks, also with lower time cost. The result demonstrates that our answer evaluation mechanism works much effectively.

6 RELATED WORK

Semantic recovery for web tables is obviously important for web search to take advantage of those high quality sources of relational information. Deng et al. tried to determine the column concepts for web tables by using large knowledge bases [4] and Wang et al. used a method based on serveral kinds of evidence to find the entity column [5]. In addition, Venetis et al. tried to recover the semantics of web tables by enriching the table with additional annotations [12]. But according to their experimental results, the accuracy is far from perfect and the machine-based method is unstable [4, 5].

Using crowd wisdom for settling problems that computers fail to solve has attracted much research attention in recent years, especially in database and data mining communities. CrowdDB used human input via crowdsourcing to process queries that neither database systems nor search engines could adequately answer [7]. Amsterdamer et al. proposed the identification of frequent itemsets in human knowledge domains by posing questions to the crowd [13]. A novel algorithm is developed in CrowdPlanr for efficiently harnessing the crowd to assist in answering planning queries [6]. Besides, crowdsourcing-based solutions of many complex algorithms are also developed, such as crowd-assisted search problem in a graph [15], crowd-based entity resolution for data cleaning [16, 17, 18, 19] and schema matching via crowdsourcing for data integration [8]. In this paper, we present a hybrid machine-crowdsourcing framework that leverages human intelligence to improve the performance of web table annotation.

There also exist several crowdsourcing platforms, such as AMT [22] and CrowdFlower [23], but they are general platforms which are facing the problem that humans are prone to errors and an incorrect answer may be provided especially when a person is lacking the required knowledge for handling the task. In AMT, a job is split into many HITs (Human Intelligence Tasks) and each HIT is assigned to multiple workers so that replicated answers are obtained and the majority voting strategy is adopted for deciding the final answer. As human cost is expensive, we will have to pay a high cost if we assign each HIT to too many workers. In order to implement quality management on AMT, Ipeirotis et al. presented an algorithm that separated the unrecoverable error rate from bias by generating a scalar score for distinguishing workers who are careful but biased with those spammers [20]. Furthermore, Karger et al. introduced a model in which a requester has a set of homogeneous labeling tasks he must assign to workers who arrive online [21]. They proposed an assignment algorithm based on random graph generation and showed that their technique is order-optimal in terms of labeling budget. Although these are important aspects for a crowdsourcing platform, our focus here is different. We implement an evaluation mechanism based on Answer Credibility to improve answer quality for annotating web tables.

To the best of our knowledge, we are the first to leverage crowdsourcing for semantic recovering of web tables. Compared with all of them, we firstly propose to reduce tasks by using clustering algorithm, define answer credibility for answer quality control by task recommendation and answer decision.

7 CONCLUSION

In this paper, we present a hybrid machine-crowdsourcing framework that leverages human intelligence to improve the performance of web table annotation. We implement task reduction by minimizing the number of tuples posed to the crowd, task recommendation and answer decision by evaluating answer credibility of every worker for each task. Furthermore, we conduct extensive experiments based on real web tables and crowdsourcing tasks, which demonstrate that our framework can obviously improve annotation accuracy and time efficiency for web tables, and our task reduction and answer mechanism is effective and efficient for improving answer quality. In the future, we will try to improve the time efficiency of CAID algorithm and use crowdsourcing to solve table fusion problem.

Acknowledgement

This work is supported by the National Key R & D Program of China (2018YFC08-09800) and the National Natural Science Foundation of China (61370060, 61300071).

REFERENCES

[1] Das Sarma, A.—Fang, L.—Gupta, N. et al.: Finding Related Tables. Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, 2012, pp. 817–828, doi: 10.1145/2213836.2213962.

[2] Limaye, G.—Sarawagi, S.—Chakrabarti, S.: Annotating and Searching Web Tables Using Entities, Types and Relationships. Proceedings of the VLDB Endowment, Vol. 3, 2010, No. 1-2, pp. 1338–1347, doi: 10.14778/1920841.1921005.

[3] Cafarella, M. J.—Halevy, A.—Wang, D. Z.—Wu, E.—Zhang, Y.: WebTables: Exploring the Power of Tables on the Web. Proceedings of the VLDB Endowment, Vol. 1, 2008, No. 1, pp. 538–549, doi: 10.14778/1453856.1453916.

[4] Deng, D.—Jiang, Y.—Li, G.—Li, J.—Yu, C.: Scalable Column Concept Determination for Web Tables Using Large Konwledge Bases. Proceedings of the VLDB Endowment, Vol. 6, 2013, No. 13, pp. 1606–1617.

[5] Wang, J.—Wang, H.—Wang, Z.—Zhu, K. Q.: Understanding Tables on the Web. In: Atzeni, P., Cheung, D., Ram, S. (Eds.): Conceptual Modeling (ER 2012). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7532, 2012, pp. 141–155.

[6] Lotosh, I.—Milo, T.—Novgorodov, S.: CrowdPlanr: Planning Made Easy with Crowd. Proceedings of the 2013 IEEE 29th International Conference on Data Engineering (ICDE), 2013, pp. 1344–1347, doi: 10.1109/ICDE.2013.6544940.

[7] Franklin, M. J.—Kossmann, D.—Kraska, T.—Ramesh, S.—Xin, R.: CrowdDB: Answering Queries with Crowdsourcing. Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data (SIGMOD '11), 2011, pp. 61–72, doi: 10.1145/1989323.1989331.

[8] Zhang, C. J.—Chen, L.—Jagadish, H. V.—Cao, C. C.: Reducing Uncertainty of Schema Matching via Crowdsourcing. Proceedings of the VLDB Endowment, Vol. 6, 2013, No. 9, pp. 757–768, doi: 10.14778/2536360.2536374.

[9] Wu, W.—Li, H.—Wang, H.—Zhu, K. Q.: Probase: A Probabilistic Taxonomy for Text Understanding. Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data (SIGMOD '12), 2012, pp. 481–492, doi: 10.1145/2213836.2213891.

[10] Han, S.-W.—Kim, J.-Y.: Rough Set-Based Decision Tree Using a Core Attribute. International Journal of Information Technology and Decision Making, Vol. 7, 2008, No. 2, pp. 275–290.

[11] Liu, H.—Wang, N.—Ren, X.: CrowdSR: A Crowd Enabled System for Semantic Recovering of Web Tables. In: Dong, X., Yu, X., Li, J., Sun, Y. (Eds.): Web-Age Information Management (WAIM 2015). Springer, Cham, Lecture Notes in Computer Science, Vol. 9098, 2015, pp. 581–583.

[12] Venetis, P.—Halevy, A.—Madhavan, J.—Paşca, M.—Shen, W.—Wu, F.—Miao, G.—Wu, C.: Recovering Semantics of Tables on the Web. Proceedings of the VLDB Endowment, Vol. 4, 2011, No. 9, pp. 528–538, doi: 10.14778/2002938.2002939.

[13] Amsterdamer, Y.—Grossman, Y.—Milo, T.—Senellart, P.: Crowd Mining. Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (SIGMOD '13), 2013, pp. 241–252, doi: 10.1145/2463676.2465318.

[14] Gonzales, H.—Halevy, A. Y.—Jensen, C. S. et al.: Google Fusion Tables: Web-Centered Data Management and Collaboration. Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data (SIGMOD '10), 2010, pp. 1061–1066.

[15] Parameswaran, A.—Das Sarma, A.—Garcia-Molina, A.—Polyzotis, N.—Widom, J.: Human-Assisted Graph Search: It's Okay to Ask Questions. Proceedings of the VLDB Endowment, Vol. 4, 2011, No. 5, pp. 267–278, doi: 10.14778/1952376.1952377.

[16] Vesdapunt, N.—Bellare, K.—Dalvi, N.: Crowdsourcing Algorithms for Entity Resolution. Proceedings of the VLDB Endowment, Vol. 7, 2014, No. 12, pp. 1071–1082, doi: 10.14778/2732977.2732982.

[17] Wang, J.—Kraska, T.—Franklin, M. J.—Feng, J.: CrowdER: Crowdsourcing Entity Resolution. Proceedings of the VLDB Endowment, Vol. 5, 2012, No. 11, pp. 1483–1494.

[18] Wang, J.—Li, G.—Kraska, T.—Franklin, M.—Feng, J.: Leveraging Transitive Relations for Crowdsourced Joins. Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (SIGMOD '13), 2013, pp. 229–240.

[19] Whang, S. E.—Lofgren, P.—Garcia-Molina, H.: Question Selection for Crowd Entity Resolution. Proceedings of the VLDB Endowment, Vol. 6, 2013, No. 6, pp. 349–360.

[20] Ipeirotis, P. G.—Provost, F.—Wang, J.: Quality Management on Amazon Mechanical Turk. Proceedings of the ACM SIGKDD Workshop on Human Computation (HCOMP '10), 2010, pp. 64–67, doi: 10.1145/1837885.1837906.

[21] Karger, D.—Oh, S.—Shah, D.: Budget-Optimal Task Allocation for Reliable Crowdsourcing Systems. Operations Research, Vol. 62, 2014, No. 1, pp. 1–24, doi: 10.1287/opre.2013.1235.

[22] AMT Web Site. Available at: `https://www.mturk.com/mturk/welcome`.

[23] CrowdFlower Web Site. Available at: `http://www.crowdflower.com/`.

[24] WordNet Web Site. Available at: `http://wordnet.princeton.edu/wordnet/`.

Ning Wang received her Ph. D. degree in computer science in 1998 from Southeast University in Nanjing, China. She is currently serving as Professor in School of Computer and Information Technology, Beijing Jiaotong University, China. Her research interests include web data integration, big data management, data quality and crowdsourcing.

Huaxi Liu received his Master degree in computer science from Beijing Jiaotong University in 2016 and he currently works in Huawei Technologies Co., Ltd. His research interests include web data integration, data mining and crowdsourcing.

Computing and Informatics, Vol. 37, 2018, 992–1010, doi: 10.4149/cai_2018_4_992

A NOVEL KERNEL FOR TEXT CLASSIFICATION BASED ON SEMANTIC AND STATISTICAL INFORMATION

Haipeng Yao, Bo Zhang, Peiying Zhang

School of Information and Communication Engineering
Beijing University of Posts and Telecommunications
No. 10 Xitucheng Road
Haidian District, Beijing, China
e-mail: yaohaipeng@bupt.edu.cn, zhangbobupt@qq.com, zhangpeiying@upc.edu.cn

Maozhen Li

Department of Electronic and Computer Engineering
Brunel University London
Uxbridge, UB8 3PH, UK
e-mail: Maozhen.Li@brunel.ac.uk

Abstract. In text categorization, a document is usually represented by a vector space model which can accomplish the classification task, but the model cannot deal with Chinese synonyms and polysemy phenomenon. This paper presents a novel approach which takes into account both the semantic and statistical information to improve the accuracy of text classification. The proposed approach computes semantic information based on HowNet and statistical information based on a kernel function with class-based weighting. According to our experimental results, the proposed approach could achieve state-of-the-art or competitive results as compared with traditional approaches such as the k-Nearest Neighbor (KNN), the Naive Bayes and deep learning models like convolutional networks.

Keywords: Text categorization, semantic information, statistical information, support vector machine

Mathematics Subject Classification 2010: 68T50

1 INTRODUCTION

In recent years, with the increasing volume of text information on the Internet and social media, text categorization has become a key technique to process these textual data. In text categorization, a Bag of Words (BOW) is usually used to represent a document. The weight in BOW is usually obtained by computing word frequency or the widely accepted TF-IDF formula. However, the BOW representation has several limitations:

1. The TF-IDF formula does not consider the calculation of class-based weights. As a result, the same word has the same weight in all categories.
2. It cannot deal with synonyms and polysemy.

In the absence of knowledge-based word similarity and statistic-based word similarity, automatic text categorization using BOW only as a document representation model [1] has not yet achieved the best performance and cannot meet the needs of all scenes in the real life. There are two ways to address this problem. First, we could use language model based on deep learning models such as word2vec [2] and Glove [3] to learn the vector representations of words. However, such new approaches do not have to be necessarily better when the corpora is not particularly large. And it takes considerable time and effort to train word vectors. The second method is to collect semantic and syntactic informations as much as possible. We mainly adopt the second method and develop a new semantic smoothing kernel function based on knowledge-based word similarity and statistic-based word similarity to increase the capability of feature vectors to represent a document.

This paper presents a novel approach for text classification. In this approach, word similarity based on HowNet is embedded into semantic information. This method promisingly improves the accuracy of text classification via using ontology knowledges. Moreover, the proposed approach takes advantage of the class-based term weighting by giving more weights on core words in each class during the transformation phase of SVM from the input space to the feature space. A term has a more discriminative power on a class if it has a higher weight for that class. The heuristic idea combining sematic and statistical information finally improves the classification accuracy.

The rest of the paper is organized as follows: In Section 2, we briefly introduce the Support Vector Machine (SVM) and discuss the related work in the field of semantic smoothing kernels, with an emphasis on the task of text classification. Section 3 describes and analyzes the proposed kernel for text classification. Experimental setup and corresponding experiment results are given in Section 4. Finally, we conclude this paper in Section 5 with a discussion on future work.

2 RELATED WORK

2.1 Support Vector Machines for Classification

Support vector machine (SVM) is a very effective machine learning algorithm developed from statistical learning theory. This algorithm was proposed by Vapnik, Guyon and Boser [4] and further analyzed in [5]. The core goal of SVM is to find the optimal segmentation hyperplane by the maximum spacing between classes. The author of [6] proposed that SVM has many advantages, such as finding the global optimal solution and having a good robustness.

SVM kernel function can be regarded as similarity function, because it calculates the similarity values of data sets. It is proposed to define a suitable kernel function [7], which has a direct influence on finding the optimal hyperplane. The commonly used kernel functions for the document vectors are given below:

$$\mathit{LinearKernel} : k(d_p, d_q) = d_p d_q, \tag{1}$$

$$\mathit{PolynomialKernel} : k(d_p, d_q) = (d_p d_q)^b, b = 1, 2, \dots, \tag{2}$$

$$\mathit{RBFKernel} : (d_p, d_q) = \exp(\gamma||d_p - d_q||)^2. \tag{3}$$

In current works, the authors of [8] proposed to develop a kernel function based on the similarity of the knowledge system, and used the Omiotis library function to measure the similarity of English words and improve the accuracy of the classification. The authors of [9] proposed to develop a kernel function based on the weights of the class which improved the accuracy of the classifier. Based on these research efforts, this paper optimizes the similarity of Chinese text words and combines the statistical methods with the knowledge-based methods to construct a kernel functions to improve the accuracy of text classification.

