Performance Analysis of a Multiuser Multipath Communication System: a Game Theoretical Approach

Béla Almási*, Tamás Balogh, János Kormos*, Balázs Kreith***

* Faculty of Informatics, University of Debrecen, Kassai u. 26, H-4028 Debrecen, Hungary, almasi.bela@inf.unideb.hu, kormos.janos@inf.unideb.hu, kreith.balazs@inf.unideb.hu

** Faculty of Economics and Business, University of Debrecen, Böszörményi u. 138, H-4032 Debrecen, Hungary, tamas.balogh@econ.unideb.hu

Abstract: The study of multipath communication technologies is a hot research area today. One natural effect of using multipath communication instead of the single path one is the higher throughput value which will result in a better performance, not only in the usual Internet communication, but also in Big Data centers where the communication infrastructure can appear as a bottleneck point of the system. In this paper, we introduce a new game theoretical model for the evaluation of multiuser-multipath communication technologies. The decision problem for the users (i.e. network clients) is studied in a multipath communication system. We develop a game theoretical model for client payoff maximization, where the decision variables for each client are defined as their path requests. Due to limited hardware performance and limited service capacity, we assume that each client's payoff depends on other clients' path requests. We apply the tools of game theory to describe equilibrium behavior of the clients in the given interaction situation. By providing two examples, we show that our model is suitable for measuring payoffs, both in money and in throughput. We also offer possible directions for the further development of our model.

Keywords: multipath communication; throughput aggregation; Data Center Networks; game theory modeling; concave games; performance analysis

1 Introduction

1.1 Overview of the Multipath Communication Technologies

The traditional Internet Communication Technology (IP), uses a single communication path between the endpoints in a communication session, since the IP address of the endnode is a part of the socket ID. In the case of multipath routing (see e.g. [1]), when the packet transmission in the Cloud is distributed by routers among multiple paths (in order to decrease the effect of congestion), the endnode uses only one of its interfaces, in the communication session. Thus, a communication session is connected to a single interface of the endnode. The currently used devices (laptops, tablets and mobile phones) usually have more than one network interface: Wi-Fi, 3 G/4 G, Bluetooth, NFC etc. (see Figure 1 below for illustration).

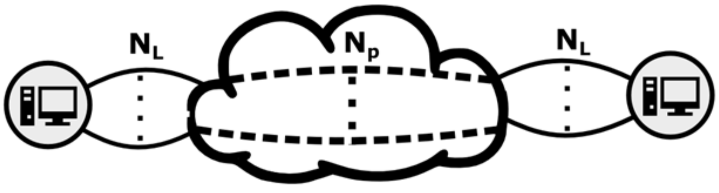


Figure 1
Illustration of multipath-multilink communication networks

The idea of combining available interfaces in one communication session is an important research topic today. The advantage of aggregating the throughput capacity of multiple interfaces is a widely used technology, also in Big Data centers where the communication infrastructure may become the bottleneck point of the overall system's performance [2]. In the modern Data Center Networks (DCNs) the throughput sensitive large flows can be effectively serviced by using multipath infrastructure background, which can be tuned according to the DCN special requirements (see [3]).

The multipath communication technology is studied in different OSI Layers: The IEEE "Convergent Digital Home Network" working group started the creation of the 1905.1 standard at the Data Link Layer. It is a special implementation of the Multipath technology (it is named as "Multilink technology" in Layer 2, as there is no multihop path in this case [4]). The 1905.1 standard focuses on the throughput aggregation possibility of the different links. The IETF RFC 6824 document "TCP Extensions for Multipath Operation with Multiple Addresses" [5] was published in January 2013. It specifies the extensions of the traditional TCP Transport Layer protocol (named as MPTCP) in order to be able to use multiple paths in one communication session. The aim of the MPTCP specification is to increase the throughput and the reliability of the TCP communication session. The MPT software library roughly targets as the MPTCP, but works at the Network Layer. One common purpose of the multipath/multilink technology is to aggregate the speed capacity of the different paths/links (i.e. throughput aggregation). Different laboratory measurements show (see [6], [7], [8]) that the multipath technology is able to efficiently aggregate the throughput capacity of the paths. In [6] Paasch et al. introduced an MPTCP based system, which produced a World Record throughput capacity of 50 Gbps by aggregating 6 pieces of 10 Gbps connection paths. Almási and Szilágyi showed an MPT based system in [7] and [8] aggregating 2 and 4 pieces of paths with the efficiency ratio better than 95% in each case.

Lencse and Kovács investigated the aggregation performance limits of the MPT multipath software library in [9] and [10]: The throughput capacity of the multipath system increased linearly with the number of paths using IPv4 (up to the systems' maximum of 12 paths with speed of 100 Mbps in each physical connection). Using IPv6, the test measurements reached the maximum limit of the aggregation in the same laboratory environment: The throughput of the system increased up to 7 paths, but then no further speed increasing could be seen even when adding more paths to the system. Considering the performance evaluation, this limitation was caused by the hardware. The overall throughput performance remained roughly constant after a special number of allocated paths. It means that allocating more paths to the system did not increase the overall throughput performance.

The performance evaluation of the multipath communication systems, their usability, utilization and consequences is also an actual research area. The utilization of MPTCP according to the client throughput aggregation to the overall network communication system is investigated in Khalili, Ramin, et al. [11]. They analyzed the performance of the MPTCP [5] communication system and in contrast to TCP based communications, they proved that it was not Pareto-optimal. A system is called Pareto-optimal if the improvement of an individual participant's benefit is impossible without decreasing another one's. In contrast to MPTCP, MPT uses a different path scheduling algorithm for link aggregation, where Pareto-optimality is not proved, but currently it serves as a basis of further investigation.

Due to the interaction of endnodes and the server in a MPT network, for modeling the overall system utilization, a game theoretical approach is also adequate.

Game theory is fundamental for modeling and evaluating systems in which the overall benefits (payoffs) of the participants depend on their individual decisions. Seminal contributions can be found in [12, 13]. Studying network communication systems by using a game theoretical approach, clients are assumed to choose their path request strategies in order to reach the best overall performance. Applying game theory tools for different network problems takes place frequently, see e.g.

[14], where cooperative and non-cooperative approaches are compared for network problems. Increasing the network capacity indefinitely and letting the routers decide the best path individually may lead to Braess's Paradox [15]. Braess's Paradox describes that in certain circumstances in which participants are choosing the best path individually without cooperation, they can actually slow down the network. It concludes that the existence of multiple paths between two nodes in the network in some circumstances can produce worse overall system utilization than using fewer paths.

The overall system utilization investigation using game theory suggests strategies on how to design or use the overall network system. As Lencse and Kovács [9] show, hardware capacity influences the aggregated bandwidth limitation at the client site, it leads us to investigate the overall system utilization further and to use game theory approaches.

A precise mathematical model of the multipath network communication system using game theoretical tools may open the possibility for many investigations related to the overall system utilization. It can be used for instance to investigate and model the path allocation strategies for multipath communication systems according to the client requests and the server interests. Other investigations may also be performed focusing on utilization and payoff interests of the participants of the system.

In this paper we introduce a non-cooperative game-theoretical framework for path allocation in a multipath network communication system to provide an analytical tool for finding the best available path allocation mechanism in the system. We combine a classic game theoretical approach with the multipath network communication system, in which clients aim at increasing their communication speeds by requesting new paths from the server. After defining the model and proving existence of equilibrium, we present two example solutions. In the first, clients are assumed to maximize their monetary payoffs which is defined as a linear combination of their utility of receiving a certain number of paths (expressed in money), and the financial costs for their path requests. The second approach considers the limited hardware capacity of the client and aims at maximizing throughput. Thus, in the second example, the payoff is not expressed as money, but as throughput.

We emphasize that the two examples employ two different, but adequate approaches to model client behavior and therefore, we do not wish to show any preferences. They can be considered as suggestions to understand and solve clients' decision problem in a multipath communication environment.

We note that our model does not yet take server payoff maximization into consideration, in order to provide clear results on the client site. A possible further research direction includes investigating the server side's decision problem. However, the server is involved in our model as well, but its role is restricted to allocate paths according to a certain allocation rule.

The main contribution of this paper is to offer a link between game theory and multipath communication systems and introduce a model that is suitable for further studies.

The rest of the paper is organized as follows… Section 2 provides the mathematical specification of multipath communication networks. Section 3 defines the path allocation game for a multipath system, offering a link between multipath communication networks and game theoretical models, where game theory is used to model the decision problem and strategic interaction of the clients in a multipath environment. Section 4 introduces two theoretical game approaches and that match the concept introduced in Section 3. We also provide the equilibrium scenario of the two games. Finally, Section 5 concludes our work.

2 Mathematical Specification of Multiuser-Multipath Systems

We define the following notations for representing the parameters of the multipath communication systems.

N_L: The number of physical links that connect the Node (client) to the Internet.

K: The maximum number of paths used by the multipath communication system

L_i : The speed (bandwidth) of the ith physical link, which connects the Node to the Internet

U_P: The throughput capacity of one path (homogeneous system)

MPSpeed(k): The speed (throughput capacity) of the system aggregating k paths.

Aggr(k): The aggregation efficiency of aggregating k paths:

$$Aggr(k) = \frac{MPSPeed(k)}{k \cdot U_P}$$

The number of physical links (N_L) is limited by the interface number of the Node. As the multipath communication system may include not totally disjoint paths (i.e. paths with common links), the number of paths may be larger than the number of physical links (i.e. $N_L < N_P$) even in the case of efficient aggregation (see [7], [8]).

Of course, the sum of the physical links' speeds gives a limit for the theoretical maximum speed of the multipath system: $N_P \cdot U_P \leq \sum_{i=1}^{N_L} L_i$.

As it was presented in [9], the aggregation needs resources from the Node (mainly CPU). The available resources are limited on the Node:

$ResMax_i$: The maximum of the available resources on the ith Node.

For simplicity, we may assume that the resource need of aggregating *k* paths (denoted by Res(k)) is linearly increasing by k: $Res(k) = k \cdot Res$.

For all k the inequality $Res(k) \leq ResMax_i$ must hold for each client *i*.

$PMax_i$: The maximal available path the Node can handle according to $ResMax_i$. If resources are infinite, then the maximal number of path the system can handle is limited by the bandwidth of the physical link: $PMax_i = \frac{\sum_{j=1}^{N_L} L_j}{U_p}$.

In what follows, we define a theoretical game framework to analyze Node (client) behavior. Later on, in Section 4 we present two models that are in coherence with the mathematical specification of the multipath communication framework.

3 The Game Theoretical Framework

We define a non-cooperative game theoretical approach for the allocation of paths in the multi-user environment. We give a formal model for Node (i.e. client, household) payoff maximization in a multipath communication environment. The main idea is that each client aims at maximizing its own profit (which we will call payoff in accordance with the game theoretic terminology). The payoff function of a certain node – precisely defined below – consists of a function of the node's total throughput and a cost function of the requested paths.