2.2 Knowledge-Based Word Similarity

Knowledge-based systems use ontology or thesaurus to capture the concepts in the documents and incorporate the domain knowledge into the words for the representation of textual data. These systems enhance the representation of terms by taking advantages of semantic relatedness among terms.

The similarity calculation of words is added to the text classification [10], which is used to modify the weights of the text feature vectors. Mavroeidis et al. [11] proposed a semantic kernel function based on WordNet [12] to improve the accuracy of English text classification. Based on the Chinese semantic knowledge system of HowNet [13], Zhang embeded the semantic similarity into the kernel function of Chinese text classification [14], and improved the performance of Chinese text classification.

In this paper, we use the Chinese dictionary HowNet to calculate the semantic similarity of words. HowNet is a very detailed dictionary of semantic knowledge.

Unlike CiLin [15] and WordNet, every word in HowNet has multidimensional knowledge representations. The structure of HowNet is described in detail below.

HowNet mainly includes concepts and primitives. Each term is described by a number of concepts, each of which is described by a sequence of primitives, so primitive is the smallest expression unit in HowNet. HowNet contains 1 500 primitives, which can be divided into three categories: basic semantics (describing the semantic features of concepts), grammatical semantics (describing the grammatical features of words) and relational semantics (describing the relationship between concepts). When we calculate the word similarity, we can define it in the following way. Word similarity calculation consists of four parts in Equation (4).

$$Sim(S_1, S_2) = \sum_{i=1}^{4} \beta_i \prod_{j=1}^{i} Sim_j(S_1, S_2). \tag{4}$$

$Sim_1(S_1, S_2)$ is the similarity of first basic primitives of these two words. The similarity $Sim_1(S_1, S_2)$ between S_1 and S_2 can be calculated using Equation (5). $Sim_2(S_1, S_2)$ is the similarity of the rest basic primitives, that is the arithmetic mean of the similarity of all pairs of elements. $Sim_3(S_1, S_2)$ is the similarity of two grammatical semantics, which can be transformed into the basic semantic meaning in the grammatical semantics. $Sim_4(S_1, S_2)$ is the similarity of two relational semantics, but the elements in the relational semantics are sets, which are basic primitives or concrete words.

There is a close relationship between word similarity and word distance. In fact, word similarity and word distance are different forms of the same feature of a pair of words. Word similarity is defined as a real number between 0 and 1.

$$Sim_1(S_1, S_2) = \frac{\alpha}{d + \alpha} \tag{5}$$

where S_1 and S_2 represent two of the words respectively, d is the distance between S_1 and S_2 in the original path hierarchy in HowNet; α is an adjustable parameter. When the distance in HowNet between words is particularly large, $Sim(S_1, S_2)$ approaches 0; when the distance in HowNet between words is particularly small, $Sim(S_1, S_2)$ approaches 1.

β_i in Equation (4) is an adjustable parameter and satisfies the Equation (6). The latter part of the Equation (6) represents the descending importance of $Sim_1(S_1, S_2)$ to $Sim_4(S_1, S_2)$.

$$\beta_1 + \beta_2 + \beta_3 + \beta_4 = 1, \quad \beta_1 \geq \beta_2 \geq \beta_3 \geq \beta_4. \tag{6}$$

2.3 Statistic-Based Word Similarity

In the absence of semantic knowledge, a statistical-based approach such as the one presented in [16] can be applied to the text categorization to solve synonymic problems. Statistical similarity calculation is based on the correlation of training words, so the statistical similarity calculation method is very sensitive to the training data sets.

The similarity calculation using statistics in text categorization contains calculation based on classes [9], high-order paths [17, 18, 19] and mean value calculations [20]. In this paper, we use the method proposed in [9] which considers the weight of a certain word in a training set depending on the relevance of terms and categories. After the text feature vector is smoothed through the semantic kernel, it can improve the weights of important words in the category and reduce the weight of common words in the category. By modifying weights of the text feature vectors, the representation capability of the feature vectors is increased.

2.4 Weight of Feature Words

In the classification system, the most commonly used method of calculating word weights is the TF-IDF formula mentioned in [21, 22] where TF denotes the term frequency and IDF denotes inverse document frequency. TF-IDF formula was first used in the field of information retrieval, because its calculation method is simple and practical. It is also widely used in the text automatic classification.

TF-IDF is a statistical method to evaluate the importance of a document in a corpus. In general, the importance of a word increases in proportion to its number of occurrences in the document and decreases inversely with its higher frequency of occurrences in the corpus.

IDF formula is given in Equation (7)

$$IDF(w) = \frac{|D|}{df_w} \tag{7}$$

where $|D|$ denotes the total number of documents in the corpus and df_w represents the number of documents which contains term w.

TF-IDF formula is given in Equation (8)

$$TFIDF(w, d_i) = tf_w * \log(IDF(w)) \tag{8}$$

where tf_w represents the term frequency which is the number of word w in document D.

TF-ICF is proposed as another method of calculating word weight in [23, 24], which is similar to TF-IDF. ICF denotes inverse class frequency. TF-ICF calculates the word weight in category level rather than in document level.

Equation (9) shows the ICF formula:

$$ICF(w) = \frac{|C|}{cf_w} \tag{9}$$

where $|C|$ denotes the total number of classes in the corpus and cf_w represents the number of classes which contain term w.

TF-ICF formula is shown in Equation (10),

$$TFICF(w, c_i) = \sum_{d \in c_j} tf_w * \log(ICF(w)). \tag{10}$$

Inspired by the IDF and ICF formulas, [25, 26] propose a new method for calculating weights:

$$W_{w,c} = \log(tfc_{w,c} + 1) * \frac{|D|}{df_w} \tag{11}$$

where $tfc_{w,c}$ represents the total number of feature term w of class c. $|D|$ denotes the total number of documents and df_w represents the number of documents which contain term w.

From the analysis above, we can see that W is a matrix which is determined by categories and feature terms. In fact, terms that are similar to the topic in the category are given a larger weight because of the W matrix. The authors of [25, 26] compare the weighting algorithm based on the category with other commonly used feature selection algorithms, and conclude that the former one can improve the classification performance significantly.

3 MERGED KERNEL FUNCTION

Both knowledge-based word similarity computation and statistical-based word similarity computation can improve the performance of text classifiers. A heuristic idea is to apply two computation methods to a kernel function which we call merged kernel function in this paper. The pseudocode for the merged kernel function is shown in Algorithm 1. This section will explain in detail how to combine the two kernel functions to improve the accuracy of classification.

3.1 Vector Space Model

Document representation is a basic problem in natural language processing. Computer cannot directly deal with document which is mainly consisted of unstructured data. The key challenge is how to map a document into a vector space model (VSM). First, a document d_i is represented as an n-dimensional vector composed of feature words as shown in Equation (12).

$$d_j = (w_{1j}, w_{2j}, \ldots, w_{nj}). \tag{12}$$

Then the weighting formula maps the document vector d_i to the word weight vector $\phi(d_j)$:

$$\phi(d_j) = [\mathit{tfidf}(t_1, d_j), \mathit{tfidf}(t_2, d_j), \ldots, \mathit{tfidf}(t_n, d_j)] \tag{13}$$

where $\mathit{tfidf}(t_i, d_j)$ denotes the TF-IDF value of the feature word t_i in the document d_j.

```
Algorithm 1 Calculation of the combined semantic smoothing matrix C
Require: Training set D
Ensure: Semantic Smoothing Matrix C
Local variables :
        $tfc_{w,k}$ : total term frequency of word $w$ in the documents of class k
        $tf_{w,d}$  : total term frequency of word $w$ in the document d
        $N$         : total number of documents in the training set
        $N_w$       : shows total number of documents in the training set those contain
    word w
        $Z$         : matrix is constructed according to the list of feature word by the
    introduction of HowNet
 1: for each word w do
 2:     for each document d contains word $w$ in the train set do
 3:         $N_w = N_w + 1$
 4:     end for
 5: end for
 6:
 7: for each word w do
 8:     for each document d contains word $w$ in class k do
 9:         $tfc_{w,k} = tfc_{w,k} + tf_{w,d}$
10:     end for
11:     $W_{w,k} = (log(tfc_{w,k}) + 1) * log(N/N_w)$
12: end for
13:
14: $S = W * W^T$
15:
16: $Z^2 = Z * Z^T$
17:
18: for  i in columns of S do
19:     for for j in rows of S do
20:         $C_{i,j} = \lambda_1 * S_{ij} + \lambda_2 * Z^2_{ij}$
21:     end for
22: end for
```

3.2 Statistic-Based Similarity Matrix

The training phase for statistic-based similarity matrix is shown in Algorithm 1 from the first line to the fourteenth line. In order to embed the statistical information into the space vector model and increase the capability of representing a document of feature vectors, we construct a matrix S based on the class-based weight, which is called the statistical similarity matrix. The formula has been described in detail in Section 2.4. To make use of this formula, we define the statistical similarity matrix S as:

$$S = WW^T \tag{14}$$

where W is the class-based weighting formula mentioned in Section 2.4. The statistical similarity matrix S is a symmetric matrix, and the element S_{ij} in matrix S is the class-based statistical similarity of the feature words w_i and w_j.

The S matrix represents the statistical similarity of words. For example, the words "patient" and "sufferer" have similar meaning. When the weight of the "patient" is higher and the weight of the "sufferer" is lower in vector $\varphi(d_j)$, the weight of the word sufferer will be increased after multiplication with the S matrix.

3.3 Combined Semantic Smoothing Matrices

The knowledge-based similarity matrix Z can be constructed according to the list of feature words by the introduction of HowNet in Section 2.2. Z_{ij} denotes the knowledge-based similarity between the feature words w_i and w_j. The knowledge-based word similarity can be embedded into the space vector model by matrix Z. In order to ensure the validity of final merged kernel, a second-order rule similarity matrix is used here, that is $Z^2 = ZZ^T$.

After computing the statistic-based similarity matrix S and the second-order knowledge-based similarity matrix Z^2, the degree of weight modification of the two matrices to the space vector model cannot be determined, which should be determined according to the data set. In this paper, the matrix S and Z^2 are combined to generate a new semantic smoothing matrix C:

$$C_{ij} = \lambda_1 * S_{ij} + \lambda_2 * Z^2_{ij} \tag{15}$$

where S_{ij} and Z^2_{ij} are described in Sections 3.2 and 3.3, λ_1 and λ_2 adjust the normalization parameters of the weights in S and Z^2. Parameters λ_1 and λ_2 satisfy $\lambda_1 + \lambda_2 = 1$. We can adjust λ_1 and λ_2 to determine how matrices S and Z^2 affect the classification performance of the classifier.

3.4 Semantic Smoothing Kernel Function

By defining the mapping architecture matrix C, we can map a document vector to a new feature space vector by using Equation (16).

$$\overline{\phi}(d_j) = \phi(d_j)C. \tag{16}$$

The mapped vectors can be directly used in many classification methods. If the high-dimensional sparse matrix appears in the text classification, there will be a high-dimensional disaster in computation. Defining a kernel function can reduce the influence of the high-dimensional sparse matrix. The inner product between documents p and q in the feature space is computed by the kernel function using Equation (17).

$$K_{CK}(d_p, d_q) =< \overline{\phi}(d_p), \overline{\phi}(d_q) >= \phi(d_p)CC^T\phi(d_q)^T \tag{17}$$

where $K_{CK}(d_p, d_q)$ denotes the similarity of document d_p and document d_q. $\overline{\phi}(d_p)$ and $\overline{\phi}(d_q)$ are the new feature space vectors of document d_p and document d_q after transformed by the semantic smoothing matrix proposed in Equation (16). The kernel function information is stored in the matrix G:

$$G_{p,q} = K_{CK}(d_p, d_q). \tag{18}$$

Then we prove the validity of the semantic smooth kernel proposed in this section. According to Mercer's theorem [27], any semi-definite function can be used as a kernel function. The semantic smoothing matrix C proposed in this paper is composed of the statistical similarity matrix S and the second-order knowledge-based similarity matrix Z^2, so the matrix C is also a symmetric matrix. The matrix S and the matrix Z^2 are the product of a matrix and its transpositions, so the matrix S and Z^2 are both semi-definite matrix, which is proved in [28]. In linear algebra, the sum of two positive semi-definite matrices is also a semi-definite matrix. Therefore, the matrix C is a semi-definite matrix, satisfying the conditions required by Mercer's theorem. The kernel function can be constructed by the semantic smoothing matrix C.

4 EXPERIMENTS AND RESULTS

4.1 Corpora

This paper selects the corpus provided by the Sogou Company[1] and the Fudan University[2]. The Sogou corpus consists of SogouCA and SogouCS news corpora containing various categories of 2 909 551 news articles, of which about 2 644 110 articles contain both a title and relevant content. We manually categorize articles by using the channel in URL, then we get a large Chinese corpus with the article contents and categories. However, there are some categories that contain few articles. So 5 categories with largest number – "sports", "finance", "entertainment", "automobile" and "technology" are finally selected for our text classification experiments. The details of the Sogou corpus are presented in Table 1. During the training process, many parameters in the model are involved. In order to determine the optimal values of parameters in this model, we use the validation set. We partition training set, validation set and test set in 8:1:1 proportions in Sogou corpora after we shuffled the corpora. So the corpora is randomly divided into training set, validation set and test set. These parameters are described in detail in Section 4.4.

To validate the combine kernel's effect on a small corpora, we also use the corpus provided by the Fudan University. The corpora contains 9 804 articles that have been already divided into 20 categories. We choose 5 categories – "economy", "sports", "environment", "politics" and "agriculture". The details of the Fudan training set

[1] `http://www.sogou.com/labs/resource/list_news.php`
[2] `http://www.nlpir.org/download/tc-corpus-answer.rar`

Category	Total	Train	Validation	Test
Sports	645 931	80 000	10 000	10 000
Finance	315 551	80 000	10 000	10 000
Entertainment	160 409	80 000	10 000	10 000
Automobile	167 647	80 000	10 000	10 000
Technology	188 111	80 000	10 000	10 000

Table 1. Sogou News corpora

are presented in Table 2. We partition training set, validation set and test set in 7:1:1 proportions in Fudan corpus after we shuffled the corpora.