In our simplified framework, at the beginning of the allocation process, each endnode announces simultaneously the requested amount of paths. Then, according to an allocation rule, paths are assigned to the endnodes and the payoff functions are evaluated.

3.1 Definition and Assumptions of the Game

The strategic (normal) form of the game is as follows.

Definition 3.1 (Path allocation game)

The players are the endnodes which we denote by 1,2... n. The strategy set of any player i is given by $S(i) = \{0, 1, 2 \ldots K\}$, where a strategy stands for the (integer) number of requested paths, which can grow up to K. K stands for the maximum number of available paths for the whole system. For any *i*, we denote the requested number of paths by k_i. The payoff function of any player *i* is given by:

$$\pi_i(k_1, k_2 ... k_n) = f_i(k_1, k_2 ... k_n) - c_i(k_i) \tag{1}$$

where f_i is the utility of player *i* (expressed in money or throughput). Note that f_i depends directly on the received number of paths, and indirectly – as indicated in the arguments of f_i – on the requests of all endnodes. Throughout this section, for the sake of simplicity we refer to f_i as depending directly on the requested amounts of paths. At the same time, c_i is a cost function of the requested paths of player *i*.

We note that throughout the analysis we relax the logical assumption of k_i being an integer in order to create continuous payoff functions. Provided that the solution is not an integer number for a certain player (endnode), we will assume that the endnodes will choose the integer value resulting in the closest payoff level to optimal.

Before giving the solution of the game, we introduce the following assumptions on the payoff functions.

The first assumption is in accordance with the principle of decreasing marginal utility. This principle can also be applied for consumers (endnodes) in the communication environment, as a certain growth in bandwidth is worth more if the initial bandwidth of a user is smaller.

Assumption 3.2. For any *i*, f_i is strictly increasing, continuous in k_i on $[0...\mathrm{K}]$, strictly concave and twice continuously differentiable on its domain.

The following assumption dictates that if an endnode increases its number of paths requested, then its extra cost is more if the initial request was larger. This ensures that the amount of paths a household needs costs proportionally less than industrial consumption.

Assumption 3.3. For any *i*, the financial cost function c_i is strictly increasing in k_i, convex and twice continuously differentiable on its domain.

From Assumptions 3.2 and 3.3 it is trivial that the combined payoff function of each player is concave in its own path request variable. We state this in the following corollary.

Corollary 3.4. For any *i*, the combined π_i is concave and twice continuously differentiable in k_i.

If the total number of requested paths is too large, then the server cannot satisfy all needs. We define a rationing (allocation) rule to determine the number of received paths as a function of all path requests for each endnode.

Definition 3.5. Rationing (allocation) rule:

For any *i*,

$$R_i(k_1,k_2...k_n) = \begin{cases} k_i & \text{if } \sum_{i=1}^{n} k_i \le K \\ \dfrac{Kk_i}{\sum_{i=1}^{n} k_i} & \text{if } \sum_{i=1}^{n} k_i > K \end{cases} \qquad (2)$$

This means that whenever the sum of requests exceeds the available paths, paths are allocated in proportion of the requests.

We note that according to Definition 3.1 endnodes pay according to the number of their requested paths (and not the received ones).

We also note that whenever the sum of path requests does not exceed *K*, the payoff functions depend only on one variable (which is the client's own path request).

Finally, we assume that every player is aware of the strategies and payoff functions of all the others. This is in accordance with the assumption that "neighbors" know each other well enough to be able to estimate their needs.

Assumption 3.6. For any *i*, the set of players, $S(i)$ and π_i are common knowledge.

Our last assumption considers the issue of time. For simplicity reasons we assume that the game is played as a one-shot game, excluding time elapse from the model. A suitable extension of this model to real-life situations is to consider the repeated version of the game.

Now we fix the ordering of actions in the one-shot game.

Step 1. Every endnode announces its requested amount of paths simultaneously by sending the need to the server.

Step 2. After receiving the needs, the server allocates the paths to the endnodes according to the rationing rule.

It follows directly that the endnodes are not aware of each other's requests at the moment they have to send their own need to the server.

3.2 Equilibrium of the Path Allocation Game

To find the payoff-maximizing strategy of each endnode, we will employ the Nash equilibrium concept.

According to the definition, the Nash equilibrium players choose mutually beneficial response strategies for each other. (For more about the Nash equilibrium concept, we refer the reader to [16].) For the multipath environment, this means that in equilibrium every endnode is satisfied with its received amount of paths, because none of them would be able to increase their payoff level (π_i) by ceteris paribus modifying its own request.

In what follows, we cite a useful theorem for the concave games and show that the path allocation game has exactly, one Nash equilibrium in pure strategies. After proving existence, we determine the unique Nash equilibrium profile.

Theorem 3.7 (Rosen). Equilibrium of concave games (from Theorems 1 and 2 of [17])

Let us consider a game of n players given in normal form. If for any *i* the payoff function π_i is strictly concave in player *i*'s own decision variable, then the game has one and only one Nash equilibrium strategy profile in pure strategies.

The proof can be found in [17].

Corollary 3.8. The path allocation game has one and only one Nash equilibrium in pure strategies.

The proof follows directly from the assumptions that for any endnode *i*, f_i is strictly concave, while c_i is convex in k_i .

To determine the single equilibrium of the path allocation game, we use the fact that Nash equilibrium strategies are best responses for each other, thus, for any *i*, π_i is maximized by the choice of the equilibrium strategy, whenever the strategies of the other players are fixed at their equilibrium levels.

Formally, this means that the one and only one Nash equilibrium profile of the game is implicitly provided as a solution of an equation system of *n* equations.

Proposition 3.9. The following system of equations provides implicitly the only Nash equilibrium profile of the path allocation game.

$$\frac{\partial \pi_1}{\partial k_1} = 0; \ \frac{\partial \pi_2}{\partial k_2} = 0 \ \ \ldots \ \ \frac{\partial \pi_n}{\partial k_n} = 0 \tag{3}$$

From the previous result, a question might arise. Namely, for which of the function classes does the above system have an explicit solution? We omit the mathematical analysis of this problem, as in the following section, we will provide numerical results for different types of payoff functions.

In the following Section 4, we present two approaches to handle client payoff. The first one considers monetary payoffs, while in the second one, payoffs are given in throughput. Both games will match the framework we presented in Section 3, and also the features of the systems defined earlier in Section 2.

4 Models of Client Behavior

As defined in Section 3, a participant's total payoff depends on its (positive) utility and the cost function value. In a multipath communication system a client may use multiple interfaces and logical paths in order to maximize its throughput efficiency. In the case of several multipath communication (MPC) clients being connected to a common multipath capable server that has finite resources (e.g.: limited number of physical interfaces and bandwidth) the server must distribute the resources among the clients by considering their requests. According to this given allocation rule, every client wants to maximize their individual payoff regardless of the others. As described in Section 3, based on the requests sent by the clients to the server, the server distributes the requested resources. The requests can be throughput needs or a number of logical paths that the client wants to use, etc. The sum of the requests may exceed the available resources.

In what follows, a game is defined for two clients as players (*1, 2*). They send their throughput requests (R_1, R_2), to the MPT server, which calculates the number of paths to allocate to them based on their requests. For simplicity we assume, as previously described in Section 2, that a number of used resources linearly increases by the number of allocated paths distributed to the players. A player, by sending its throughput request and then participating in the game, must consider that other clients send their requests to the server as well. Every client has their own cost function limiting their payoffs. Whenever the server allocates paths from a finite resource that is exceeded by the sum of client needs, in equilibrium, one player's payoff may increase only by decreasing another player's payoff.

We note here, that the models presented, can easily be generalized for the more-than-two player cases, using the concept described in Section 3.

The payoff functions are defined as follows:

$$\pi_1(k_1,k_2)=f_1(\mathrm{R}_1(k_1,k_2))-c_1(k_1) \tag{4}$$

$$\pi_2(k_1,k_2)=f_2(\mathrm{R}_2(k_1,k_2))-c_2(k_2) \tag{5}$$

Here, k_i denotes the value of request of the *i*th player, R_i indicates the allocation rule used by the server for distributing the paths according to the players' requests, function f_i transfers throughput to utility (e.g. expressed in money) and $c_i(k_i)$ denotes i^{th} player's cost function.

If the above payoff functions are used in the framework game defined in Section 3, one and only one Nash equilibrium point exists in the game. In that case the Nash equilibrium point can be determined by finding the only mutual best response strategies of the two players, as described in Section 3. Technically, this means that both players maximize their payoffs taking into consideration the other player's strategy. Thus, the following system of equations has to be solved:

$$\frac{\partial \pi_1(k_1,k_2)}{\partial k_1} = 0 \tag{6}$$

$$\frac{\partial \pi_2(k_1,k_2)}{\partial k_2} = 0 \tag{7}$$

To calculate the equilibrium after every request has arrived to the server, we need to find the numerical solution of the equation system defined above.

In what follows, we present two examples with different payoff functions and different focus. In the first one, MPC clients aim at maximizing their individual payoffs expressed in money. This game serves as an example of how an MPC client aims at increasing its payoff related to the throughput it receives, while the payoff is limited by the cost function. In the second game, payoff functions are represented by throughput gains and the cost functions are represented as hardware resource limitations caused by the increasing number of requested paths as investigated by Lencse and Kovacs [9].

4.1 Monetary Payoffs

In this subsection we present the analytical solution of a simple, two-player version of the path allocation game. We assume that the capacity of the server is 8 paths (i.e. K=8). The payoff functions of the two endnodes are as follows (π_i always consists of a utility function and a financial cost function).

$$\pi_1(k_1,k_2) = \begin{cases} 10K\dfrac{k_1}{k_1+k_2} - k_1^2 & if\ k_1+k_2 > K \\ 10k_1 - k_1^2 & if\ k_1+k_2 \le K \end{cases} \tag{8}$$

$$\pi_2(k_1,k_2)=\begin{cases}10K\dfrac{k_1}{k_1+k_2}-k_2^{3/2} & \text{if } k_1+k_2>K\\ 10k_2-k_2^{3/2} & \text{if } k_1+k_2\le K\end{cases} \tag{9}$$

It is easy to see that the first (utility) part in the payoff function is strictly concave for both players, while the cost part is strictly convex. Therefore, π_i remains strictly concave for both players.

To decide which payoff functions provide the equilibrium, we have to solve the equation system generated by the simpler one and check whether the sum of path requests are below 8 or not. From the latter payoff functions we obtain

$$\frac{\partial\pi_1(k_1,k_2)}{\partial k_1}=10-2k_1\;=0 \tag{10}$$

$$\frac{\partial\pi_2(k_1,k_2)}{\partial k_2}=10-\frac{3}{2}\sqrt{k_2}\;=0 \tag{11}$$

It follows directly that in equilibrium, $k_1=5;\ k_2=44\frac{4}{9}$. The sum of the requests exceeds 8, therefore the former, more complicated payoff functions, should be used to find the equilibrium. Thus, we obtain the following system of equations:

$$\frac{\partial\pi_1(k_1,k_2)}{\partial k_1}=80\frac{k_2}{(k_1+k_2)^2}-2k_1\;=0 \tag{12}$$

$$\frac{\partial\pi_2(k_1,k_2)}{\partial k_2}=80\frac{k_1}{(k_1+k_2)^2}-\frac{3}{2}\sqrt{k_2}\;=0 \tag{13}$$

We obtain, that in equilibrium, $k_1=3.041;\ k_2=5.334$. If we require integer values, then the solution is 3 and 5, respectively. This result is seemingly far from that of the former equation system, but we note that the previous solution did not take into consideration the server's capacity limit. And finally, it does not matter, where the unconditional optimal level lies, if the sum of requests exceeds *K*. The following table presents the payoff function values of the two players when receiving certain amounts of paths.

Clearly, the only Nash equilibrium point is played if Player 1 requests 3 paths, while Player 2 requests 5 paths. This can easily be justified by analyzing the previous Table 1. Here, at the (3;5) allocation profile, it is in neither of the players interest to alter their strategies individually, while the other player's strategy remains fixed. This is because neither of the players can increase their payoff by a unilateral modification of her request.

Table 1

Payoff function values according to the number of requested paths of players 1;2

k2 / k1	2	3	4	5	6
2	16.00;17.17	16.00;24.80	16.00;32.00	16.00;38.82	16.00;45.30
3	21.00;17,17	21.00;24.80	21.00;32.00	**21.00;38.82**	17.67;38.64
4	24.00;17.17	24.00;24.80	24.00;32.00	19.56;33.26	16.00;33.30
5	25.00;17.17	25.00;24.80	19.47;27.56	15.00;28.82	11.36;28.94
6	24.00;17.17	17.33;21.47	12.00;24.00	7.64;25.18	4.00;25.30

The following figures illustrate the payoffs of player 1 and 2 based on the previous results.

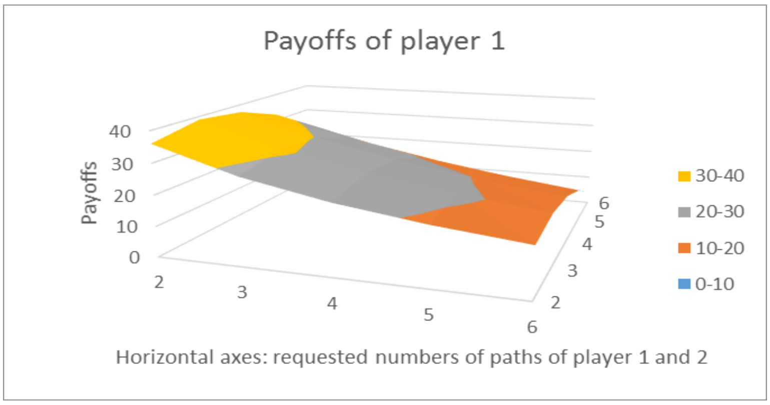


Figure 2

Monetary payoffs of player 1

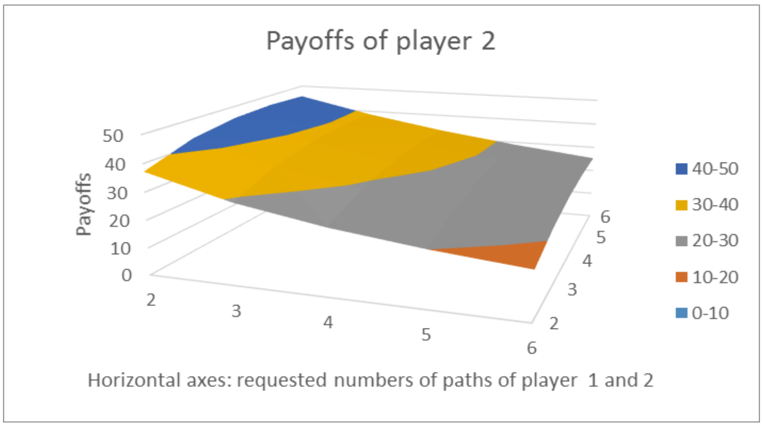


Figure 3

Monetary payoffs of player 2

It is visible from the figures that the discussed game belongs to the family of concave games. This is because, if we fix the number of requested paths for a player, then the payoff function of the other player is concave in its own variable. The only Nash equilibrium lies at the profile of requested paths where *both* players obtain maximum possible payoffs provided that the other player's choice is fixed. The importance of Theorem 3.7, lies in the unique nature of the Nash equilibrium profile.

As it can also be seen, the server can calculate the Nash equilibrium point based on the throughput requests the players sent to it, and thus, the server is also involved in the process, as declared in the introduction.

In the following example, we will change the payoff functions and use throughput increases and decreases representing the utility and cost values.

4.2 Throughput Payoffs

In the previous example the server calculated the number of paths to be given to the players based on the throughput needs and monetary payoff calculation. Based on the assumptions we argued that multipath communication system path allocation strategy for finite resources can be modeled by game theory and the path distribution is calculated according to the players' benefits. In the following we create a game between players sharing common limited resources and requests for paths in order to maximize their own throughput capacity, but their overall increases are limited by the hardware capacity they have. The utility is represented by the gained throughput and the cost function is represented by the loss of the throughput, for an MPC client. We use the observation described by Lencse and Kovacs in [9] that the throughput aggregation of a client that uses multipath communication system is limited by the hardware capacity. In this game, the cost function representing this limitation appeared after a certain number of paths. The payoff function increases quasi-linearly according in the received path number, and remains quasi-constant after it reaches its hardware limitation.

A suitable function that represents what we defined in the previous paragraph is defined below as the cost function of the client.

$$\pi_1(k_1,k_2)=0.01(p(k_1)U_p-\ln(1+e^{p(k_1)-PMax_1})) \tag{14}$$

$$\pi_2(k_1,k_2)=0.01(p(k_2)U_p-\ln(1+e^{p(k_2)-PMax_2})) \tag{15}$$

Here, k_1,k_2 represent the number of requested paths of players 1 and 2, respectively. U_p is the total throughput capacity of the server, which is used by the players. The 0.01 multiplier is used only to avoid large function values. The $p(k_i)$ in the utility part of the function (14) and (15) denotes the possessed number of paths the ith player acquired from the server. If the sum of the number of requested paths ($k_1 + k_2$) does not exceed the total number of available path (K), then $p(k_i) = k_i$. If the sum of the requested paths exceeds those available, K, the server must distribute the paths in proportion to the requests originated from the players (we refer the reader to the rationing (allocation) rule defined in Section 3). In that case, $p(k_i) = \lfloor k_i K / (k_1 + k_2) \rfloor$. The cost function is the throughput loss for the *i*th player that happens above a certain number of allocated paths, caused by the hardware limitation of the client. The maximal number of paths the player can handle is determined by $PMax_i$. The cost function $\ln(1 + e^{p(k_i) - PMax_i})$ is a softplus sigmoid function. This function increases quasi-linearly until a certain point ($PMax_i$) and then remains quasi-constant. Its usability and the property of being easily differentiable makes this function applicable to represent the hardware limitation that occurs at a certain point, where the MPC client acquires a number of paths that exceeds its limitation.

It is easy to see that the game is concave, as the utility part of each payoff function is concave, while the cost part is convex. Therefore, we can apply Theorem 3.7 and state that this game has again one and only one Nash equilibrium point.

To determine the one and only Nash equilibrium profile of the game, we have to solve the following system of equations.

$$\frac{\partial \pi_1(k_1, k_2)}{\partial k_1} = \left(\left\lfloor \frac{2k_1 + k_2}{K^2} \right\rfloor U_p - \frac{1}{1 + e^{p(k_1) - PMax_1} p(k_1) dk_1} \right) 0.01 \tag{16}$$

$$\frac{\partial \pi_2(k_1, k_2)}{\partial k_2} = \left(\left\lfloor \frac{2k_1 + k_2}{K^2} \right\rfloor U_p - \frac{1}{1 + e^{p(k_2) - PMax_2} p(k_2) dk_2} \right) 0.01 \tag{17}$$

The payoff function of players remains quasi-constant after they acquired the number of paths given by $PMax_i$. That payoff function is applicable for modeling the throughput increases, according to the players path requests, from the server in a multipath communication system. The following illustrates the hardware resource changes, by changing the requested number of paths.

We fixed $PMax_1$ and $PMax_2$ to both players for 6 and 5 respectively. The throughputs of the players regarding their requests are presented in the following tables and figures. The Nash equilibrium profile is also indicated.

Table 2

Payoff function values according to the number of requested paths of players 1;2

k2 k1	2	3	4	5	6
2	19.51;19.51	19.51;28.73	19.51;36.86	19.51;43.06	19.51;46.86
3	29.51;19.51	29.51;28.73	29.51;36.86	29.51;43.06	19.81;43.06
4	38.73;19.51	38.73;28.73	38.73;36.86	29.51;36.86	29.51;36.86
5	46.86;19.51	46.86;28.73	36.86;28.73	38.73;36.86	29.51;36.86
6	53.06;19.51	46.86;28.73	46.86;28.73	38.73;28.73	**38.73;36.86**

The following figures illustrate the payoffs of player 1 and 2 based on the previous results.