Category	Total	Train	Validation	Test
Economy	1 589	700	100	100
Sports	1 188	700	100	100
Environment	1 022	700	100	100
Politics	1 013	700	100	100
Agriculture	992	700	100	100

Table 2. Fudan University corpora

4.2 Word Segmentation and Stop Words

The current English word segmentation tool has been well developed, while the Chinese word segmentation technology is still evolving. For Python language, NLTK [29] nltk.tokenize module can be used for word segmentation in English, and jieba tool can be used for word segmentation in Chinese.

In order to save storage and improve the efficiency of classification, the classification system will ignore certain words after word classification, which are called stop words[3]. There are two kinds of stop words: the first one can be found everywhere in all kinds of documents with which the classification system cannot guarantee the true classification result. The second kind of stop words includes the modal particle, adverb, preposition, conjunction and so on.

4.3 Feature Word Extraction

In the problem of text categorization, a certain feature word and its class obey the CHI square distribution. The larger the CHI value is, the more the CHI value can be used to identify the category. The CHI formula is given:

$$\chi^2(t,c) = \frac{N(AD-BC)^2}{(A+C)(A+B)(B+D)(C+D)} \tag{19}$$

[3] `https://github.com/Irvinglove/Chinese_stop_words/blob/master/stopwords.txt`

where N is the number of texts in the training set, A is the number of documents belonging to class c and containing the word w; B is the number of documents that do not belong to class c but contain word w; C is the number of documents belonging to c class, but not containing the word w; D is the number of documents that do not belong to class c and do not contain the word w.

4.4 Experiment Settings

The classifier uses the SVM function provided in the machine learning library sklearn [30] in Python environment. We change the kernel function by using the interface it provides. We observe how the statistical similarity matrix S and the second-order knowledge-based similarity matrix Z^2 affect the performance of the classifier when the ratio of training set and parameter λ are different.

In the experiment, we set parameter values based on the experience gained from the validation set. First, we describe several parameters mentioned in Section 2.2. We compute word similarity using α 1.6, β_1 0.5, β_2 0.2, β_3 0.17 and β_4 0.13. Second, the length of VSM used for representing Sogou corpus is 10 000, and Fudan corpus is 1 000. Sogou corpus is large, so the feature vectors need to be longer. Finally, we set some parameters of the model. Penalty parameter C of the error term is 1.0 and there is no hard limit on iterations within solver, so that the SVM algorithm will stop training before over-fitting.

A number of parameters have been used to assess the performance of classification model output, such as accuracy [31] and f-measure (F1) [32]. To demonstrate that the combined kernel does improve the accuracy and F1 value of text classification, we use other machine learning methods for comparison, including KNN, Naive Bayes, and SVM with linear kernel and RBF kernel. The corpora is processed in the same way in Section 4.2 and Section 4.3, and then we call the interface of different machine learning algorithms in sklearn. We compare the results with character-level convolutional [33] networks which is the state-of-the-art method in text classification. Finally we adopt the accuracy and the F1 value of text classification as the evaluation standard.

4.5 Experiments and Results

As shown in Tables 3 and 4, the first column in the table shows the compared training algorithms, and the first row indicates the value of λ_1. The value of λ_2 corresponding to this is $1 - \lambda_1$. The second row shows the performance of the combined kernel in the case of different λ_1 value. The rest rows represent the performance of other machine learning methods. The values in Table 3 represent the accuracy rate of Sogou corpus and the values in Table 4 represent the values of F1 of Sogou corpus.

The values obtained in Tables 3 and 4 are shown by the line chart in Figures 1 and 2, respectively, from which it is easier to see the effect of the combination of the statistical similarity matrix S and the second-order knowledge-based similarity matrix Z^2 on the classification accuracy.

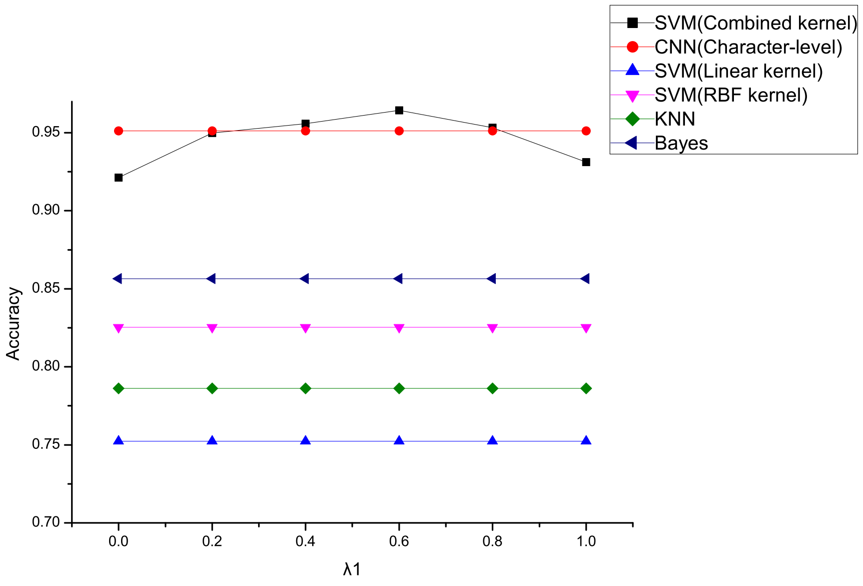


Figure 1. Curve: accuracy rate of Sogou corpus

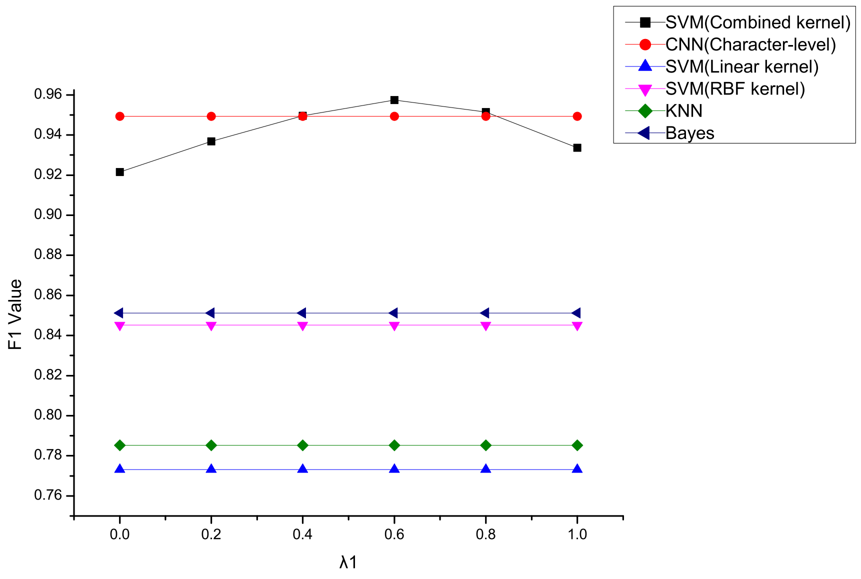


Figure 2. Curve: F1 value of Sogou corpus

	0	0.2	0.4	0.6	0.8	1
SVM (Combined kernel)	92.12 %	94.98 %	95.58 %	96.43 %	95.32 %	93.12 %
CNN (Character-level)	95.12 %					
SVM (Linear kernel)	75.23 %					
SVM (RBF kernel)	82.53 %					
KNN	78.62 %					
Bayes	85.65 %					

Table 3. Accuracy rate of Sogou corpus

	0	0.2	0.4	0.6	0.8	1
SVM (Combined kernel)	92.16 %	93.68 %	94.96 %	95.75 %	95.14 %	93.36 %
CNN (Character-level)	94.93 %					
SVM (linear kernel)	77.32 %					
SVM (RBF kernel)	84.52 %					
KNN	78.53 %					
Bayes	85.12 %					

Table 4. F1 value of Sogou corpus

As shown in the Figure 1, the accuracy rate is lower than that using character-level convolutional networks which is a very effective classification method in text categorization when λ_1 is 0, 0.2, and 1. However, the accuracy rate can be maintained at a high level when λ_1 is between 0.4 and 0.8. And the accuracy rate is always higher than that using KNN, Bayes and SVM with linear kernel and RBF kernel, proving the combination of the two is meaningful for Chinese text classification.

The values in Table 5 represent the accuracy rate of Fudan corpus and the values in Table 6 represent the values of F1 of Fudan corpus. The values obtained in Tables 5 and 6 are shown by the line chart in Figures 3 and 4, from which we can confirm that the combination of the two kernels is meaningful for Chinese text classification.

	0	0.2	0.4	0.6	0.8	1
SVM (Combined kernel)	95.34 %	95.75 %	96.68 %	96.23 %	95.56 %	95.14 %
CNN (Character-level)	95.15 %					
SVM (linear kernel)	89.41 %					
SVM (RBF kernel)	94.32 %					
KNN	90.21 %					
Bayes	95.59 %					

Table 5. Accuracy rate of Fudan corpus

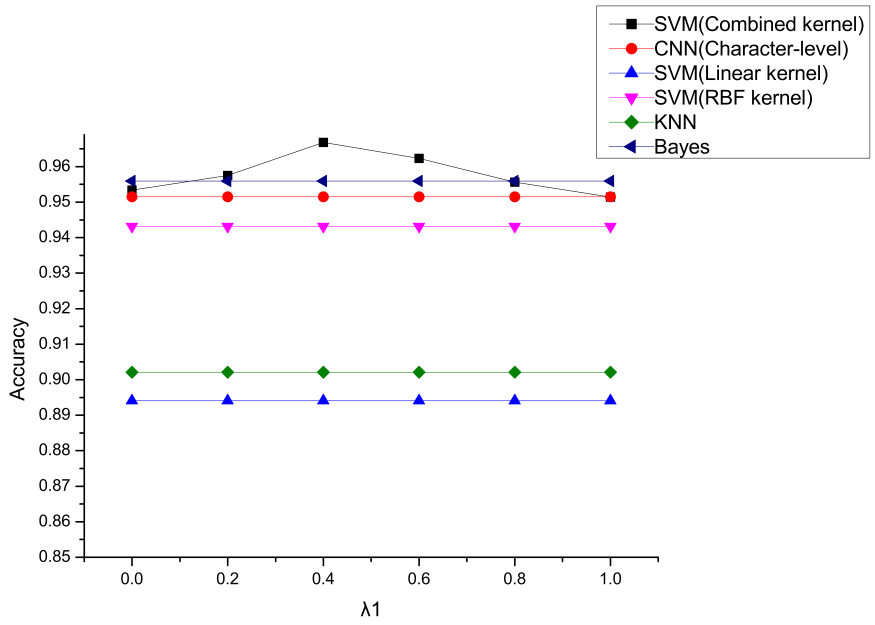


Figure 3. Curve: accuracy rate of Fudan corpus

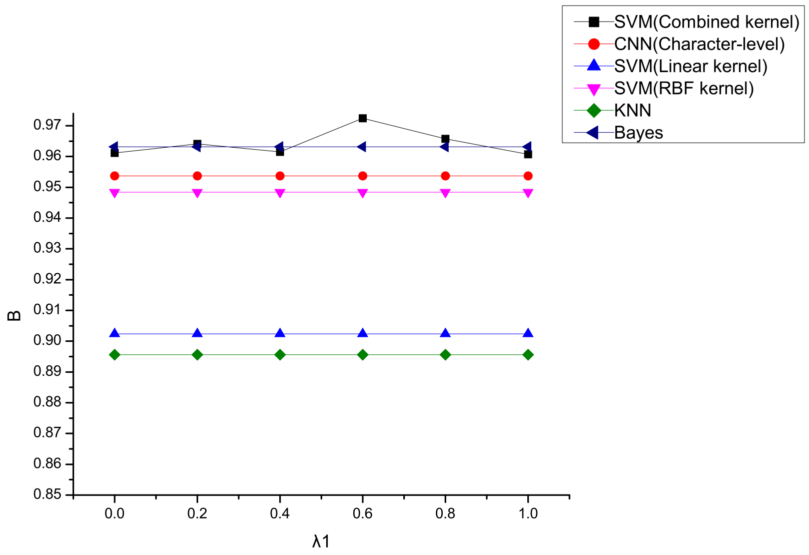


Figure 4. Curve: F1 value of Fudan corpus

	0	0.2	0.4	0.6	0.8	1
SVM (Combined kernel)	96.12 %	96.41 %	96.15 %	97.24 %	96.58 %	96.07 %
CNN (Character-level)	95.37 %					
SVM (linear kernel)	90.24 %					
SVM(RBF kernel)	94.84 %					
KNN	89.56 %					
Bayes	96.32 %					

Table 6. F1 value of Fudan corpus

5 CONCLUSION AND FUTURE WORKS

In this paper, a new combined kernel function is proposed based on semantic similarity and corpus similarity. Experiments show that the proposed method can improve the accuracy of classification compared with traditional machine learning methods.

In the future, we plan to study the statistical similarity matrix and how to capture the semantic information based on the weighting calculation. And we plan to further optimize the Chinese word-based similarity calculation method based on the HowNet.

Acknowledgements

This work is supported by the Shandong Provincial Natural Science Foundation, China (Grant No. ZR2014FQ018), BUPT-SICE Excellent Graduate Students Innovation Fund, National Natural Science Foundation of China (Grant No. 61471056). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] Dumais, S.—Platt, J.—Heckerman, D.—Sahami, M.: Inductive Learning Algorithms and Representations for Text Categorization. Proceedings of the Seventh International Conference on Information and Knowledge Management (CIKM '98), ACM, 1998, pp. 148–155, doi: 10.1145/288627.288651.

[2] Le, Q. V.—Mikolov, T.: Distributed Representations of Sentences and Documents. International Conference on Machine Learning, June 22–24, 2014, Bejing, China. Proceedings of Machine Learning Research (PMLR), Vol. 32, 2014, No. 2, pp. 1188–1196.

[3] Pennington, J.—Socher, R.—Manning, C. D.: GloVe: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543, doi: 10.3115/v1/D14-1162.

[4] Boser, B. E.—Guyon, I. M.—Vapnik, V. N.: A Training Algorithm for Optimal Margin Classifiers. Proceedings of the Fifth Annual Workshop on Computational Learning Theory (COLT '92), ACM, 1992, pp. 144–152, doi: 10.1145/130385.130401.

[5] Sain, S. R.: The Nature of Statistical Learning Theory. Technometrics, Vol. 38, 1996, No. 4, pp. 409–409.

[6] Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In: Nédellec, C., Rouveirol, C. (Eds.): Machine Learning: ECML-98. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 1398, 1998, pp. 137–142, doi: 10.1007/BFb0026683.

[7] Amari, S.-I.—Wu, S.: Improving Support Vector Machine Classifiers by Modifying Kernel Functions. Neural Networks, Vol. 12, 1999, No. 6, pp. 783–789, doi: 10.1016/S0893-6080(99)00032-5.

[8] Nasir, J. A.—Karim, A.—Tsatsaronis, G.—Varlamis, I.: A Knowledge-Based Semantic Kernel for Text Classification. In: Grossi, R., Sebastiani, F., Silvestri, F. (Eds.): String Processing and Information Retrieval (SPIRE 2011). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7024, 2011, pp. 261–266.

[9] Altinel, B.—Diri, B.—Ganiz, M. C.: A Novel Semantic Smoothing Kernel for Text Classification with Class-Based Weighting. Knowledge-Based Systems, Vol. 89, 2015, pp. 265–277, doi: 10.1016/j.knosys.2015.07.008.

[10] Siolas, G.—d'Alché-Buc, F.: Support Vector Machines Based on a Semantic Kernel for Text Categorization. Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000), 2000, IEEE, Vol. 5, pp. 205–209, doi: 10.1109/IJCNN.2000.861458.

[11] Mavroeidis, D.—Tsatsaronis, G.—Vazirgiannis, M.—Theobald, M.—Weikum, G.: Word Sense Disambiguation for Exploiting Hierarchical Thesauri in Text Classification. In: Jorge, A. M., Torgo, L., Brazdil, P., Camacho, R., Gama, J. (Eds.): Knowledge Discovery in Databases: PKDD 2005. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 3721, 2005, pp. 181–192.

[12] Fellbaum, C.—Miller, G.: WordNet: An Electronic Lexical Database. MIT Press, 1998.

[13] Zhu, Y.-L.—Min, J.—Zhou, Y.-Q.—Huang, X.-J.—Wu, L.-D.: Semantic Orientation Computing Based on HowNet. Journal of Chinese Information Processing, Vol. 20, 2006, No. 1, pp. 14–20.

[14] Zhang, P.-Y.: A HowNet-Based Semantic Relatedness Kernel for Text Classification. Indonesian Journal of Electrical Engineering and Computer Science (TELKOMNIKA), Vol. 11, 2013, No. 4, pp. 1909–1915.

[15] Mei, J. L.: Tongyi ci Cilin. Shangai Cishu Chubanshe, 1985.

[16] Evangelopoulos, N. E.: Latent Semantic Analysis. Wiley Interdisciplinary Reviews, Cognitive Science, Vol. 4, 2013, No. 6, p. 683–692, doi: 10.1002/wcs.1254.

[17] Altinel, B., Ganiz, M. C.—Diri, B.: A Novel Higher-Order Semantic Kernel for Text Classification. 2013 International Conference on Electronics, Computer and Computation (ICECCO), 2013, IEEE, pp. 216–219, doi: 10.1109/ICECCO.2013.6718267.

[18] Altinel, B.—Ganiz, M. C.—Diri, B.: A Semantic Kernel for Text Classification Based on Iterative Higher-Order Relations Between Words and Documents. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L. A.,

Zurada, J. M. (Eds.): Artificial Intelligence and Soft Computing (ICAISC 2014). Springer, Cham, Lecture Notes in Computer Science, Vol. 8467, 2014, pp. 505–517.

[19] Altinel, B.—Ganiz, M. C.—Diri, B.: A Simple Semantic Kernel Approach for SVM Using Higher-Order Paths. Proceedings of the 2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA), IEEE, 2014, pp. 431–435, doi: 10.1109/INISTA.2014.6873656.

[20] Altinel, B.—Ganiz, M. C.—Diri, B.: A Corpus-Based Semantic Kernel for Text Classification by Using Meaning Values of Terms. Engineering Applications of Artificial Intelligence, Vol. 43, 2015, pp 54–66, doi: 10.1016/j.engappai.2015.03.015.

[21] Sparck Jones, K.: A Statistical Interpretation of Term Specificity and Its Application in Retrieval. Journal of Documentation, Vol. 28, 1972, No. 1, pp. 11–21, doi: 10.1108/eb026526.

[22] Salton, G.—Buckley, C.: Term-Weighting Approaches in Automatic Text Retrieval. Information Processing and Management, Vol. 24, 1988, No. 5, pp. 513–523, doi: 10.1016/0306-4573(88)90021-0.

[23] Ko, Y.—Seo, J.: Automatic Text Categorization by Unsupervised Learning. Proceedings of the 18th Conference on Computational Linguistics, Vol. 1, Association for Computational Linguistics, 2000, pp. 453–459, doi: 10.3115/990820.990886.

[24] Lertnattee, V.—Theeramunkong, T.: Analysis of Inverse Class Frequency in Centroid-Based Text Classification. IEEE International Symposium on Communications and Information Technology (ISCIT 2004), 2004, Vol. 2, pp. 1171–1176, doi: 10.1109/ISCIT.2004.1413903.

[25] Biricik, G.—Diri, B.—Sönmez, A. C.: A New Method for Attribute Extraction with Application on Text Classification. Fifth International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control (ICSCCW 2009), IEEE, 2009, pp. 1–4, doi: 10.1109/ICSCCW.2009.5379479.

[26] Biricik, G.—Diri, B.—Sönmez, A. C.: Abstract Feature Extraction for Text Classification. Turkish Journal of Electrical Engineering and Computer Sciences, Vol. 20, 2012, No. 1, pp. 1137–1159.

[27] Parsons, S.: Introduction to Machine Learning by Ethem Alpaydin, MIT Press, 0-262-01211-1, 400 pp. The Knowledge Engineering Review, Vol. 20, 2005, No. 4, pp. 432–433.

[28] Cristianini, N.—Shawe-Taylor, J.—Lodhi, H.: Latent Semantic Kernels. Journal of Intelligent Information Systems, Vol. 18, 2002, No. 2-3, pp. 127–152.

[29] Loper, E.—Bird, S.: NLTK: The Natural Language Toolkit. ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics (ETMTNLP '02), Vol. 1, 2002, pp. 63–70.

[30] Pedregosa, F.—Varoquaux, G.—Gramfort, A.—Michel, V.—Thirion, B.—Grisel, O.—Blondel, M.—Prettenhofer, P.—Weiss, R.—Dubourg, V.—Vanderplas, J.—Passos, A.—Cournapeau, D.—Brucher, M.—Perrot, M.—Duchesnay, E.: SciKit-Learn: Machine Learning in Python. Journal of Machine Learning Research, Vol. 12, 2011, No. 10, pp. 2825–2830.

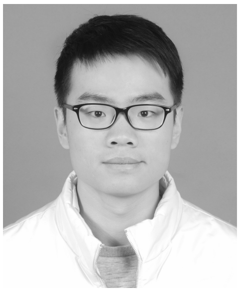
[31] El Kourdi, M.—Bensaid, A.—Rachidi, T.-E.: Automatic Arabic Document Categorization Based on the Naïve Bayes Algorithm. The Workshop on Computa-

tional Approaches to Arabic Script-Based Languages (Semitic '04), 2004, pp. 51–58, doi: 10.3115/1621804.1621819.

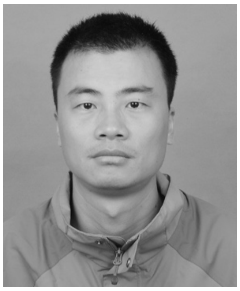
[32] Syiam, M. M.—Fayed, Z.T.—Habib, M. B.: An Intelligent System for Arabic Text Categorization. International Journal of Cooperative Information Systems, Vol. 6, 2006, No. 1, pp. 1–19.

[33] Zhang, X.—Zhao, J.—LeCun, Y.: Character-Level Convolutional Networks for Text Classification. Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS '15), Vol. 1, 2015, pp. 649–657.

Haipeng Yao is currently Associate Professor with the School of Information and Communication Engineering, from the State Key Laboratory of Networking and Switching Technology in Beijing University of Posts and Telecommunications. His main research interests include future network architecture, big data for networking, the architecture and key technology of the new generation mobile communication system.



Bo Zhang is currently Postgraduate Student with the School of Information and Communication Engineering, from the State Key Laboratory of Networking and Switching Technology in Beijing University of Posts and Telecommunications. His main research interests include natural language processing, big data and deep learning for networking.



Peiying Zhang received his Master degree from China University of Petroleum (East China) in 2006. He is currently Lecturer in the College of Computer and Communication Engineering from China University of Petroleum (East China). He is a Ph.D. candidate in information and communication engineering, from the State Key Laboratory of Networking and Switching Technology in Beijing University of Posts and Telecommunications. His research interests include natural language processing, semantic computing, future internet architecture, network virtualisation, and data center network.

Maozhen Li is currently Professor in the Department of Electronic and Computer Engineering at Brunel University London, UK. He received his Ph.D. degree from the Institute of Software, Chinese Academy of Sciences in 1997. He was Post-Doctoral Research Fellow in the School of Computer Science and Informatics, Cardiff University, UK in 1999–2002. His research interests are in the areas of high performance computing, big data analytics and intelligent systems. He is on the Editorial Boards of a number of journals. He has over 150 research publications in these areas. He is a Fellow of the British Computer Society.

Computing and Informatics, Vol. 37, 2018, 1011–1036, doi: 10.4149/cai_2018_4_1011

INFORMATION TECHNOLOGY AND PRAGMATIC ANALYSIS

Pavol Božek

Institute of Production Technologies
Faculty of Materials Science and Technology
Slovak University of Technology
J. Bottu 25, 917 24 Trnava, Slovakia
e-mail: pavol.bozek@stuba.sk

Alexander Lozhkin

Department of Software
M. T. Kalashnikov Izhevsk State Technical University
Studencheskaya st., 7, Izhevsk, 426069, Russia
e-mail: lag.izh@gmail.com

Alena Galajdová

Department of Automation, Control and Human Machine Interactions
Faculty of Mechanical Engineering
Technical University of Košice
Letná 9, 042 00 Košice, Slovakia
e-mail: alena.galajdova@tuke.sk

Igor Arkhipov, Konstantin Maiorov

Department of Software
M. T. Kalashnikov Izhevsk State Technical University
Studencheskaya st., 7, Izhevsk, 426069, Russia
e-mail: aio1024@mail.ru, gibiskus@gmail.com

Abstract. Similarity method has been in science for several centuries. The basis of the study is closely connected with mathematical linguistics. This approach has allowed obtaining new results in the analytical geometry, which, in turn, is used in different applications in information technology. The results are described briefly. Binary relations in linguistics and geometry are compared with the position of system analysis. The modified hypothesis of space as a binary structure is put forward on the basis of singular linear transformations. The hypothesis of the human sensory system is given shortly. Architecture computing appliance for solving this class of problems is proposed. The modified method is also applied in pattern recognition. The presence of symmetry in natural languages is shown briefly.

Keywords: Symmetry, characteristic equation, singular transformations, sensor system, binary relations

Mathematics Subject Classification 2010: 68T30, 93A10, 51N20

1 INTRODUCTION

The software and hardware of the computer are determined by the theory of automata. Semiotic analysis of formal languages is the basis for software. There are two approaches to semiotic analysis, distinguished by the number and structural components: a proposal of Academician A. P. Ershov [1] and the theory of the American mathematical linguists [2]. Theory of formal languages involves three stages of semiotic analysis: morphological (lexical), syntactic and semiotic ones. Ershov does not share the analysis of the first two stages and offers another – pragmatic option. A group of scientists called "pragmatics is research" exists among Scandinavian information technology scholars. Pragmatic analysis as a method of study might not be supported, but the author's team uses it for a long time.

Linguists of natural languages offer the following definition of pragmatics: pragmatos (from Greek = business, action) is a branch of science (semiotics, linguistics) studying the functioning of linguistic signs in speech. We will expand the scope of this definition to information technology and not just linguistics. In fairness, we note that Ershov's structure was for the first time proposed by Charles Morris [3]. There is a distinction between the purpose of the work of Morris and Ershov. Morris developed linguistics and philosophy and Ershov wrote works for its actual implementation using a computer.

Modern linguists and scholars in the field of cognitive research divide pragmatics into two components: functional analysis (natural language, not mathematics) and directly pragmatic analysis [4]. Tasks of functional analysis are mathematically coinciding with the objectives of algebra. We have a lot of algebra in mathematics and information technology, such as universal, relational, etc., so this stage of solving problems is redundant for mathematical methods. Absolutely new knowledge to obtain is a complex process. It is hard to compete with a genius of the previous centuries. Let us try to get new results based on work from earlier ages and compare them with those of the XX century.

Main studies postulate ideas, following Galileo's quote: Mathematics is the language in which God has written the universe. Semantics moved from the last word to the entire aphorism. Similarity method of Leibniz allowed making a lot of discoveries in various fields of science, but it has been forgotten nowadays. H. Weyl argued that the similarity method is based on symmetries. Galileo's aphorism with Weyl-Leibniz's addition in the current study can be formulated like: the automorphisms of the space define the knowledge, on the basis of which the universe is built. But this idea is not a new one. Einstein said that his research is based on the harmony of the world. Analytic geometry was chosen as the base of research for its simplicity, accessibility and allowing multiple applications.

2 SEMIOTIC ANALYSIS OF MACHINE BUILDING DRAWING

The method was described first for applying to the problems of verifying vector geometric model after input from the digitizer in the early 80s. It singled to consider machine building drawing as a text. V. A. Zvegintsev proposed the levels of natural language study in linguistic semantics in the classical linguistics. He singled seven levels [5]: a sentence, phrase (syntagma), word, morpheme, syllable, phoneme, and a distinctive feature. The text (discourse) is included additionally as the eighth level of the synthesis. The second most important result of this study is the idea of binary structure of the levels of study. Structure with six levels of study of machine building drawing language was proposed on the Zvegintsev's basis [6]: a drawing, form, cut, geometric shape, point, and a hypercomplex number or scalar. The geometry and pattern recognition theory can alternate in the structure each by each. The method of semiotic analysis by A. P. Ershov interpretation is chosen for solving the verification problems. The means is the most powerful method of artificial intelligence according to the authors. The method is dividing into three components [1]: syntax, semantics and pragmatics. Semantics studies the relation of the structures on different levels. Communication construction of the language with the text subject is called pragmatics. All sections are combined in semiotics. An additional level is considered in natural languages and theory of programming languages [2]. The structure of the language by Zvegintsev was used to communicate with the computer in the natural language [5]. These assumptions were used for mechanical drawing language. The rule was allocated to resolve uncertainties [6] of the study structure. Each level of the study with the number was semantically related to the level and it is the rule of the relations of levels. The law applies to artificial intelligence and could not be proved in the last century. This rule and semiotic analysis are allowed both to conduct research on semiotics of language drawings and solve the problems of clarifying the geometric model.