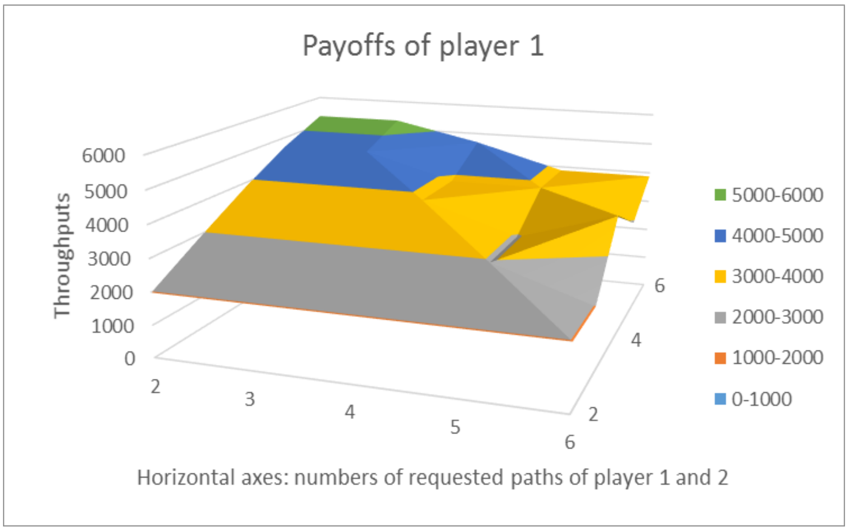


Figure 4

Throughput payoffs of player 1

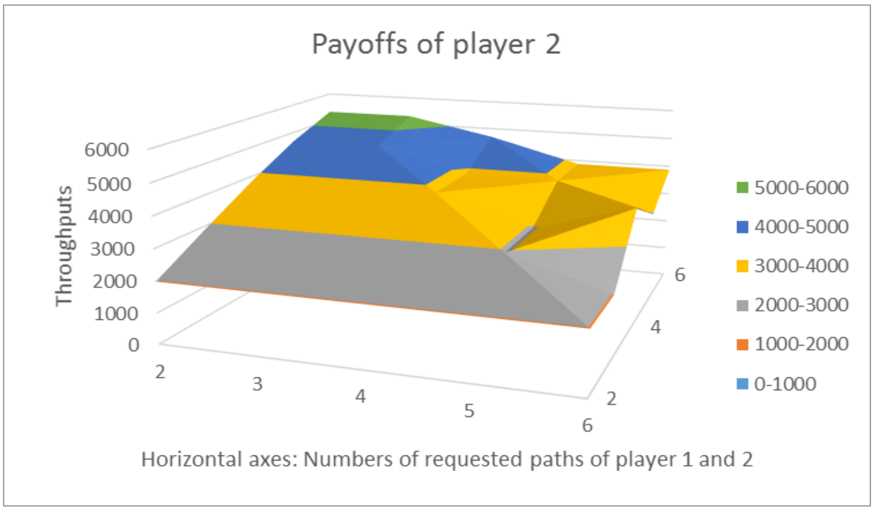


Figure 5
Throughput payoffs of player 2

We can draw the same conclusion as in Section 4.1. As the game is concave, we have only one Nash equilibrium profile of path requests. The payoff functions of both players are strictly concave, if the request of the other player is fixed.

The models can easily be extended to more than two players. As presented in Section 3, the construction of the equation system, leading to the unique Nash equilibrium profile for the game remains the same, for more than two endnodes.

Conclusions

These current systems usually have more than one networking interface: Wi-Fi, 3G/4G, Bluetooth, NFC etc. The idea of combining available interfaces in one communication session is a hot research area today. Multipath/multilink technology aims at aggregating speed capacities – i.e. throughputs – of the available paths/links. Different laboratory measurements show that the multipath technology, is able, to efficiently aggregate the throughput capacity of the paths.

Due to the interaction of endnodes and the server in a multipath/multilink environment, we found that a game theoretical approach could also be useful for modeling the overall system utilization. In this paper we introduced a non-cooperative, game-theoretical framework, for path allocation in a multipath network communication system to provide an analytical tool for finding the best available path allocation mechanism in the system. We applied a classical game theoretical approach for the multipath/multilink network communication system, in which clients aim at increasing their bandwidths by requesting new paths from the server.

After a precise mathematical definition of the multipath/multilink environment, we introduced our game-theoretical model. We stressed the use of the Nash equilibrium concept throughout the paper. We showed that the defined game belonged to the class of concave games, ensuring the existence of a single Nash equilibrium strategy profile in the game. Our results were proven to applicable for a wider range of payoff functions. Therefore, we had the possibility to present two different examples to solve a path allocation game. In the first, clients were assumed to maximize their monetary payoffs, which was defined as a linear combination of their utility of receiving a certain number of paths (expressed in money), and their financial cost for their path requests. The second approach considered the limited hardware capacity of the clients. In the second example, the payoff was not expressed in money, but in throughput which can be used also to investigate modeling of Big Data centers' performance, using multipath internal communication infrastructure (see [2]).

For both games a rationing (allocation) rule was defined. We illustrated the payoff vectors of the two endnodes for different path request combinations, and besides, we gave explanation for the single (unique) Nash equilibrium. The two example models employed two different approaches, to model client behavior, and we did not wish to show a preference to either side. These can be considered as suggestions, to understand and solve clients' decision problem, in a multipath/multilink environment. Our models can be considered as clear, positive results for client side payoff maximization. A possible further research direction includes investigating the server side's decision problems. Another direction might include a more exact specification (estimation) of clients' payoff functions, based on lab experiments.

To conclude, this paper has contributed to multipath/multilink network research, by offering a link between game theory and multipath/multilink systems, and has introduced a model that is suitable for further studies.

Acknowledgement

The work was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project has been supported by the European Union, co-financed by the European Social Fund.

References

[1] M. L. M. Kiah, L. K. Qabajeh, M. M. Qabajeh, "Unicast Position-based Routing Protocols for Ad-Hoc Networks", Acta Polytechnica Hungarica, Vol. 7. No. 5, pp. 19-46, 2010

[2] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, M. Handley, "Improving Datacenter Performance and Robustness with Multipath TCP", Proceedings of the ACM SIGCOMM 2011 conference, pp. 266-277, ACM New York, NY, USA, 2011

[3] Y. Cao, M. Xu, X. Fu, E. Dong, “Explicit Multipath Congestion Control for Data Center Networks”, Proceedings of the CoNEXT’13, pp. 73-84, 2013, Santa Barbara, California, USA, 2013

[4] IEEE Standards Association, "1905.1-2013 - IEEE Standard for a Convergent Digital Home Network for Heterogeneous Technologies", 2013. Available: http://standards.ieee.org/findstds/standard/1905.1a-2014.html (Downloaded: 12/01/2015)

[5] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, “TCP Extensions for Multipath Operation with Multiple Addresses”; IETF RFC-6824, 2013. Available: http://tools.ietf.org/html/rfc6824 (Downloaded 20/01/2015)

[6] C. Paasch, G. Detal, S. Barré, F. Duchêne, O. Bonaventure: “The Fastest TCP Connection with Multipath TCP”; ICTEAM, UCLouvain, Louvain-la-Neuve, Belgium; http://multipath-tcp.org/pmwiki.php?n=Main.50Gbps (Downloaded: 04/03/2015.)

[7] B. Almási, Sz. Szilágyi: "Throughput Performance Analysis of the Multipath Communication Library MPT", Proceedings of the 36th International Conference on Telecommunications and Signal Processing (TSP 2013, ISBN:978-1-4799-0403-7), pp. 86-90, Rome, Italy, 2013

[8] B. Almási, Sz. Szilágyi: “Investigating the Throughput Performance of the MPT Multipath Communication Library in IPv4 and IPv6”, Journal of Applied Research and Technology, To appear

[9] G. Lencse, Á. Kovács, "Advanced Measurements of the Aggregation Capability of the MPT Multipath Communication Library", International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems, Vol. 4. No. 2, pp. 41-48, 2015

[10] G. Lencse, Á. Kovács, "Testing the Channel Aggregation Capability of the MPT Multipath Communication Library", World Symposium on Computer Networks and Information Security 2014 (WSCNIS 2014), Hammamet, Tunisia, 13-15 June, 2014, ISBN: 978-9938-9511-9-6, Paper ID: 1569946547

[11] Khalili, Ramin, et al. "MPTCP is not Pareto-Optimal: Performance Issues and a Possible Solution." Proceedings of the 8th international conference on Emerging networking experiments and technologies. ACM, 2012

[12] Morgenstern, Oskar, and John Von Neumann. "Theory of Games and Economic Behavior" (1953)

[13] Nash Jr, John F. "The Bargaining Problem." Econometrica: Journal of the Econometric Society (1950): 155-162

[14] Easley, David, and J. Kleinberg. "Modeling Network Traffic using Game Theory."Networks, Crowds, and Markets: Reasoning about a Highly Connected World (2010): 229-247

[15] Murchland, John D. "Braess's Paradox of Traffic Flow." Transportation Research4.4 (1970): 391-394

[16] Fudenberg, Tirole, J. "Game Theory" MIT Press, 1991

[17] J. B. Rosen. "Existence and Uniqueness of Equilibrium Points for Concave N-Person Games" Econometrica, Volume 33, Issue 3, 1965, pp. 520-534

Classification of Electroencephalograph Data: A Hubness-aware Approach

Krisztian Buza, Júlia Koller

BioIntelligence Lab, Institute of Genomic Medicine and Rare Disorders, Semmelweis University, Tömő u. 25-29, H-1083 Budapest, Hungary, buza@biointelligence.hu, jkoller@biointelligence.hu

Abstract: Classification of electroencephalograph (EEG) data is the common denominator in various recognition tasks related to EEG signals. Automated recognition systems are especially useful in cases when continuous, long-term EEG is recorded and the resulting data, due to its huge amount, cannot be analyzed by human experts in depth. EEG-related recognition tasks may support medical diagnosis and they are core components of EEG-controlled devices such as web browsers or spelling devices for paralyzed patients. State-of-the-art solutions are based on machine learning. In this paper, we show that EEG datasets contain hubs, i.e., signals that appear as nearest neighbors of surprisingly many signals. This paper is the first to document this observation for EEG datasets. Next, we argue that the presence of hubs has to be taken into account for the classification of EEG signals, therefore, we adapt hubness-aware classifiers to EEG data. Finally, we present the results of our empirical study on a large, publicly available collection of EEG signals and show that hubness-aware classifiers outperform the state-of-the-art time-series classifier.

Keywords: electroencephalograph; nearest neighbor; classification; hubs

1 Introduction

Ongoing large-scale brain research projects – such as the European Human Brain Project, the BRAIN initiative announced by President Obama[1] and the Hungarian National Brain Research Project – are expected to generate an unprecedented amount of data describing brain activity. This is likely to lead to an increased need for enhancement of statistical analysis techniques, development of new methods and computer software that support the analysis of such data.

One of the most wide-spread devices for monitoring and recording the electrical activity of the brain is the electroencephalograph (EEG). EEG is used in clinical practice, research and various other domains. Its numerous applications contain

[1] see also https://www.humanbrainproject.eu/ , http://braininitiative.nih.gov/

pre-surgical evaluation [1], diagnostic decision-making [2] and the assessment of chronic headaches [3]. EEG "is an important diagnostic tool for patients with seizures and other paroxysmal behavioral events" [4], it may provide diagnostic information in case of epilepsy [5], Alzheimer's disease [6], [7], schizophrenia [8] or after a brain injury [9]. EEG is used in various brain-computer interfaces [10] which are core components of EEG-controlled devices, such as spelling tools [11] or web browsers [12] for paralyzed patients. EEG was used to study sleepiness in long distance truck driving [13] and there were attempts to predict upcoming emergency braking based on EEG signals [14].

Continuous, long-term EEG monitoring is required in many cases, such as some forms of epilepsy [15], [16], coma, cerebral ischemia, assessment of a medication [17], sleep disorders and disorders of consciousness [18], psychiatric conditions, movement disorders [19], during anesthesia, in intensive care units and neonatal intensive care units [20], [17]. In these cases, EEG signals are recorded for hours or days. Due to the large amount of captured data, the detailed analysis of the entire records is usually not possible by human experts. Therefore, in order to allow for real-time diagnosis and thorough analysis of the data, various techniques were developed to assist medical doctors and other employees of hospitals and to allow for the (semi-)automated analysis of EEG signals.