Let us consider any area of knowledge $\mathbf{K}$. Let us propose that knowledge was produced on two theories: $\mathbf{K}'_1$ and $\mathbf{K}'_1$. Let the theory $\mathbf{K}'_1$ have of study levels $\mathbf{L}_{ij}$, where i is a number of theory, j is a routine number of study levels. The question is how to deduce new knowledge from the existing knowledge. This question is solved by the methods of logical deduction, anthologies, etc., but the relationship between knowledges has not been identified so far. Completeness and consistency of the acquired knowledge is tested poorly. Incomplete knowledge of the initial theories also can adversely affect the output. Check the completeness of the basic theory, the generation of new knowledge, and and we get the relations between different theories offered by the similarity method. The levels of the study can be further detailed and explained by various mathematical descriptions. Let any binary normalisation f be applied to the rule $\mathbf{L}'_{ij}$ so that $\mathbf{L}'_{ij} = f(\mathbf{L}_{ij})$, where $\mathbf{L}_{ij} \in B$, B is the Boolean lattice. Nowadays, the structural linguistics may be used to the study levels (symmetries in Euclidean plane) $\mathbf{L}'_{ij}$ by Zvegintsev's interpretation. Relationships of $\mathbf{L}'_{1j}$ and $\mathbf{L}'_{2i}$ is the main question in theory $\mathbf{K}$.

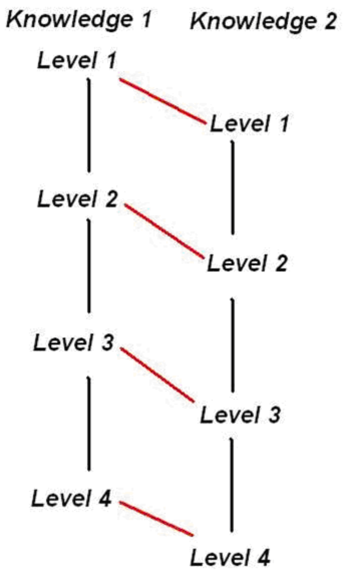


Figure 1. Knowledge symmetry

We may suggest that the study levels in generic theory $\mathbf{K}$ are positioned by law: $\mathbf{L}'_{11}$, $\mathbf{L}'_{21}$, $\mathbf{L}'_{12}$, $\mathbf{L}'_{22}$, …, $\mathbf{L}'_{1i}$, $\mathbf{L}'_{2i}$ (knowledge symmetry Figure 1) with the analogy of H. Weyl transfer symmetry [7]. Any rule $\mathbf{L}'_{ji}$ is unique. The practical use of it shows that if the cardinality of a set is different then the addition rules appear in law: $\mathbf{L}'_{11}$, $\mathbf{L}'_{21}$, $\mathbf{L}'_{12}$, $\mathbf{L}'_{22}$, …, $\mathbf{L}'_{1m}$, $\mathbf{L}'_{2n}$, where $m \neq n$. Let us define the rule of equality of power studying level $m = n$ by simplifying. Study level $\mathbf{L}'_{ij}$ for knowledge area $\mathbf{K_i}$ may be defined in another descriptive language. This notation seems to be distant from mathematics.

Therefore, syntax rules must be formulated:

1. Binary rule $\mathbf{L'}_{ij} \in B$;
2. Rule of definition punctuality. For example, the rule of accessories of an element in the set by Fraenkel. ZFC-axiomatics (Zermelo-Fraenkel-Curatowski) follows from this notation.
3. Implication rule: $\mathbf{s}_{ijk} \rightarrow \mathbf{s}_{ij+2m} = 1$, where semantic rule $\mathbf{s}_{ijk} \in \mathbf{L'}_{ij}$.

The structure of knowledge symmetry is very similar to the structure of DNA but one can understand more of it once it was discovered for the first time in nature. Analytic geometry does not allow solving all problems to increase the accuracy of geometric modeling. The differential and algebraic geometry helped only in minor matters. Authors present the theory of analytic geometry from the standpoint of artificial intelligence and relational algebra. Let us try to consider the symmetry from a new viewpoint.

3 AUTOMORPHISM OF KNOWLEDGE AS THE MAIN SYMMETRY

Axioms for Euclidean plane was formulated by Hilbert. He suggested that the construction of linguistic rules must be considered additionally. We obtain the Euclidean plane as a text by analogy with semiotic analysis of drawings according to Leibniz's method of similarity. Levels of study of the text are internal relations in plane. The following basic postulates were used: permutation, mirror, and unitary matrix symmetries by Dieudonne [8]; table automorphisms and transfer symmetry by H. Weyl [7]; definition of symmetry by M. Born [9]; and binary automorphisms by F. Bachmann [10].

Cartesian product was studied within limits of ZF-set theory (Zermelo-Fraenkel) because the set $\{\langle x, y \rangle, x\}$ created asymmetry in $\mathbf{R}^2$. Let us consider the ZFC-theory of sets without Curatowski's axiom. This theory was used in many applications quite a few times, but this supposition is in conflict with ZF-theory due to its primary property: connections by Codd and set by A. Fraenkel. ZF-theory was built from the automorphic rule $a \in A$ [12]. This rule is senseless without relation $\in$. The relational algebra is not used if $A \equiv \emptyset$. Therefore, relational algebra is the part of ZFC-theory.

The Euclidean plane is a relation table. The proof is easy, because relation algebra can work with both finite and indefinite tables. The conduct of symmetry was considered by relation algebra and semiotic analysis. Application of the method to the Dieudonne automorphisms shows that binary symmetry belongs to two mathematical disciplines: the set theory and the universal algebra.

Extended table of Dieudonne symmetries was built on the basis of knowledge symmetry and relational algebra:

1. Existence of set ($A \neq \emptyset$ Zermelo).
2. Existence of relation ($a_1\mathbf{R}a_2$ Codd).
3. Membership element of set ($a \in A$ Fraenkel).
4. Universal relation ($f : \Omega \rightarrow \Omega'$ implication).
5. Linguistic description of the set (Descartes).
6. Linguistic presentation of the relation (Descartes).
7. Perdurability cardinality ($\mathbf{m}(A) = const$ Lagrange).
8. Perdurability power relations ($n = const$ in $C_1x^n + C_2y^n + C_3x^{n-1}y^{n-1} + \ldots + C_{k-1}x + C_ky + C_0$ Klein).
9. Linguistic order ($\vec{v} = xi + yj + zk + w$ Hamilton).
10. Mathematical order ($a_i \prec a_{i+1}$, where $a_i, a_{i+1} \in \mathbf{R}$ Cantor).
11. Permutation ($a_i \leftrightarrow a_j$).
12. Mirror ($a_i \bullet -1 = -a_i$).

Since the connection between automorphisms 10 and 12 was for the first time opened by Gilbert, the symmetry of structure symmetries (knowledge) may be named in his honour. Gilbert opened the structural properties of symmetry, and Euler's formula $e^{i\pi} = -1$ is to determine the relationship between them in universal algebra. The main feature of the table is a hierarchy. The higher symmetries determine the manifestations of lower automorphisms. Symmetry with numbers one and two are combined on the plane in the unitary matrix. The unitary features are derived from this.

The table joined symmetries by Dieudonne (1, 2, 11, 12) and Weyl (10). Zvegintsev proposed to consider the floors for dividing the levels in the semantic load. The first floor is the floor of existence. It combines the first four symmetries and flows out of the unitary matrix. The last four automorphisms can be combined as numeric. Their influence on the plane (space) is particularly bright. Middle symmetries do not have a unifying semantics. It can be assumed that there may be more automorphisms. The presence of additional symmetries follows from the discovery of Klein (automorphism 8) and is confirmed by Klein's and Born's invariants theory. The completeness of the symmetries system may be insufficient. Additionally, we have noted an important property of the table that a dual symmetry is arranged twice in each floor.

Thus, mathematical linguistics, relational algebra and structural linguistic analysis allowed defining new types of symmetry. Let us try to analyse how accurately the new symmetries describe the nature. The analysis should begin obviously with the analytical geometry.

4 APPLICATION IN ANALYTIC GEOMETRY

One of the biggest problems in the geometry which has a wide application in science and engineering is to find the solution of the characteristic equation $\mathbf{T}\vec{v} = \lambda\vec{v}$, where

T – the matrix of transformations, $\vec{v}$ – vector, λ – set of scalar. The solution of the characteristic equation is required for many sciences. It is particularly important in cryptography.

A proper angle of rotation of the quadratic form α for the linear transformation is usually considered. Mirror symmetry is determined by the set **R**, and there is no permutation symmetry. Permutation symmetry is determined on a direct $x = y$ on plane. Hence the angle α exists with this direction for the angle β by symmetry. The set of unit vectors along these angles gives its own non-orthogonal basis (Figure 2).

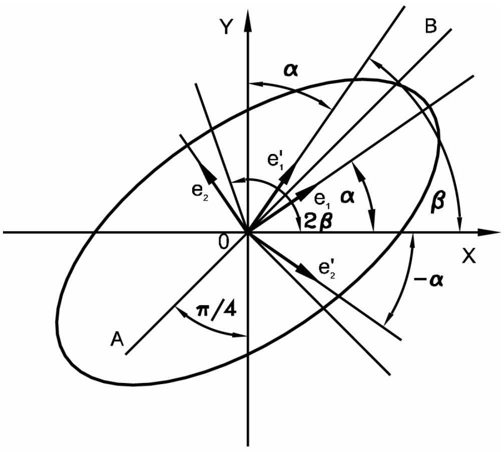


Figure 2. Own non-orthogonal basis

The symmetries table is combined of the two sciences, so there are two methods for solving the trajectories calculation. Direct analytical method of linear transformations of central symmetric conic sections was found [13, 14]. Elliptic and hyperbolic trajectory can be calculated using non-projective transformations by the classical method [15] or direct analytical method. Non-projective solutions for Jordan curves are absent.

Let there be an arbitrary figure Φ – Jordan curve in the Euclidean plane $\mathbf{R} \times \mathbf{R}$ in a Cartesian coordinate system defined by the parametric equation

$$\begin{cases} x = f_x(t), \\ y = f_y(t) \end{cases} \tag{1}$$

where $x, y, t \in \mathbf{R}$, $t \in [-\pi, \pi]$. Functions $f_x(t)$ and $f_y(t)$ are piecewise continuous. If the equation's figure is defined $y = f(t)$, it is always possible to write $\begin{cases} x = t \\ y = f(t) \end{cases}$.
Class of shapes defined only by the implicit function $F(x, y) = 0$ will be not considered in this research. We carry out any transformation of the figure Φ defined

by the matrix $\mathbf{T} = \begin{pmatrix} a & h \\ g & b \end{pmatrix}$, where $a, b, h, g \in \mathbf{R}$. It is necessary to obtain the parameters of the transformed figure (to solve the characteristic equation).

Let us consider the solution of the characteristic equation for the centre symmetric conical sections, where the own α angle takes the form [15]:

$$\tan 2\alpha = \frac{2\,(bh + ag)}{(a^2 + h^2) - (b^2 + g^2)}. \tag{2}$$

Parameters semiaxes are considered as difficult using the classical method since they represent a radical dependence.

A new direct analytical method for the linear transformation was proposed earlier. It is free from radicals, so it is more simple and accessible for further mathematical derivations. The method is based on the permutation symmetry and other symmetries [8, 14].

We calculate the angle β

$$\tan 2\beta = \frac{2(gb + ha)}{a^2 - h^2 - b^2 + g^2} \tag{3}$$

in the first step. The angle α is determined from the two equations

$$\tan \alpha_1 = \frac{a \tan \beta - h}{b - g \tan \beta} \tag{4}$$

and

$$\tan \alpha_2 = \frac{b \tan \beta + g}{a + h \tan \beta}. \tag{5}$$

Angles are equal if the calculation and transformation are correct. The coefficients $\lambda_1 = \lambda_2 = d$ are equal

$$c_1 = \frac{a \cos \beta + h \sin \beta}{\cos \alpha} \tag{6}$$

and

$$c_2 = \frac{b \sin \beta + g \cos \beta}{\sin \alpha}, \tag{7}$$

$$d_1 = \frac{a \sin \beta - h \cos \beta}{\sin \alpha} \tag{8}$$

and

$$d_2 = \frac{b \cos \beta - g \sin \beta}{\cos \alpha}. \tag{9}$$

The initial system will be: $\begin{cases} x = c_1 \cos(t + \alpha) \\ y = d_1 \sin(t + \alpha) \end{cases}$ and $\begin{cases} x = c_2 \cos(t + \alpha) \\ y = d_2 \sin(t + \alpha) \end{cases}$. Both solutions are valid and the choice is not necessary. Semiaxes are equivalent $c_1 = c_2$

and $d_1 = d_2$. For example, let us consider the motion of a point located outside the axis of a flat rod. The system of equations describing the motion of the third point is $\begin{cases} x = e\cos t + c\sin t \\ y = c\cos t + (d-e)\sin t \end{cases}$, where d – crank length, e, c – coordinates of the point (x, y) in the coordinate system of the flat rod. Solution using proposed method is $\begin{cases} x = (d-e+c\cot\alpha)\cos(t+\alpha) \\ y = (e-c\cot\alpha)\sin(t+\alpha) \end{cases}$ and $\begin{cases} x = (e+c\tan\alpha)\cos(t+\alpha) \\ y = (d-e-c\tan\alpha)\sin(t+\alpha) \end{cases}$, where $\tan 2\alpha = 2c/(2e-d)$.

Solution using classical method for the transformation matrix $T = \begin{pmatrix} e & c \\ c & d-e \end{pmatrix}$ by the inverse matrix $T^{-1} = \frac{1}{\det T}\begin{pmatrix} d-e & -c \\ -c & e \end{pmatrix}$ is $\begin{cases} x = \lambda_1\cos(t+\alpha) \\ y = \lambda_2\sin(t+\alpha) \end{cases}$ or $\begin{cases} x = \lambda_2\cos(t+\alpha) \\ y = \lambda_1\sin(t+\alpha) \end{cases}$, where $\lambda_{1,2} = \frac{d\pm\sqrt{d^2-4((d-e)e-c^2)}}{2((d-e)e-c^2)}$. Angle α is not changed.