A common feature of the aforementioned diagnostic problems and EEG-based tools (such as EEG-controlled web browsers or spelling tools) is that they involve recognition tasks related to EEG signals. As EEG signals can be considered as multivariate time-series, these recognition tasks can be formulated as multivariate time-series classification problems, for which state-of-the-art solutions are based on machine learning. For example, Boostani et al. used Boosted Direct Linear Discriminant Analysis for the diagnosis of schizophrenia [21], Sabeti et al. selected best channels based on mutual information and utilized genetic programming in order to select best features [22], while Srinivasan et al. used neural networks for EEG classification [23]. Sun et al. studied ensemble methods [24]. For an excellent survey about EEG-related analysis tasks we refer to [25].

Nearest-neighbor classifiers using dynamic time warping (DTW) as distance measure have been shown to be competitive, if not superior, to many state-of-the-art time-series classification methods such as neural networks or hidden Markov models, see, e.g. [26]. The experimental evidence is underlined by theoretical results about error bounds for the nearest neighbor classifiers. While classic works, such as [27], considered vector data, in their recent work, Chen et al. [28] focused on the nearest neighbor classification of time series and proved error bounds for nearest neighbor-like time-series classifiers. Besides their accuracy, nearest neighbor classifiers deliver human-understandable explanations for their classification decisions in the form of sets of similar instances which makes them preferable to medical applications. As nearest neighbor classifiers are attractive both from the theoretical and practical points of view, considerable research was performed to enhance nearest neighbor classification. Some of the most promising recent methods were based on the observation that a few time-series tend to be nearest neighbors of surprisingly large amount of other time-series [62]. We refer to this phenomenon as *the presence of hubs* or *hubness* for short, and the classifiers that take this phenomenon into account are called *hubness-aware classifiers*. Hubness-aware classifiers were originally proposed for vector data and image data [29], [30], [31], and only few works considered hubness-aware classification of time series [32], [33], [34], but none of them considered hubness-aware classifiers for EEG data.

In this paper, we focus on hubness-aware classification of EEG signals. As we will show, hubness-aware classifiers lead to statistically significant improvements over the state-of-the-art in terms of accuracy, precision, recall and F-score.

The paper is organized as follows. In Section 2 we introduce basic concepts and notations, while Section 3 is devoted to the presence of hubs in EEG data and hubness-aware classifiers. In Section 4 we present the results of our experiments. Finally, we conclude in Section 5.

2 Basic Concepts and Notations

We use $\boldsymbol{D}$ to denote the set of EEG signals used to construct the recognition model, called *classifier*. $\boldsymbol{D}$ is called *training data* and each signal in $\boldsymbol{D}$ is associated with a class label. For example, in the simplest case of diagnosing epilepsy, there are two classes of signals, one of them contains the EEG signals of healthy individuals, while the second class contains the EEG signals of epileptic patients. The class label of each signal denotes to which class that signal belongs, i.e., in the previous example, the class label of a particular signal denotes whether this signal originates from a healthy or epileptic individual. The class labels of the training data are known while constructing the classifier. The process of constructing the classifier is called *training*. Once the classifier is trained, it can be applied to new signals, i.e., the classifier can be used to predict the class labels of new signals. In order to evaluate our classifier we will use a second set of EEG signals $\boldsymbol{D}^{\text{test}}$, called *test data*. $\boldsymbol{D}^{\text{test}}$ is disjoint from $\boldsymbol{D}$ and the class labels of the signals in $\boldsymbol{D}^{\text{test}}$ are unknown to the classifier. We only use the class labels of the signals in $\boldsymbol{D}^{\text{test}}$ to quantitatively assess the performance of the classifier (by comparing the predicted and true class labels and calculating statistics regarding the performance).

3 Hubness-aware Classification of EEG Data

3.1 Hubs in EEG Data

The presence of hubs, i.e., the presence of a few instances (objects, signals) that occur surprisingly frequently as neighbors (peers) of other instances, while many instances (almost) never occur as neighbors, has been observed for various natural and artificial networks, such as protein-protein-interaction (PPI) networks or the internet [40], [41], [42], [43], [44]. Hubs were shown to be relevant in various contexts, including text mining [45], [46], music retrieval and recommendation [47], [48], [49], [50], image data [51], [52] and time series [34], [53].

In this study, we focus on EEG signals, and we will describe our novel observation that nearest neighbor graphs built from EEG signals contain hubs.

In context of EEG classification, informally, the hubness phenomenon means that some (relatively few) EEG signals appear as nearest neighbors of many EEG signals. Note that, throughout this paper, an EEG signal is *never* treated as the nearest neighbor of itself. Intuitively speaking, very frequent neighbors, or hubs, dominate the neighbor sets and therefore, in the context of similarity-based learning, they represent the centers of influence within the data. In contrast to hubs, there are signals that occur rarely as neighbors and therefore they contribute little to the analytic process. We will refer to them as *orphans* or *anti-hubs*.

In order to express hubness more precisely, for an EEG dataset ***D*** one can define the *k-occurrence* of a signal t from ***D***, denoted by $N_k(t)$, as the number of signals in ***D*** having t among their k-nearest neighbors:

$$N_k(t) = |\{t_i | t \in \mathcal{N}_k(t_i)\}| \tag{1}$$

where $\mathcal{N}_k(t_i)$ denotes the set of k-nearest neighbors of t_i. With the term *hubness* we refer to the phenomenon that the distribution of $N_k(t)$ becomes significantly skewed to the right. We can measure this skewness with the third standardized moment of $N_k(t)$:

$$\mathcal{S}_{N_k(t)} = \frac{E[(N_k(t) - \mu_{N_k(t)})^3]}{\sigma^3_{N_k(t)}} \tag{2}$$

where $\mu_{N_k(t)}$ and $\sigma_{N_k(t)}$ are the mean and standard deviation of the distribution of $N_k(t)$ and the notation E stands for the expected value of the quantity between the brackets. When the skewness is higher than zero, the corresponding distribution is skewed to the right and starts presenting a long tail. It should be noted, though, that the occurrence distribution skewness is only one of indicator statistics and that the distributions with same or similar skewness can still take different shapes.

In the presence of class labels, we distinguish between *good hubness* and *bad hubness*: we say that an EEG signal *t'* is a *good k-nearest neighbor* of the signal *t*, if (i) *t'* is one of the *k*-nearest neighbors of *t*, and (ii) both have the *same* class labels. Similarly: we say that the signal *t'* is a *bad k-nearest neighbor* of the signal *t*, if (i) *t'* is one of the *k*-nearest neighbors of *t*, and (ii) they have *different* class labels. This allows us to define *good (bad) k-occurrence* of a signal *t*, $GN_k(t)$ (and $BN_k(t)$ respectively), which is the number of other signals that have *t* as one of their good (bad respectively) *k*-nearest neighbors. For EEG signals, both distributions of $GN_k(t)$ and $BN_k(t)$ are skewed, as it is exemplified in Fig. 1, which depicts the distributions of $GN_1(t)$, $BN_1(t)$ and $N_1(t)$ for a publicly available EEG dataset from the UCI Machine Learning repository. We describe this dataset in more detail in Section 4. As shown, the distributions have long tails.

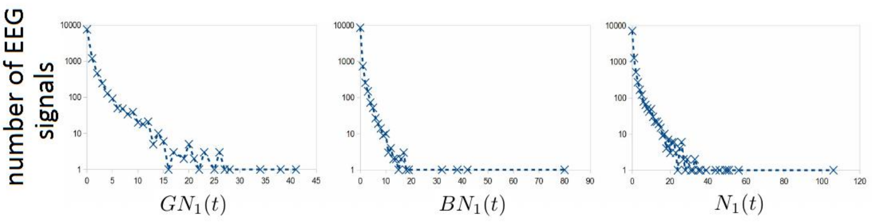


Figure 1
Distribution of $GN_1(t)$, $BN_1(t)$ and $N_1(t)$ for the EEG dataset from the UCI Machine Learning repository. Note that the scale is logarithmic on the vertical axis.

We say that a signal *t* is a good (or bad) hub, if $GN_k(t)$ (or $BN_k(t)$ respectively) is exceptionally large for *t*. For the nearest neighbor classification of time series, such as EEG signals, the skewness of good occurrence is of major importance, because some few time series are responsible for large portion of the overall error: *bad hubs* tend to *misclassify* a surprisingly large number of time series [34]. Therefore, one has to take into account the presence of good and bad hubs in EEG datasets.

In the light of the previous discussion, the total occurrence count $N_k(t)$ of an EEG signal *t* can be decomposed into good and bad occurrence counts: $N_k(t) = GN_k(t) + BN_k(t)$. More generally, we can decompose the total occurrence count into the class-conditional counts:

$$N_k(t) = \sum_{C \in \mathcal{C}} N_{k,C}(t), \tag{3}$$

where $\mathcal{C}$ denotes the set of all the classes and $N_{k,C}(t)$ denotes how many times *t* occurs as one of the *k*-nearest neighbors of signals belonging to class *C*, i.e.,

$$N_{k,C}(t) = |\{t_i | t \in \mathcal{N}_k(t_i) \ \wedge \ y_i = C\}|, \tag{4}$$

where y_i denotes the class label of t_i.

3.2 Hubness-aware Classifiers

In this section, we give a detailed description of classifiers that work *under the assumption of hubness*. In our experiments in Section 4, we will examine how these algorithms perform on EEG signals. The algorithms are general, in the sort of sense that they can be applied to any kind of data, provided that an appropriate distance measure between the instances of the dataset is available. In case of EEG-data, we use multivariate DTW as distance measure as described in [36]. As in our case instances are EEG-signals, we will mostly use the term *EEG-signal* instead of *instance* while describing hubness-aware classifiers.

In order to predict how hubs will affect classification of non-labeled signals (e.g. signals arising from observations in the future), we can model the influence of hubs by considering the training data. The training data can be utilized to learn a neighbor occurrence model that can be used to estimate the probability of individual neighbor occurrences for each class. There are many ways to exploit the information contained in the occurrence models. Next, we will review the most prominent approaches. While describing these approaches, we will consider the case of classifying the signal t^*. We will denote its unknown class label as y^* and its nearest neighbors as t_i, where i is an integer number in the range from 1 to k. We assume that the test data is not available when building the model, and therefore $N_k(t)$, $N_{k,C}(t)$, $GN_k(t)$ and $BN_k(t)$ are calculated on the training data.