As can be seen from the results, the proposed method allows to be used in subsequent calculations easily. For example, for solving differential equations.

Unfortunately, general solution for Jordan curve could not be obtained. The solution may be found when four systems are obtained [16]. The characteristic parametrical system $\mathbf{T}\vec{v} = \lambda\vec{v}'$, where $\mathbf{T}$ – matrix of transformations, $\vec{v}$ – vector, λ – set of scalar, is used in many fields of science, such as mechanics, physics, economics, cryptography, etc. The main result of research by the authors is the vector's belongs $\vec{v}' \in \left\{\begin{pmatrix} x \\ y \end{pmatrix}, \begin{pmatrix} -x \\ -y \end{pmatrix}, \begin{pmatrix} y \\ x \end{pmatrix}, \begin{pmatrix} -y \\ -x \end{pmatrix}\right\}$. The result of the conversion depends on the permutation and mirror symmetries. We continue to study the general case [16]. Point change direction is reversed for some linear transformation additionally. Geometric modeling transformations were carried out separately (Figure 3). The calculation method is verified on mechatronic systems in the laboratory [16, 17, 18, 19].

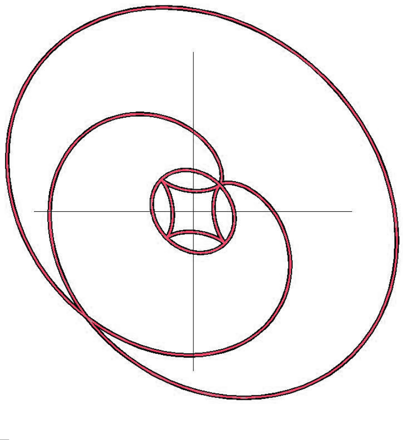


Figure 3. Complex mechatronic system

5 SINGULAR TRANSFORMATIONS

The differences between the methods are described in references [20]. Singular conversions are paramount. The confluent conversion transforms the plane in a straight line [15]. This definition by Efimov allows distinguishing six groups of linear transformations: $S_1 = \begin{pmatrix} a & a \\ b & b \end{pmatrix}$, $S_2 = \begin{pmatrix} a & b \\ a & b \end{pmatrix}$, $S_3 = \begin{pmatrix} 0 & a \\ 0 & b \end{pmatrix}$, $S_4 = \begin{pmatrix} a & 0 \\ b & 0 \end{pmatrix}$, $S_5 = \begin{pmatrix} a & b \\ 0 & 0 \end{pmatrix}$, $S_6 = \begin{pmatrix} 0 & 0 \\ a & b \end{pmatrix}$, where $a, b \in \mathbf{R}$ and $a, b \neq 0$. Groups define the singular transformation of universal algebra automorphisms.

Theorem about algebraical singularity 1. Any singular transformation in geometry algebraic branch of arbitrary figure - Jordan curve belonging to the plane described by the parametric system of equations with continuous functions defined on the interval of the real axis, transforms this figure in a straight line, a straight line segment, or a ray.

Proof. Let there be an arbitrary figure Φ to be the Jordan curve in the Euclidean plane $\mathbf{R}^2$ in a Cartesian coordinate system defined by the parametric equation. Functions $f_x(t)$ and $f_y(t)$ are piecewise continuous.

Let us apply singular transformation $S_1 = \begin{pmatrix} a & a \\ b & b \end{pmatrix}$, $S_2 = \begin{pmatrix} a & b \\ a & b \end{pmatrix}$, $S_3 = \begin{pmatrix} 0 & a \\ 0 & b \end{pmatrix}$, $S_4 = \begin{pmatrix} a & 0 \\ b & 0 \end{pmatrix}$, $S_5 = \begin{pmatrix} a & b \\ 0 & 0 \end{pmatrix}$, $S_6 = \begin{pmatrix} 0 & 0 \\ a & b \end{pmatrix}$, where $a, b \in \mathbf{R}$ and $a, b \neq 0$.

Let us consider the transformation $S_1 = \begin{pmatrix} a & a \\ b & b \end{pmatrix}$. The resulting system of equations is $\begin{cases} x' = af_x(t) + af_y(t) \\ y' = bf_y(t) + bf_x(t) \end{cases}$, $\begin{cases} x' = a(f_x(t) + f_y(t)) \\ y' = b(f_y(t) + f_x(t)) \end{cases}$. Let us consider three arbitrary parameter values $t \in \{t_1, t_2, t_3\}$, where $t_1 \neq t_2 \neq t_3$, and substitute them in the system. Let us choose the settings so that the function $f_x(t_i)$ and $f_y(t_i)$ had no breakage and parameters $t_j \neq t_k$ were not equal to each other. Let the function $f(t)$ to be the sum of functions $f_x(t) + f_y(t)$, then the coordinates of the points will be given by the vectors $v_i = \begin{pmatrix} af(t_i) \\ bf(t_i) \end{pmatrix}$. If we take the result of the conversion to get a straight line, then any three points lie on it and the determinant $\begin{vmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{vmatrix} = 0$. We form the determinant D of the obtained $D = \begin{vmatrix} af(t_2) - af(t_1) & bf(t_2) - bf(t_1) \\ af(t_3) - af(t_1) & bf(t_3) - bf(t_1) \end{vmatrix}$. Let us disclose the determinant: $D = \begin{vmatrix} a(f(t_2) - f(t_1)) & b(f(t_2) - f(t_1)) \\ a(f(t_3) - f(t_1)) & b(f(t_3) - f(t_1)) \end{vmatrix}$, $D = a(f(t_2) - f(t_1))b(f(t_3) - f(t_1)) - a(f(t_3) - f(t_1))b(f(t_2) - f(t_1))$, $D = ab((f(t_2) - f(t_1))(f(t_3) - f(t_1)) - (f(t_3) - f(t_1))(f(t_2)$

$- f(t_1)))$. The determinant is $D = 0$. Consequently, a straight line is obtained by a singular transformation $S_1 = \begin{pmatrix} a & a \\ b & b \end{pmatrix}$ always.

Let us consider the kind of figures obtained after the conversion. Values in line coefficients are expressed by the determinant $\begin{vmatrix} af(t_2) - af(t_1) & bf(t_2) - bf(t_1) \\ x - af(t_1) & y - bf(t_1) \end{vmatrix} = 0$, $(af(t_2)-af(t_1))(y-bf(t_1))-(x-af(t_1))(bf(t_2)-bf(t_1)) = 0$, $(af(t_2)-af(t_1))y-bf(t_1)(af(t_2) - af(t_1)) - (bf(t_2) - bf(t_1))x + af(t_1)(bf(t_2) - bf(t_1)) = 0$, $a(f(t_2) - f(t_1))y - b(f(t_2) - f(t_1))x = 0$. We turn out a few equations of lines for different values of the parameters t_1 and t_2, since the coefficients depend on the magnitude $f(t_2) - f(t_1)$. This problem is solved, if we consider the normal equation of the line, which in this case would be $\frac{a}{\sqrt{a^2+b^2}}y - \frac{b}{\sqrt{a^2+b^2}}x = 0$.

Coordinate of a point t_0 different from the point t_1 by a small amount ε such as $f_x(t_0) - f_x(t_0 + \varepsilon) = \varepsilon_x$ and $f_y(t_0) - f_y(t_0 + \varepsilon) = \varepsilon_y$, where ε_x and ε_y are small quantities, define by linearly dependent conversion S_1 for functions $f_x(t)$ and $f_y(t)$ on the continuous range $[t_2, t_3]$.

Let us consider the difference in the coordinate x is $x_{t_0} - x_{t_1} = a(f_x(t_0) + f_y(t_0))-a(f_x(t_1)+f_y(t_1))$, $x_{t_0}-x_{t_1} = a\left((f_x(t_0) - f_x(t_0 + \varepsilon)) + (f_y(t_0) - f_y(t_0 + \varepsilon))\right)$, $x_{t_0} - x_{t_1} = a(\varepsilon_x+\varepsilon_y)$. The difference coordinate y similarly to be equal to $y_{t_0} - y_{t_1} = b(\varepsilon_x + \varepsilon_y)$. Since $\varepsilon \approx \varepsilon_x \approx \varepsilon_y$ then $x_{t_0} - x_{t_1} \approx 2a\varepsilon$ and $y_{t_0} - y_{t_1} \approx 2b\varepsilon$. Let us select an infinitely small ε, so that $1/\varepsilon \gg 2\max(a, b)$. It follows that the line differential is not equal to the differentials of functions $f_x(t)$ and $f_y(t)$, but a sequence of points is provided. Like in previous reasoning, if there is a parameter t_0 to the function $f_x(t)$ or $f_y(t)$ in which at least one function has a break, get a direct break at the point $x = a\left(f_x(t_0) + f_y(t_0)\right)$ and $y = b\left(f_x(t_0) + f_y(t_0)\right)$.

Branch proved for a singular transformation S_1. The proof is similar for the other groups. The full text of the theorem is presented by Theorem 2 [21]. $\square$

We have not reviewed the singular groups $S_7 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$ and $S_8 = \begin{pmatrix} \infty & \infty \\ \infty & \infty \end{pmatrix}$. These groups will transform the figure into a point. The zero rates is manifested in different ways in the singular transformations. Therefore two groups are proposed.

Bachmann proposed to consider automorphisms of the binary, but the binary relations are absent in the classical axioms of Euclidean space. Let us consider the function of the algebra of statements by their properties. We assign a variety of functions to the first subset $BF_1 = \{x, y, \bar{x}, \bar{y}, x \oplus y, x \equiv y, 0, 1\}$. They depend only on a single utterance. Truth and false are not dependent on the statements, but we will consider them as belonging to this set. Power of the first subset is equal to 8. Let us consider that all other functions belong to the second set BF_2. A second plurality of power is equal to the first power.

Let us look at the singular transformations of universal algebra branch in information – linguistic interpretation of the geometry [13, 21]. Automorphisms trans-

Value	Function Name	Dependence on Two Variables
0	Identical zero	no
1	Pierce's arrow	yes
2	Inversion of direct implication	yes
3	Inverse of the second operand	no
4	Inversion reverse implication	yes
5	Inversion of the first operand	no
6	Excluding or	no
7	Sheffer's stroke	yes
8	Conjunction	yes
9	Equivalence	no
10	First operand	no
11	Backward implication	yes
12	Second operand	no
13	Implication	yes
14	Disjunction	yes
15	Tautology	no

Table 1. Boolean functions

form any curve in a couple of lines (line segments, rays) in the plane. Let us look at the properties of a given set of functions in detail. Each function is subject to the law, and then there is a symmetry in the classical definition. The result of the singular transformation of the curve in the geometry set-theoretic branch depends on the type of the curve. The number of elementary functions (line, trigonometric function, shot, etc.) is limited and less than 8, which makes the discovery of new elementary functions possible. Thus, it is possible to build a much-to-one correspondence between the propositional algebra and singular linear transformations in their properties, algebra utterances take features of a single point in space. The theoretical basis of a computer are algebra statements. Since $\mathbf{R}^0 = \mathbf{Z}^0 = \mathbf{C}^0 = \ldots$ then a computer can work with different types of data: numbers, text, images, etc. The versatility of a computer prevents effective way of recognition, which is doubled at least.

Let us consider the dependence of Boolean two-place functions on the number of operands in more detail (Table 1). The structure of the table for this parameter differs from the table of automorphisms of the Euclidean plane at first glance. Let us assume that Boolean functions are located in three-dimensional space. Let their order determine a double helix similar to the DNA structure. Then the non-orthogonal projection onto some plane gives an alternation similar to the automorphism's table. A special case of the location of functions with values of seven and eight. We are dealing with additional symmetry. It allows traversing both from top to bottom and back.

Mathematical basis for the nature computer should consider the relational algebra as possible. It is unclear what to do with the theory of automata for such a device. Building it with the current theoretical framework is difficult.

The binary principle of the structure of the language was for the first time proposed by Ferdinand de Saussure and much later by Bachmann's automorphisms. It has been applied in natural language linguistics during 150 years and it was used, for example, in Zvegintsev's works. The main conclusion of our material is that space can be studied as a pure computer science object. Every point in space can be seen as an elemental part of the global supercomputer.

Unfortunately, the authors do not offer additional results in the formulation of research. Let us try to do it in the theory of pattern recognition as an informational science closely related to the geometry.

6 SENSE OF BEAUTY

Let us start with the repeat information about the stages of development of a human vision.

The main sensoric organ, which is responsible for spatial orientation in humans, is the vision. It is generally known that there are several view types [22]. For each type of a view there is a person at a certain age range. In physiology we distinguish the age of appearance of the binocular, colour, and spatial type of a person. Binocular (two-eyed) vision appears in the first few weeks of life. Colour is formed by 4–5 years of age. The spatial orientation, especially the ability of a particular person to accurately design, for example objects of engineering, may not ever be developed. Apparently, one should speak about the absolute mechanism of symmetry on the level of reflex. Previously, there was a brief study on the presence of symmetries in the work of children [23]. Small children without colour vision have several kinds of symmetries (Figure 4).

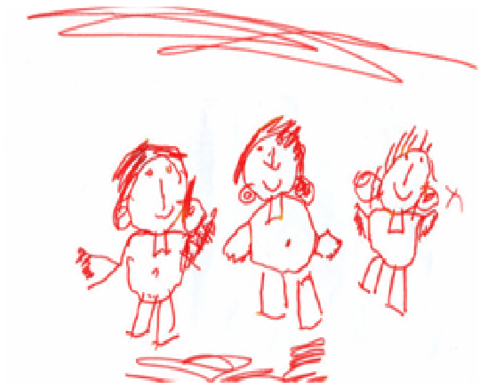


Figure 4. Picture of small children

Figures of people look alike (the symmetry of 3), but each has its own unique features. It is the hair, facial expressions, and hands (symmetry 9). The child is painted big almost as parents, only a little smaller (symmetry 10). Those are the last symmetries on the picture. The author does not have a colour vision. Work of an older child in another figure is shown for comparison (Figure 5). Colour plays an important role in this picture. We are dealing with a product of education. Girl is the biggest, mom is a little smaller, and father is very small.

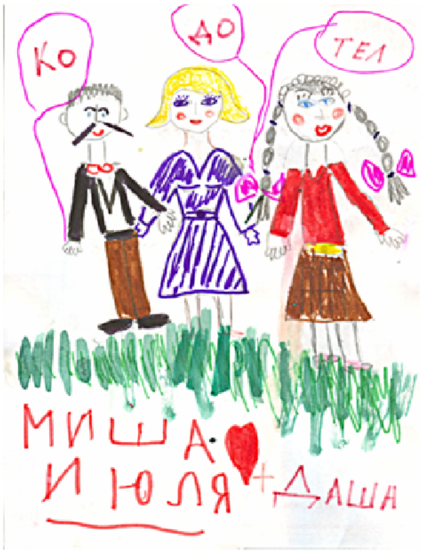


Figure 5. Children's picture

The authors are not specialists in biology, medicine or physiology, so we turn to the research of psychologists. Since the beginning of the psychology development as an independent science it appears in nonverbal tests for diagnosing mental condition of the person [24].

Rorschach offered the first nonverbal test for the analysis of human psychological state. The test consists of several pictures, but each of them presents a mirror symmetry (Figure 6). The test does not have much reliability, but it is used so far. The test uses the simplest symmetry.

Luscher Colour Test (1948) was widely adopted. The test results are frequently used to analyse the nature of the individual. Each colour in the palette is unique, so it must be concluded that Luscher uses linguistic symmetry order. The same type of test should may be psycho-geometric test by Delinzher [25]. Testing is carried out on five different pictures: a square, triangle, zigzag, ellipse, and a cross. Although geometric shapes test also uses only the uniqueness of each picture, that is, linguistic symmetry order (Figure 7).

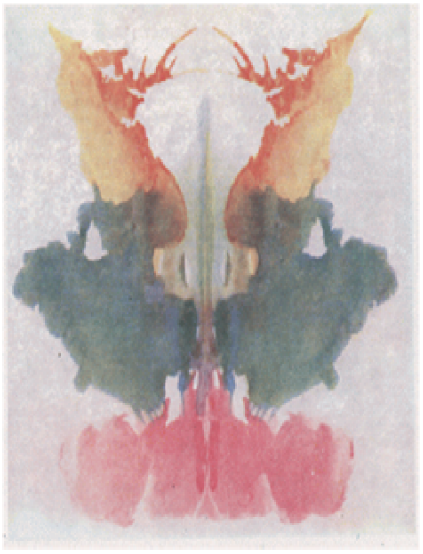


Figure 6. Picture of Rorschach

The creative tests analysing the human family and image by Karen Machover are mostly used. The test is based on the figure of a subject, so the children's creativity can be analysed with it. More symmetry is used in this test as it is a mirror, transportability, mathematical order, linguistic order, etc. [23]. The question is, when you can use these tests. German authors [26] say that the best age of a child to take the test is 4–5 years. It allows testing by the above mentioned types of psychological experiments as the child has already got some sort of sense besides colour vision.

Weyl considers to study the symmetry with poems and works of art [7]. He emphasises that beauty is defined by automorphisms. DNA structure and symmetry of knowledge are similar (see Section 3). On the contrary, the discovery of Hilbert allows explaining the structure of DNA, to express it more precisely. Therefore, symmetry defines a new feeling, known from Homer times. This is the sense of beauty.

Psychologists have found acceptable brain solutions to awareness of beauty in the recent studies [27, 28]. Such mechanism may be a comparison table of automorphisms within a man and the symmetry of the surrounding space. It can be much different from the space table regarding the age and human actions, to be harmonious or to have a dissociation with it. The appointment of a recently discovered gene can be regarded as an indirect confirmation of the hypothesis [29]. Physiological studies are lengthy, so the mechanism and the sensoric system are not known exactly. We can unambiguously confirm the fact that the human recognition mechanism depends not only on the vision.

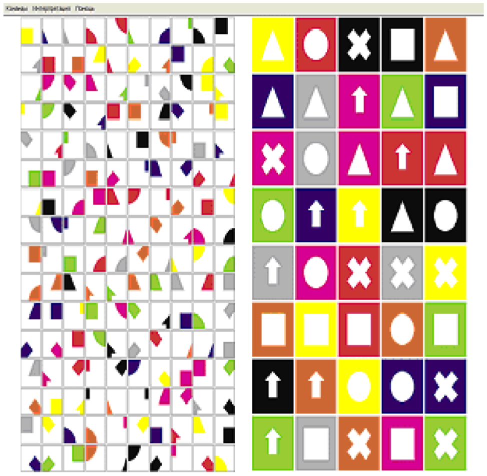


Figure 7. The combination of tests by Luscher and Dillinger

The symmetries table and DNA have the same structure and mechanism and the human touch can be present in every living cell of the human body. Based on that we have developed the hypothesis in terms of computer science saying that a man should not be seen as a central processor with peripheral equipment but as the cloud structure. The same principle can be incorporated in the pattern of recognition systems and robotics. Again we are faced with a contradiction between relational algebra and automata theory. The duality of the sensor systems is used in robots for a long time, for example, echolocation and thermal imager combined in one device. Man does not have the duality of the vision as it was previously thought. Hypotheses used in the framework of modern architecture of computing devices are possible only for software purposes.

7 SELECTING IMAGE FEATURE POINTS

Search for the image characteristics is an important task of a computer vision. Feature points can effectively accomplish the following tasks: stitching panoramas and aerial photographs (image stitching); 3D reconstruction search and identification of objects, fingerprints verification [30], etc. A large number of works in this direction [31, 32, 33] testifies indirectly the relevance of the problem of finding image feature points. Unfortunately, at present there is hardly any universal algorithm for generating an acceptable solution to the problem in different subject areas. The reason is all sorts of distortions of recorded images such as affine and projective transformations associated with the movement of the camera or objects in the scene, scene change illumination, occlusion and shadowing objects in the scene, etc. In this work, the task of finding the image feature points image has attempted to be solved in the context of a comparison of the current photograph area, made on board of the aircraft, with the search field composed of aerial photographs or satellite images. Such comparisons will enable to position the aircraft.

Every object in the nature has its specific symmetries. Points of the device orientation, in contrast to other points, have a pronounced symmetry linguistic order. However, most objects are characterised by a simple mirror symmetry. A test added for the manifestation of symmetry allows to increase the reliability of solutions of known algorithms.

The successful positioning of the aircraft needs detected feature points belonging to the objects that will be called stable area reference objects. Not all objects in the current image area can be classified as stable reference objects. In these images some moving or variable objects are often present. People and vehicles should be classified in the class of moving objects. The objects, which are easily subject to changes, such as single trees, paths, small single buildings, etc., are considered to be variable objects. The presence of these objects in the image makes it difficult to find a stable reference object. Figure 8 shows original image areas. There are roads, buildings and clusters of trees as stable reference objects. Cars, trails, single trees and bushes, hedges are changeable.

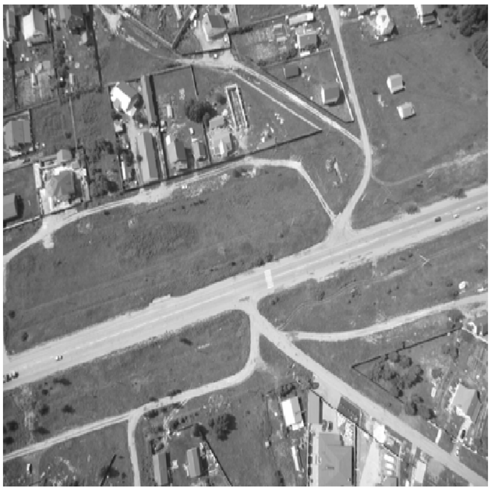


Figure 8. The source image of terrain

To reduce the number of moving and changing objects in an image we use the median image filtering with the $N \times N$ pixel aperture, which allows preserving the boundaries of the major objects in an image [34].

The value of the aperture should be selected with the following considerations. The maximum aperture size of the median filter is dependent on the minimum width of the road. The minimum size of the median filter aperture is calculated from the condition of suppression of moving objects such as buses, passenger cars and trucks. Depending on the scale and resolution of the shooting aperture value typically ranges from 3 to 9 pixels.

Median filtering suppressed all vehicles, narrow dirt roads, small elements of vegetation and small buildings. In addition, the image preserved road network, large buildings and groups of trees. It is necessary to emphasise the preservation of boundaries among the objects which remained on the image.

However, even if the boundary of the object area is close to the ideal, after sampling it becomes unsharp and dim which makes it more complicated to highlight the contours of objects. To restore the boundaries an extreme filter is used [34]. When the extreme filter is used it calculates the distance between the current brightness of the input image pixel brightness and extreme values in the neighborhood of $(x,y)\begin{cases} d_{\min}(x,y) = \left|b_1(x,y) - e^N_{\min}(x,y)\right| \\ d_{\max}(x,y) = \left|b_1(x,y) - e^N_{\max}(x,y)\right| \end{cases}$, where $e^N_{\min}(x,y)$ and $e^N_{\max}(x,y)$ are the minimum and maximum extreme values of brightness in the $N \times N$ pixel vicinity of the current image.

The value of the extreme (maximum or minimum), closest to the brightness of the central pixel, is assigned to the current pixel of the output image $b_2(x,y) = \min\left(d_{\min}(x,y); d_{\max}(x,y)\right)$.

As a result, the boundary gets aggravated and actually becomes a step one, which highly facilitates the construction of the circuit stable area landmarks. Extreme filter aperture size is set to the minimum value of 3×3. However, it has been found by experiments that slightly better results can be obtained by increasing the filter extreme aperture to median filter aperture size while significantly increasing the cost of computational complexity. Further, any available contour detector builds the contours of objects. In this paper, we used Canny detector [35].

In addition, the curvature function for each circuit is calculated [36]. The choice of the curvature function as a basis for feature points is due to its invariance to the shift, rotation and lighting conditions. If we do not use the information about the value of curvature at the point of its extreme for the further comparison, and will only fix the coordinates of extreme points, the proposed method will have an invariance and scale changes.

Curvature function is one-dimensional and requires unbranched circuit, but the description of complex objects may call for contour branching. In this case, the branching point becomes the anchor point, and the outline of a complex object becomes an integral of multiple curvature functions. Positive and negative curvature function extremes exceed a predetermined threshold, the markers are marked on the original image. It results in an image of the current location frame marked up with feature points. If the number of feature points is not enough to match the current frame with the search area image or the comparison is made with a low reliability, the extreme threshold of curvature function is adaptively reduced and the extreme search is repeated. A large number of isolated feature points can significantly slow down the process of further comparison. In this case, raising the extreme search threshold is provided. Checks revealing points on the above symmetry are carried out at the end of the process.

Figure 9 shows the original image, and areas marked thereon are the singular points found as described above. Figure 9 shows that all the major junctions and bends of the roads are marked by feature points. The rest of the feature points are located at the corners of buildings and in the bends of the outlines of large clusters of trees. At the same time, cars, small buildings, light fencing, single trees and bushes remained untagged. Modification methods described apparatus can also speed up the algorithm.

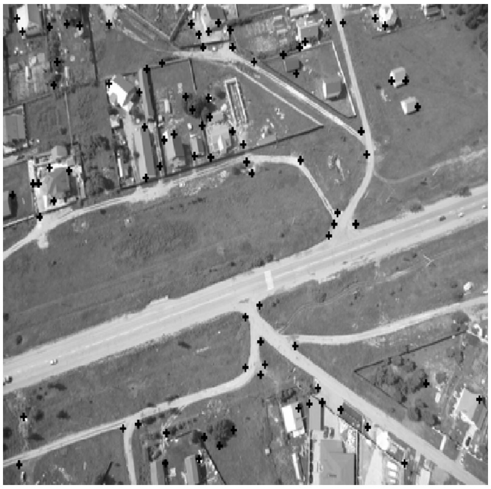


Figure 9. The source image labeled by feature points

We looked at several actual tasks of informatics on the basis of mathematical linguistics. It would be logical to turn to the analysis of a natural language texts.

8 AUTOMORPHISMS AND UNIVERSALS

We started from linguistics, and finished there as well. Let us consider how natural language texts will meet the proposed table automorphisms. By the linguistic research we understand not only the semiotic analysis of texts, but natural languages and linguistics in general.

Formation of semiotics as a science is associated with the name of C. S. Pierce [37], the son of an eminent mathematician who obtained mathematical education. Erlangen program and similarity method had a great influence on the development of language linguistics, too [38]. However, unfortunately, the modern principle of humanitarian knowledge representation does not allow to apply precisely the results of linguistics in solving mathematical problems. Different natural languages have different study structures [4]. Common mechanism of abstraction [39] developed in the 30's, 40's is rarely used.

Regarding linguistics in general, regarding any language, many terms are invariant, such as a "synonym", "homonym" and "antonym". This category is called linguistic universals [39]. The binary nature of structural analysis in the interpretation of Zvegintsev allowed for the past forty years to apply his theory in information technology, and proposed a binary linguistic principle long ago [39].

We begin our consideration with universals from antonyms. Antonyms are the words where one word has an opposite meaning [39, p. 36]. Antonym is the ratio of lexical units having the opposite meaning, for example "cold" and "hot". Let A be the set of words that have antonyms and C be a lot of antonyms from A. Antonyms describe the attitude $f_{An} : a_i \to c_j$. Let shelf ratio f_{An} twice: $f_{An} : a_i \to c_j$, $f_{An} : c_j \to a_i$. Therefore, it is a symmetry (automorphism) in a limited definition [7]. Of course, due to the presence of considered table automorphisms [14, 17] in the latter case may be $f_{An} : c_j \to a_i$ or $f_{An} : c_j \to a_k$, when $k \neq i$. This process is random. If we continue the transformation, then in Markov process of random variables, we always get $f_{An} : c_j \to a_i$. That is the symmetry determined by M. Born [9]. This type of automorphism is comparable with mirror symmetry of the Euclidean plane and defined by the sign n and $-n$.

Synonym is a word of the same part of speech, with fully or partially identical meaning/values [39, p. 447], for example "buy" and "purchase". Let A be a lot of words – synonyms of one concept. Dual-use relationship of synonymy also leads to the original state of the text, either directly or by a Markov process. The mathematical operation of a set theory to describe the relationship of synonymy is $a_i \leftrightarrow a_k$, when $k \neq i$. Consequently, synonymy reflects the permutation symmetry. Other set with automorphism of the domain forms the Markov process. Markov process mechanism for the conservation of automorphisms is possible.

Homonym is a word which matches different linguistic units values of which are not related to each other [39, p. 344]. For example, mouse is an animal and mouse is a device. This is the most complex form of language universals. The Euclidean dimensional space described by Cartesian product $\mathbf{R}_1 \times \mathbf{R}_2 \times \ldots \times \mathbf{R}_n$, with the description in the literature can be $\mathbf{R} \times \mathbf{R} \times \ldots \times \mathbf{R}$. The unique name of the coordinates is lost, but the property of uniqueness of each coordinate axis is preserved. Consequently, homonym reflects the symmetry of a linguistic order.