3.2.1 hw-*k*NN: Hubness-aware Weighting

The weighting scheme proposed by Radovanović et al. [54] is one of the simplest ways to reduce the influence of bad hubs. In this approach, lower voting weights are assigned to bad hubs in the nearest neighbor classifier. In hw-*k*NN, the vote of each neighbor t_i is weighted by $e^{-h_b(t_i)}$, where

$$h_b(t_i) = \frac{BN_k(t_i) - \mu_{BN_k(t)}}{\sigma_{BN_k(t)}} \tag{5}$$

is the standardized bad hubness score of the neighbor signal t_i in $\mathcal{N}_k(t^*)$, while $\mu_{BN_k(t)}$ and $\sigma_{BN_k(t)}$ are the mean and standard deviation of $BN_k(t)$.

In hw-*k*NN all neighbors vote by their own label. As this may be disadvantageous in some cases [51], in the algorithms considered below, the neighbors do not always vote by their own labels, which is a major difference to hw-*k*NN.

3.2.2 h-FNN: Hubness-based Fuzzy Nearest Neighbors

Consider the relative class hubness $u_C(t_i)$ of each nearest neighbor t_i:

$$u_C(t_i) = \frac{N_{k,C}(t_i)}{N_k(t_i)} \tag{6}$$

where C denotes one of the classes. The above $u_C(t_i)$ can be interpreted as the fuzziness of the event that t_i occurred as one of the neighbors. Integrating fuzziness as a measure of uncertainty is usual in k-nearest neighbor methods and h-FNN [30] uses the relative class hubness when assigning class-conditional vote weights. The approach is based on the fuzzy k-nearest neighbor voting framework [55]. Therefore, the probability of each class C for the signal t^* is estimated as:

$$u_C(t^*) = \frac{\sum_{t_i \in \mathcal{N}_k(t^*)} u_C(t_i)}{\sum_{t_i \in \mathcal{N}_k(t^*)} \sum_{C' \in \mathcal{C}} u_{C'}(t_i)}. \quad (7)$$

Special care has to be devoted to anti-hubs. Their occurrence fuzziness is estimated as the average fuzziness of points from the same class. Optional distance-based vote weighting is possible.

3.2.3 NHBNN: Naive Hubness Bayesian k-Nearest Neighbor

For each class C, Naive Hubness Bayesian k-Nearest Neighbor (NHBNN) estimates $P(y^* = C \mid \mathcal{N}_k(t^*))$, i.e., the probability that t^* belongs to class C given its nearest neighbors. Then, NHBNN selects the class with highest probability.

NHBNN follows a Bayesian approach to assess $P(y^* = C \mid \mathcal{N}_k(t^*))$. For each training EEG signal t of the training dataset, one can estimate the probability of the event that t appears as one of the k-nearest neighbors of any training instance belonging to class C. This probability is denoted by $P(t \in \mathcal{N}_k | C)$.

Assuming conditional independence between the nearest neighbors given the class, $P(y^* = C \mid \mathcal{N}_k(t^*))$ can be assessed as follows:

$$P(y^* = C | \mathcal{N}_k(t^*)) \quad \propto \quad P(C) \prod_{t_i \in \mathcal{N}_k(t^*)} P(t_i \in \mathcal{N}_k | C) \quad (8)$$

where $P(C)$ denotes the prior probability of the event that an instance belongs to class C. From the labeled training data, $P(C)$ can be estimated as $|\boldsymbol{D}_C|/|\boldsymbol{D}|$, where $|\boldsymbol{D}_C|$ denotes the number of EEG signals instances belonging to class C in the training data, and $|\boldsymbol{D}|$ is the total number of EEG signals in the training data. The maximum likelihood estimate of $P(t_i \in \mathcal{N}_k | C)$ is the fraction

$$P(t_i \in \mathcal{N}_k | C) \approx \frac{N_{k,C}(t_i)}{|\boldsymbol{D}_C|}. \quad (9)$$

Estimating $P(t_i \in \mathcal{N}_k | C)$ according to Eq. (9) may simply lead to zero probabilities. In order to avoid it, we can use a simple Laplace-estimate for $P(t_i \in \mathcal{N}_k | C)$ as follows:

$$P(t_i \in \mathcal{N}_k | C) \approx \frac{N_{k,C}(t_i) + m}{|\boldsymbol{D}_C| + mq} \quad (10)$$

where $m > 0$ and q denotes the number of classes. Informally, this estimate can be interpreted as follows: we consider m additional pseudo-instances from each class and we assume that t_i appears as one of the k-nearest neighbors of the pseudo-instances from class C. We use $m=1$ in our experiments.

Even though k-occurrences are highly correlated, as shown in [61], NHBNN offers improvement over the basic k-NN. This is in accordance with other results from the literature that state that Naive Bayes can deliver good results even in cases with high independence assumption violation [56].

3.2.4 HIKNN: Hubness Information *k*-Nearest Neighbor

In h-FNN, as in most *k*NN classifiers, all neighbors are treated as equally important. The difference is sometimes made by introducing the dependency on the distance to t^*, the signal to be classified. However, it is also possible to deduce some sort of global neighbor relevance, based on the occurrence model, which is the basic idea behind HIKNN [29]. It embodies an information-theoretic interpretation of the neighbor occurrence events. In that context, rare occurrences have higher self-information, see Equation (11). The more frequently an EEG signal t occurs as nearest neighbor of other EEG signals, the less surprising is the occurrence of t as one of the nearest neighbors while classifying a new signal.

The EEG signals that rarely occur as neighbors are, therefore, more informative and they are favored by HIKNN. The reasons for this lies hidden in the geometry of high-dimensional feature spaces. Namely, hubs have been shown to lie closer to the cluster centers [57], as most high-dimensional data lies approximately on hyper-spheres. Therefore, hubs are points that are somewhat less 'local'. Therefore, favoring the rarely occurring points helps in consolidating the neighbor set locality. The algorithm itself is a bit more complex, as it not only reduces the vote weights based on the occurrence frequencies, but also modifies the fuzzy vote itself so that the rarely occurring points vote mostly by their labels and the hub points vote mostly by their occurrence profiles. Next, we will present the approach in more detail.

The self-information I_{t_i} associated with the event that t_i occurs as one of the nearest neighbors of an EEG signal to be classified can be calculated as

$$I_{t_i} = \log \frac{1}{P(t_i \in \mathcal{N}_k)} \quad , \quad P(t_i \in \mathcal{N}_k) \approx \frac{N_k(t_i)}{|\mathcal{D}^{train}|}. \tag{11}$$

Occurrence self-information is used to define the relative and absolute relevance factors in the following way:

$$\alpha(t_i) = \frac{I_{t_i} - \min_{t_j \in \mathcal{N}_k(t_i)} I_{t_j}}{\log |\mathcal{D}^{train}| - \min_{t_j \in \mathcal{N}_k(t_i)} I_{t_j}}, \quad \beta(t_i) = \frac{I_{t_i}}{\log |\mathcal{D}^{train}|}. \tag{12}$$

The final fuzzy vote of a neighbor t_i combines the information contained in its label with the information contained in its occurrence profile. The relative relevance factor is used for weighting the two information sources. This is shown in Eq. (13).

$$P_k(y^* = C|t_i) \approx \begin{cases} \alpha(t_i) + (1-\alpha(t_i)) \cdot u_C(t_i), & y_i = C \\ (1-\alpha(t_i)) \cdot u_C(t_i), & y_i \neq C \end{cases}. \tag{13}$$

Hubness-aware classifiers are illustrated by an elaborated example in [61].

3.2.5 On the Computational Aspects of the Implementation of Hubness-aware Classifiers

When classifying EEG signals, i.e., multivariate time series, with hubness-aware classifiers, the computationally most expensive step is the computation of the nearest neighbors of training instances, which is used to determine hubness-scores such as $N_k(t)$, $N_{k,C}(t)$, $GN_k(t)$ and $BN_k(t)$. On the one hand, approaches known to speed-up nearest neighbor classification of time series can be used to reduce the computational costs of hubness-aware classifiers. Such techniques include: speeding-up the calculation of the distance of two time series (by, e.g. limiting the warping window size), indexing and reducing the length of the time series used. For more details we refer to [32] and the references therein. On the other hand, we note that distances between different pairs of training instances can be calculated independently, therefore, computations can be parallelized and implemented on a distributed supercomputer (cloud).

4 Experimental Evaluation

In this section, first, we describe the data we used in our experiments. Next, we provide details of the experimental settings. Subsequently, we present our experimental results.

4.1 Data

In order to evaluate our approach, we used the publicly available EEG dataset[2] from the UCI machine learning repository. This collection contains in total 11028 EEG signals recorded from 122 people. Out of the 122 people, 77 were alcoholic patients and 45 were healthy individuals. Each signal was recorded using 64 electrodes at 256 Hz for 1 second. Therefore, each EEG signal is a 64 dimensional

[2] http://archive.ics.uci.edu/ml/datasets/EEG+Database

time series of length 256 in this collection. In order to filter noise, as a simple preprocessing step, we reduced the length of the signals from 256 to 64 by binning with a window size of 4, i.e., we averaged consecutive values of the signal in non-overlapping windows of length 4.

As noted before, the examined EEG dataset exhibits remarkable *hubness*, as the neighbor occurrence frequency is significantly skewed. For instance, if we set $k=1$, there exists a hub signal that acts as a nearest neighbor of 113 other signals from the data. For $k = 10$, the top neighbor occurrence frequency peaks at 707. This illustrates the significance of hub signals in practice. They influence many classification decisions.

As shown in Figure 2, the distribution of such hub signals, as well as detrimental (bad) hubs and anti-hubs differ between the two classes of the EEG dataset and do not follow the prior class distribution. In particular, most hub signals emerge among the Alcoholic class, while most anti-hubs appear among the signals from the Healthy class. Many anti-hubs are in fact known to be outliers and points that lie in borderline regions, far away from local cluster means – and they are, therefore, more difficult to handle and properly associate with a particular class in a prospective study. This suggests that the two classes might not be equally difficult for *k*-NN classification.

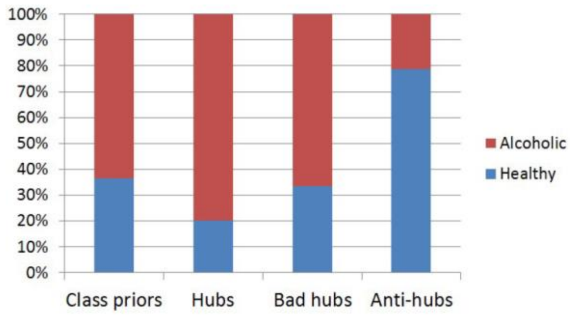


Figure 2
Distribution of hub signals, bad hubs and anti-hubs in the Healthy and Alcoholic class

4.2 Experimental Settings

In our experiments we examined the performance of hubness-aware classifiers. We compared these algorithms to *k*-NN. Both in case of *k*-NN and the hubness-aware classifiers, we used multivariate DTW as distance measure as described in [36]. We set $k = 10$ for the hubness-aware classifiers. This value was chosen, since most hubness-aware methods are known to perform better in cases when k is somewhat larger than 1, because more reliable neighbor occurrence models can be inferred from more occurrence information, see also [29]. In case of the baseline, *k*-NN, we experimented with both $k = 1$ and $k = 10$.