Thus, the three from four low automorphisms of the Euclidean plane are used in linguistics. The four is only an automorphism defined by a mathematical order sets $\mathbf{Z}$, $\mathbf{R}$, $\mathbf{C}$, etc. Let us recall that most alphabets have a strict order of the characters. Ideographic and logographic systems display the language [39, p. 29] transmitting the knowledge of the sign system used in a particular order as well. The learning process is always built from a simple level to a complex one. Moreover, the basis of the phoneme alphabet and their number varies in different languages from 10 to 80 [39, p. 291]. Each phoneme has its own frequency sound vibrations. Therefore, they are arranged in a natural way. Cardinality of the set should not confuse us. The screen display is not more than 2 000 pixels along a single axis. Nevertheless, we can receive complex geometric images, watch videos, watch sports in real time, etc. However, this type of automorphism can be regarded as a manifestation of the second linguistic order.

Referring to senior automorphisms. chief automorphism of the existence of not empty set (Zermelo) for the language is correct. The second most important is the existence of relations as well. Without symmetry universal set-theoretic assertions semiotics is not possible. Symmetry of universal algebraic relation differs from the consumed in the Euclidean plane. Any grammar contains the rules of inference $A \to B$, where A, B are any (terminal or nonterminal in formal languages terms) symbol.

Euler's formula $e^{i\pi} = -1$ was replaced by implication. Number of phonemes transmitting sound statement is constant as stated above. Let us consider all the world's languages and to make a general interlingual phonemic dictionary. The number of phonemes in this dictionary is limited. Thus symmetry conservation of cardinality of the set is performed. Let us consider the rules of making the grammar. Let us recall that all the characters are divided into terminal and nonterminal ones. Symmetry degenerates to preserve the exact order equaling two. This type corresponds to Klein's symmetry. Thus, most of the relevant automorphisms exist on the sets of texts.

9 CONCLUSIONS

Similarity method is one of the oldest tools of science. It has been almost forgotten now. The similarity method allowed to explore electricity on the basis of mechanics, that is, it laid the foundations of modern civilisation. The concept of an invariant by Klein entered almost inside all the fields of science. We propose to consider the method of similarity on a new level with the use of mathematical linguistics.

Methods of mathematical linguistics allow identifying a new type of symmetry, the symmetry of knowledge. Its manifestation in mathematics has been known for a long time, but it has never been considered as a major one. The relational algebra and symmetry of knowledge are brought together in the theory of symmetry by Dieudonne and Weyl, who formulated a new method of linear transformations of the plane. One result of this method is the modified concept of singular transformations.

Binary relations are used in geometry and natural language linguistics – why was it possible it is not known. The modified binarity hypothesis by De Saussure and Bachmann has been put forward on the basis of singular transformations. The space must be considered primarily as the object of information technology. Symmetries in the table should be the number of sixteen by extended hypothesis. The middle floors can encounter a semantic burden in this case. Klein's theory is mentioned briefly in the article, so that the reader's attention is not taken away from the subject matter.

The development of the person's vision system was analysed from the standpoint of the proposed theory. The authors are not specialists in the field of medicine, biology, psychology, etc., so the research is based on information technology and mathematics. We propose to consider a new understanding of a man as a computer system. This is a cloud structure, not a computer with peripheral equipment. This principle can be incorporated into the design of mechatronic systems and CAD, both on hardware and on software levels. The work is based on the principle of people-cloud, it has been applied to solve the problems with images recognition. Precision solutions can thus be increased by 7 %.

Symmetries exist in any natural language as it has been shown. This principle can be useful for scientists dealing with the problems of artificial intelligence in natural languages.

Russian researchers have developed this problem for a long time in an interdisciplinary field. The terminology of the works combines classical geometry, information technology, and some natural and human sciences. Therefore, it is necessary to appreciate the approach of Slovak scientists who understood the materials, supplemented and edited it. The most important is that they brought the theory to specific new technologies, and therefore without their participation, this article would not be written.

The article has the character of problem formulation. Specific solutions are for scientists in the field of informatics for production only, but the submitted hypothesis may be the key for other relevant sections of computer science.

Acknowledgments

This publication is the result of implementation of the project: "University Scientific Park: Campus MTF STU – CAMBO" (ITMS: 26220220179) supported by the Research & Development Operational Program funded by the EFRR and the contribution is sponsored by the project 015STU-4/2018 Specialised laboratory supported by multimedia textbook for subject "Production systems design and operation" for STU Bratislava.

REFERENCES

[1] Ershov, A. P.: Language: Mathematical Encyclopedic Dictionary. Soviet Encyclopedia, Moscow, 1988, p. 845.

[2] Aho, A. V.—Lam, M. S.—Sethi, R.—Ullman, J. D.: Compilers. Principles, Techniques and Tools. Addison Wesley, Reading, Mass., 2007.

[3] Morris, C. W.: Logical Positivism, Pragmatism and Scientific Empiricism. AMS Press, New York, 1979.

[4] Trifonas, P.P. (Ed.): International Handbook of Semiotics. Springer Netherlands, Haarlem, 2015, doi: 10.1007/978-94-017-9404-6.

[5] Zvegintsev, V. A.: The Sentence and Its Relation to Language and Speech. Moscow University, Moscow, 1976 (in Russian).

[6] Lozhkin, A. G.: Method Formalization of Semantic Relations in Language of Machine-Building Drawings. Designing and Manufacturing of Metal-Plastical Structure, IMI-UdSU, Izhevsk, 1983, pp. 81–86 (in Russian).

[7] Weyl, H.: Symmetry. Princeton University Press, Cambridge, 2016. ISBN 978-0-691-17325-2.

[8] Dieudonne, J.: Linear Algebra and Geometry. Kenhaw, London, 1983.

[9] Born, M.: My Life: Recollections of a Nobel Laureate. Routledge Library Editions: 20th Century Science. Kindle Edition, 2014.

[10] Bachmann, F.: Aufbau der Geometrie aus dem Spiegelungsbegriff. Springer-Verlag, Berlin, 1959 (in German), doi: 10.1007/978-3-662-01234-5.

[11] Codd, E. F.: The Relational Model for Database Management: Version 2. Addison Wesley, Reading, Mass, 2000.

[12] Fraenkel, A. A.—Bar-Hillel, Y.: Foundations of Set Theory. North-Holland Publishing Company, Amsterdam, 1958.

[13] Lozhkin, A. G.: Applied Planemetry with Singular Transformations. IE UrO RAS, Ekaterinburg, 2009 (in Russian).

[14] Lozhkin, A.—Dyukina, N.: Structurization of Analytical Geometry on the Base of Symmetries. LAP, Saarbruken, 2012 (in Russian).

[15] Efimov, N. V.: Quadratic Forms and Matrices. Science, Moscow, 2012.

[16] Božek, P.—Pokorný, P.—Svetlík, J.—Lozhkin, A.—Arkhipov, I.: The Calculations of Jordan Curves Trajectory of Robot Movement. International Journal of Advanced Robotic Systems, Vol. 13, 2016, No. 5, doi: 10.1177/1729881416663665.

[17] Lozhkin, A.—Božek, P.—Lyalin, V.—Tarasov, V.—Tothova, M.—Sultanov, R.: Reverse and Direct Methods for Solving the Characteristic Equation. AIP Conference Proceedings, Vol. 1738, 2016, No. 1, doi: 10.1063/1.4954935.

[18] Božek, P.—Pivarčiová, E.: Flexible Manufacturing System with Automatic Control of Product Quality. Strojarstvo, Vol. 55, 2013, No. 3, pp. 211–221. ISSN 0562-1887.

[19] Pivarčiová, E.—Božek, P.: Industrial Production Surety Factor Increasing by a System of Fingerprint Verification. Proceedings of the International Conference on Information Science, Electronics and Electrical Engineering (ISEEE 2014), April 26–28, 2014, Sapporo City, Hokkaido, Japan. IEEE, 2014, p. 5. ISBN 978-1-4799-3197-2.

[20] Božek, P.—Ivandić, Ž.—Lozhkin, A.—Lyalin, V.—Tarasov, V.: Solutions to the Characteristic Equation for Industrial Robot's Elliptic Trajectories. Tehnicheski Vjesnik (Technical Gazette), Vol. 23, 2016, No. 4, pp. 1017–1023, doi: 10.17559/TV-20150114112458.

[21] Lozhkin, A. G.: Set-Theoretic and Information Methods of Analysis Geometric Modeling in CAD Engineering Products. Doc. thesis, ISTU, Izhevsk, 2013.

[22] Mosby's Medical Dictionary. 9^{th} Edition, Elsevier, 2013.

[23] Lozhkin, A. G.: Symmetry as a Property of Space and a Living Organism. Tietta, Vol. 3, 2010, No. 13, pp. 23–32.

[24] Gregory, R. J.: Psychological Testing: History, Principles, and Applications. Allyn & Bacon, NY, 2003. ISBN 978-0-205-35472-6.

[25] Dellinger, S. E.: Communicating Beyond Our Differences: Introducing the Psycho-Geometrics System. Prentice-Hall/Jade Ink, 1996.

[26] Beelmann, W.—Schmidt-Denter, U.: Standardization of the German Version of the Family Relations Test (FRT) for Children, Ages 4–5 Years. Psychologisches Institut der Universitet zu Keln, Lehrstuhl IV: Entwicklungs und Erziehungspsychologie, Vol. 48, 1999, No. 6, pp. 399–410 (in German).

[27] Orsini, C. A.—Moorman, D. E.—Young, J. W.—Setlow, B.—Floresco, S. B.: Neural Mechanisms Regulating Different Forms of Risk-Related Decision-Making: Insights from Animal Models. Neuroscience and Biobehavioral Reviews, Vol. 58, 2015, pp. 147–167, doi: 10.1016/j.neubiorev.2015.04.009.

[28] Slioussar, N.—Kireev, M. V.—Chernigovskaya, T. V. et al.: An ER-fMRI Study of Russian Inflectional Morphology. Brain and Language, Vol. 130, 2014, pp. 33–41, doi: 10.1016/j.bandl.2014.01.006.

[29] Chesler, A. T.—Szczot, M.—Bharucha-Goebel, D. et al.: The Role of PIEZO2 in Human Mechanosensation. The New England Journal of Medicine, Vol. 375, 2016, pp. 1355–1364, doi: 10.1056/NEJMoa1602812.

[30] Božek, P.—Pivarčiová, E.: Registration of Holographic Images Based on Integral Transformation. Computing and Informatics, Vol. 31, 2012, No. 6, pp. 1369–1383. ISSN 1335-9150.

[31] Rosten, E.—Porter, R.—Drummond, T.: Faster and Better: A Machine Learning Approach to Corner Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 32, 2010, No. 1, pp. 105–119, doi: 10.1109/TPAMI.2008.275.

[32] Goshin, E.—Fursov, V. A.: Conformed Identification in Corresponding Points Detection Problem. Computer Optics, Vol. 36, 2012, No. 1, pp. 131–135.

[33] Arkhipov, I.—Murynov, A.: Multiscale Centroid Filtration of Noisy Graphics Image. Instrumentation Engineering, Electronics and Telecommunications, 2015, Paper Book of the I International Forum IEET-2015, 2015, pp. 32–36.

[34] Szeliski, R.: Computer Vision: Algorithms and Applications. Springer, 2010.

[35] Parker, J. A.: Algorithms for Image Processing and Computer Vision. Indianapolis, Wiley Publishing Inc., 2011.

[36] Eruslanov, R. V.—Orehova, M. N.—Dubrovin, V. N.: Retroperitoneal Space Organ Segmentation from CT Images Based on the Level Set Function. Computer Optics, Vol. 39, 2015, No. 4, pp. 592–599.

[37] Crease, R. P.: Charles Sanders Peirce and the First Absolute Measurement Standard: In His Brilliant But Troubled Life, Peirce Was a Pioneer in Both Metrology and Philosophy. Physics Today, Vol. 62, 2009, No. 12, pp. 39–44, doi: 10.1063/1.3273015.

[38] Maslova V. A.: Modern Trends in Linguistics. Academy, Moscow, 2007.

[39] Linguistics. Big Encyclopedic Dictionary. Big Russian Encyclopedia, Moscow, 1998.

Pavol Božek graduated from Slovak University of Technology Bratislava, Slovakia in 1972. Diploma in engineering, Candidate of Science, Associated Professor at the Institute of Applied Informatics, Automation and Mathematics. Presently he holds the position of the Professor at the Institute of Production Technologies, Faculty of Materials Science and Technology, Slovak University of Technology, Trnava, Slovakia. Over 338 scientific publications in international journals and conference proceedings. Main fields of research: informatics, mathematics, robotics, mechatronics.

Alexander Lozhkin received his Ph.D. degree from SRI of applied mathematics of Lobachevsky Nizhegorodsky State University, Nizhny Novgorod, Russia, in 1992 and Doctor of Technology degree from M. T. Kalashnikov Izhevsk State Technical University, Izhevsk, Russia, in 2013 by specialisation in CAD/CAM and system analyses. Now he is Professor of Software Department. Over 100 scientific publications in international journals and conference proceedings. His research interests include AI, geometry, structurisation of natural and liberal arts disciplines.

Alena Galajdová is Associate Professor at the Department of Automation, Control and Human Machine Interactions, Faculty of Mechanical Engineering, Technical University of Košice. She graduated from Technical University of Košice, Faculty of Electrical Engineering in 1984. She received his doctorate from the Technical University of Košice, Faculty of Mechanical Engineering in 2009. She also works in the Access Centre as Technical Advisor for students with disability. She is member of team in Human Motion Analysis Laboratory and Ambient Intelligence Laboratory at Technical University of Košice. She is a member of Association for the Advancement of Assistive Technology in Europe (AAATE), European Design for All e-Accessibility Network (EDeAN) and ICTA Europe Rehabilitation International.

Igor Arkhipov is Associate Professor, Dean of the Institute of Informatics and Hardware, Head of the Department of Computer Software at M. T. Kalashnikov Izhevsk State Technical University. Research interests: processing and analysis of speech signals, processing, image recognition and analysis, development of software, and hardware systems.

Konstantin Maiorov is Ph.D. student of the Department of Computer Software at M. T. Kalashnikov Izhevsk State Technical University. Research interests: AI and system analysis.