Based on the EEG signals, we aimed to recognize whether a person is affected by alcoholism or not. In other words: the class label of an EEG-signal reflects whether this signal originates from an alcoholic patient or a healthy individual. Both for hubness-aware classifiers and the baseline, we make use of the information that we know which signals originate from the same person: we classify a person as healthy (or alcoholic, respectively) if majority of the signals originating from that person were classified as healthy (alcoholic, respectively).

In all the experiments, we used the *10x10-fold crossvalidation* protocol to evaluate hubness aware classifiers and the baseline. With 10-fold crossvalidation we mean that we partition the entire dataset into 10 disjoint random splits and we use 9 out of these splits as train data, while the remaining split is used as test data. We repeat the experiment 10 times, in each round we use a different split as test data. With 10x10-fold crossvalidation we mean that we repeat the above 10-fold crossvalidation procedure 10 times, each time beginning with a different random partitioning of the data. While partitioning the data, we pay attention that all the signals belonging to the same person are assigned to the same split, and therefore each person either appears in the training data or in the test data, but not in both. On the one hand, this allows to simulate the real-world scenario in which the recognition system is applied to new users; on the other hand, EEG signals are somewhat characteristic to individuals, see e.g. person identification systems using EEG [58], therefore, if the same person would appear in both the train and test data, this could lead to overoptimistic results.

4.3 Performance Metrics

As primary performance measure we used accuracy, i.e., the number of correctly classified persons divided by the number of all the persons in the dataset. We performed t-test at significance level of 0.05 in order to decide whether the differences are statistically significant.

Additionally, we measured precision, recall and F-score for the class of alcoholic patients. Precision and recall regarding class C are defined as $Prec(C) = TP(C) / (TP(C) + FP(C))$ and $Recall(C) = TP(C) / (TP(C) + FN(C))$ respectively, where $TP(C)$ denotes the true positive signals, i.e., signals that are classified as belonging to class C and they really belong to this class; $FP(C)$ denotes false positive signals, i.e., signals that are classified as belonging to class C, but they belong to some other class in reality; and $FN(C)$ denotes false negative, i.e., signals that are *not* classified as belonging to class C, but they belong to class C in reality. F-score is the harmonic mean of precision and recall: $F(C) = 2\ Prec(C)\ Recall(C) / (Prec(C) + Recall(C))$.

4.4 Results

The results of our experiments are summarized in Tab. 1 and Tab. 2. Tab. 1 shows accuracy of the examined methods averaged over 10x10 folds, while Tab. 2 shows precision, recall and F-score for the identification of alcoholic patients. This experiment simulates the medically relevant application scenario in which EEG is used to diagnose a disease. In both tables, we provide standard deviations after the ± sign. Additionally, in the last two columns of Tab. 1 we provide the results of statistical significance tests (t-test at significance level of 0.05) in the form of a symbol ± where + denotes significance, and – its absence when comparing to 1-NN and 10-NN respectively. In both tables, we underlined those hubness-aware classifiers that outperformed both baselines (in terms of accuracy and F-score).

Table 1

Accuracy ± standard deviation of hubness-aware classifiers and the baselines

Method	Accuracy	Significant difference compared to	
		1-NN	10-NN
1-NN	0.650 ± 0.055		
10-NN	0.662 ± 0.053		
<u>h-FNN</u>	0.690 ± 0.060	+	+
<u>NHBNN</u>	0.780 ± 0.112	+	+
<u>HIKNN</u>	0.663 ± 0.050	+	–
hw-*k*NN	0.660 ± 0.050	+	–

Table 2

Precision, recall and F-score ± its standard for the class of alcoholic patients

Method	Precision	Recall	F-score
1-NN	0.65 ± 0.04	0.99 ± 0.04	0.78 ± 0.03
10-NN	0.65 ± 0.04	1.00 ± 0.00	0.79 ± 0.03
<u>h-FNN</u>	0.67 ± 0.05	1.00 ± 0.00	0.80 ± 0.03
<u>NHBNN</u>	0.81 ± 0.10	0.87 ± 0.11	0.83 ± 0.09
HIKNN	0.65 ± 0.04	1.00 ± 0.00	0.79 ± 0.03
hw-*k*NN	0.65 ± 0.04	1.00 ± 0.00	0.79 ± 0.03

4.5 Discussion

Hubness-aware classifiers yield significant overall improvements over *k*-NN. However, some hubness-aware methods perform better than others.

In particular, all of the hubness-aware classifiers significantly outperformed 1-NN in terms of classification accuracy, whereas two hubness-aware classifiers, namely h-FNN and NHBNN, outperformed 10-NN significantly. Although in terms of accuracy, HIKNN appears to have outperformed 10-NN on average, the difference is not significant statistically. The simple weighting approach (hw-*k*NN) did not outperform 10-NN of the examined task of EEG signal classification. The highest improvements in accuracy were achieved by NHBNN, which seems to be very promising for this task.

In medical applications, as we have to deal with class-imbalanced data in many cases, precision and recall are often more important than accuracy. Therefore, in order to further assess the performance of hubness-aware classifiers, we measured their precision, recall and F-score on the class of alcoholic patients. We observed similar trends as in case of accuracy: the performance of the simple hw-kNN was comparable to the baselines, while NHBNN, h-FNN and HIKNN showed clear advantages. Again, NHBNN showed the best overall performance: NHBNN achieved the highest F-score as the relatively low recall of NHBNN was compensated by precision.

In order to interpret these improvements, we have analyzed how different signal types were handled by the tested classifiers. According to [59], we distinguish between four different types of signals: *safe* signals, that lie in class interiors and have all or most of their neighbors belong to the same class, *borderline* signals, that lie in borderline regions between different classes, *rare* signals that are somewhat unusual and distant from the class prototypes and *outliers*. Apart from safe signals, all other signal types are difficult to properly classify.

Figure 3 shows that the two classes in this EEG dataset are formed of different signal type distributions. Most signals of healthy individuals seem to be either borderline, rare or outliers. On the other hand, most signals of alcoholic patients seem to be safe in terms of *k*-NN classification. This indicates that there is probably a common pattern to most alcoholic EEG signals, while the healthy group might be less coherent and comprise different subgroups.

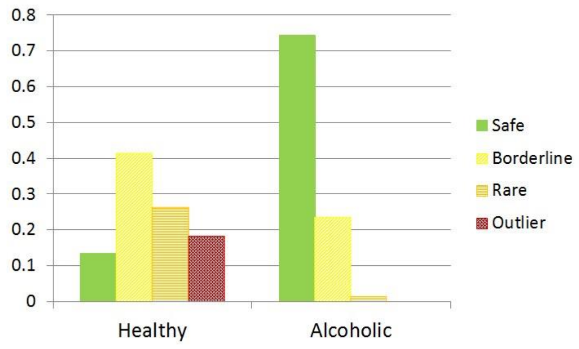


Figure 3

Distribution of different signal types in the Healthy and Alcoholic classes. The two classes have different signal type distributions: compared to the Healthy class, the Alcoholic class seems to be composed of more compact clusters, where most signals lie in class/cluster interiors.

The examined hubness-aware classifiers that improve over 10-NN achieve their improvement by increasing precision of classification for difficult signal types, i.e., borderline, rare and outlier signals, see Fig. 4. This is in concordance with prior observations in other class-imbalanced classification studies [60].

Finally, in order to clarify why hubness-aware classifiers might be well suited for EEG signal classification, we briefly discuss the merits of using neighbor occurrence models on this EEG dataset. Namely, unlike the baseline *k*-NN, hubness-aware classifiers are based on building neighbor occurrence models that learn from prior occurrences on the training set. Predicting the occurrence profiles of individual points requires us to consider reverse neighbor sets, in contrast to the direct *k*-NN sets in the *k*-NN baseline. With reverse neighbors of a signal *x*, we mean the set of signals that have *x* as one of their *k*-nearest neighbors. As Fig. 5 suggests, the average entropy of the reverse neighbor sets in the EEG dataset is lower that the entropy of the direct *k*-NN sets. This means that less uncertainty is present on average in the reverse neighbor sets.

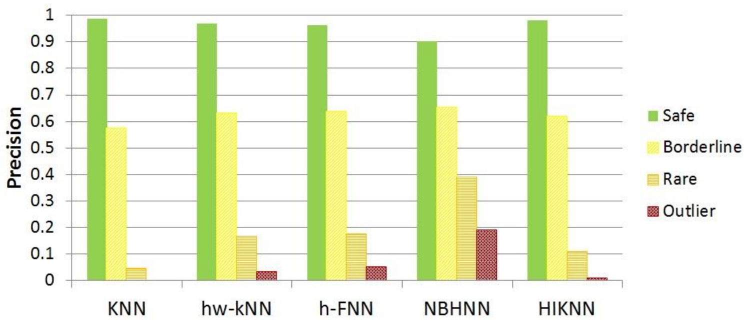


Figure 4

Precision of hubness-aware classifiers and *k*-NN on different signal types. Performance decomposition indicates clear improvements in case of the difficult signal types (borderline and rare signals, outliers).

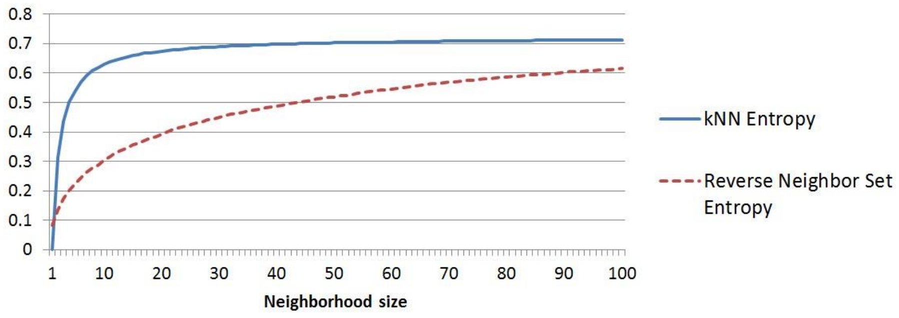


Figure 5

Average entropy (vertical axis) of *k*-nearest neighbor sets and reverse *k*-neighbor sets for various neighborhood sizes (horizontal axis). The lower uncertainty of reverse neighbor sets may explain why hubness-aware classifiers outperform *k*-NN.

Conclusions and Outlook

Classification is a common denominator across biomedical recognition tasks. We examined the effectiveness of hubness-aware classifiers in case of EEG signals.

Hubness-aware classification methods have recently been proposed for classifying complex and intrinsically high-dimensional datasets, under the assumption of *hubness*, which is the skewness of the neighbor occurrence distribution and characterizes many high-dimensional datasets. We have demonstrated that EEG data indeed exhibits significant hubness and that some recently proposed hubness-aware classification methods can be successfully used for signal class recognition.

These recent advances had not been applied to EEG data before and this study attempts to evaluate their usefulness in this context, as well as familiarize domain experts with the potential that these methods seem to hold for these data types.

We have experimentally compared several recently proposed hubness-aware classifiers on a large, publicly available EEG dataset. Our experiments demonstrate significant improvements over the baseline. Naive Hubness-Bayesian *k*-Neareset Neighbor classifier (NHBNN) showed very promising performance. As future work, we will consider different possibilities for boosting hubness-aware methods or combining them into classification ensembles.

Acknowledgement

Discussions with Dr. Nenad Tomašev, researcher of the Artificial Intelligence Laboratory, Jožef Stefan Institute, Ljubljana, Slovenia as well as his contributions to the paper, esp. in Section 4.5 are greatly appreciated. This research was performed within the framework of the grant of the Hungarian Scientific Research Fund - OTKA 111710 PD. This paper was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences. We thank Henri Begleiter at the Neurodynamics Laboratory at the State University of New York Health Center at Brooklyn for making the EEG data publicly available.

References

[1] S. Knake, E. Halgren, H. Shiraishi, K. Hara, H. Hamer, P. Grant, V. Carr, D. Foxe, S. Camposano, E. Busa, T. Witzel, M. Hmlinen, S. Ahlfors, E. Bromfield, P. Black, B. Bourgeois, A. Cole, G. Cosgrove, B. Dworetzky, J. Madsen, P. Larsson, D. Schomer, E. Thiele, A. Dale, B. Rosen, S. Stufflebeam, The Value of Multichannel Meg and Eeg in the Presurgical Evaluation of 70 Epilepsy Patients, Epilepsy Research 69 (2006) pp. 80-86

[2] J. Askamp, M. J. van Putten, Diagnostic Decision-Making after a First and Recurrent Seizure in Adults, Seizure 22 (2013) pp. 507-511

[3] U. Kramer, Y. Nevo, M. Y. Neufeld, S. Harel, The Value of EEG in Children with Chronic Headaches, Brain and Development 16 (1994) pp. 304-308

[4] J. Alving, S. Beniczky, Diagnostic Usefulness and Duration of the Inpatient Long-Term Video-EEG Monitoring: Findings in Patients Extensively Investigated before the Monitoring, Seizure 18 (2009) pp. 470-473

[5] E. Montalenti, D. Imperiale, A. Rovera, B. Bergamasco, P. Benna, Clinical Features, EEG Findings and Diagnostic Pitfalls in Juvenile Myoclonic Epilepsy: a Series of 63 Patients, Journal of the Neurological Sciences 184 (2001) pp. 65-70

[6] K. Bennys, G. Rondouin, C. Vergnes, J. Touchon, Diagnostic Value of Quantitative EEG in Alzheimers Disease, Neurophysiologie Clinique/Clinical Neurophysiology 31 (2001) pp. 153-160

[7] J. Dauwels, F. Vialatte, T. Musha, A. Cichocki, A Comparative Study of Synchrony Measures for the Early Diagnosis of Alzheimer's Disease Based on Eeg, NeuroImage 49 (2010) pp. 668-693

[8] M. Sabeti, S. Katebi, R. Boostani, Entropy and Complexity Measures for EEG Signal Classification of Schizophrenic and Control Participants, Artificial Intelligence in Medicine 47 (2009) pp. 263-274

[9] Large-Scale Brain Dynamics in Disorders of Consciousness, Current Opinion in Neurobiology 25 (2014) pp. 7-14

[10] M. Schreuder, A. Riccio, M. Risetti, S. Dhne, A. Ramsay, J. Williamson, D. Mattia, M. Tangermann, User-centered Design in Braincomputer Interfacesa Case Study, Artificial Intelligence in Medicine 59 (2013) pp. 71-80, Special Issue: Brain-computer interfacing

[11] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kübler, J. Perelmouter, E. Taub, H. Flor, A Spelling Device for the Paralysed, Nature 398 (1999) pp. 297-298

[12] M. Bensch, A. A. Karim, J. Mellinger, T. Hinterberger, M. Tangermann, M. Bogdan, W. Rosenstiel, N. Birbaumer, Nessi: an EEG-controlled Web Browser for Severely Paralyzed Patients, Computational Intelligence and Neuroscience (2007)

[13] G. Kecklund, T. Åkerstedt, Sleepiness in Long Distance Truck Driving: an Ambulatory EEG Study of Night Driving, Ergonomics 36 (1993) pp. 1007-1017

[14] S. Haufe, M. S. Treder, M. F. Gugler, M. Sagebaum, G. Curio, B. Blankertz, Eeg Potentials Predict Upcoming Emergency Brakings during Simulated Driving, Journal of neural engineering 8 (2011) 056001

[15] E. Rodin, T. Constantino, J. Bigelow, Interictal Infraslow Activity in Patients with Epilepsy, Clinical Neurophysiology (2013)

[16] A. Serafini, G. Rubboli, G. L. Gigli, M. Koutroumanidis, P. Gelisse, Neurophysiology of Juvenile Myoclonic Epilepsy, Epilepsy & Behavior 28, Supplement 1 (2013) S30-S39

[17] M. L. Scheuer, Continuous EEG Monitoring in the Intensive Care Unit, Epilepsia 43 (2002) pp. 114-127

[18] U. Malinowska, C. Chatelle, M.-A. Bruno, Q. Noirhomme, S. Laureys, P. J. Durka, Electroencephalographic Profiles for Differentiation of Disorders of Consciousness, Biomedical Engineering online 12 (2013) p. 109

[19] W. O. Tatum IV, Long-Term EEG Monitoring: a Clinical Approach to Electrophysiology, J. Clinical Neurophysiology 18 (2001) pp. 442-455

[20] B. McCoy, C. D. Hahn, Continuous EEG Monitoring in the Neonatal Intensive Care Unit, J. Clinical Neurophysiology 30 (2013) pp. 106-114

[21] R. Boostani, K. Sadatnezhad, M. Sabeti, An Efficient Classifier to Diagnose of Schizophrenia Based on the EEG Signals, Expert Systems with Applications 36 (2009) pp. 6492-6499

[22] M. Sabeti, S. Katebi, R. Boostani, G. Price, A New Approach for EEG Signal Classification of Schizophrenic and Control Participants, Expert Systems with Applications 38 (2011) pp. 2063-2071

[23] V. Srinivasan, C. Eswaran, N. Sriraam, Artificial Neural Network-based Epileptic Detection Using Time-Domain and Frequency-Domain Features, Journal of Medical Systems 29 (2005) pp. 647-660

[24] S. Sun, C. Zhang, D. Zhang, An Experimental Evaluation of Ensemble Methods for EEG Signal Classification, Pattern Recognition Letters 28 (2007) pp. 2157-2163

[25] D. P. Subha, P. K. Joseph, R. Acharya, C. M. Lim, EEG Signal Analysis: A Survey, Journal of Medical Systems 34 (2010) pp. 195-212

[26] X. Xi, E. Keogh, C. Shelton, L. Wei, C. A. Ratanamahatana, Fast Time Series Classification using Numerosity Reduction, in: Proceedings of the 23rd International Conference on Machine Learning, ICML '06, ACM, New York, NY, USA (2006) pp. 1033-1040

[27] L. Devroye, L. Györfi, G. Lugosi, A Probabilistic Theory of Pattern Recognition, Springer Verlag (1996)

[28] G. H. Chen, S. Nikolov, D. Shah, A Latent Source Model for Nonparametric Time Series Classification, in: Advances in Neural Information Processing Systems 26 (2013) pp. 1088-1096

[29] N. Tomašev, D. Mladenić, Nearest Neighbor Voting in High Dimensional Data: Learning from Past Occurrences, Computer Science and Information Systems 9 (2012) pp. 691-712

[30] N. Tomašev, M. Radovanović, D. Mladenić, M. Ivanović, Hubness-based Fuzzy Measures for High-Dimensional k-nearest Neighbor Classification, International Journal of Machine Learning and Cybernetics (2013)

[31] N. Tomašev, M. Radovanović, D. Mladenić, M. Ivanović, A Probabilistic Approach to Nearest Neighbor Classification: Naive Hubness Bayesian k-nearest Neighbor, in: Proceeding of the CIKM conference

[32] K. Buza, A. Nanopoulos, L. Schmidt-Thieme, Insight: Efficient and Effective Instance Selection for Time-Series Classification, in: Proceedings of the 15th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining -Volume Part II, PAKDD'11, Springer-Verlag (2011) pp. 149-160

[33] K. Buza, A. Nanopoulos, L. Schmidt-Thieme, J. Koller, Fast Classification of Electrocardiograph Signals via Instance Selection, in: First International Conference on Healthcare Informatics, Imaging and Systems Biology, IEEE Computer Society, Washington, DC, USA (2011) pp. 9-16

[34] M. Radovanović, A. Nanopoulos, M. Ivanović, Time-Series Classification in Many Intrinsic Dimensions, in: Proceedings of the 10th SIAM International Conference on Data Mining (SDM) pp. 677-688

[35] H. Sakoe, S. Chiba, Dynamic Programming Algorithm Optimization for Spoken Word Recognition, Acoustics, Speech and Signal Processing 26 (1978) pp. 43-49

[36] K. A. Buza, Fusion Methods for Time-Series Classification, Peter Lang Verlag (2011)

[37] T. F. Smith, M. S. Waterman, Identification of Common Molecular Subsequences, Journal of molecular biology 147 (1981) pp. 195-197

[38] W. R. Cohen, P. S. Ravikumar, P. S. Fienberg, A Comparison of String Distance Metrics for Name-Matching Tasks, in: Proceedings of the IJCAI-03 Workshop on Information Integration on the Web (2003) pp. 73-78

[39] V. Levenshtein, Binary Codes Capable of Correcting Deletions, Insertions, and Reversals 10 (1966) pp. 707-710

[40] A. Barabási, Linked: How Everything Is Connected to Everything Else and What It Means for Business, Science, and Everyday Life, Plume (2003)

[41] J. B. Axelsen, S. Bernhardsson, M. Rosvall, K. Sneppen, A. Trusina, Degree Landscapes in Scale-Free Networks, Physical Review E-Statistical, Nonlinear and Soft Matter Physics 74 (2006) 036119

[42] A.-L. Barabási, E. Bonabeau, Scale-Free Networks, Sci. Am. 288 (2003) pp. 50-59

[43] X. He, J. Zhang, Why Do Hubs Tend to Be Essential in Protein Networks?, PLoS Genet 2 (2006)

[44] N. N. Batada, L. D. Hurst, M. Tyers, Evolutionary and Physiological Importance of Hub Proteins, PLoS Comput Biol 2 (2006) e88

[45] A. Nanopoulos, M. Radovanović, M. Ivanović, How does High Dimensionality Affect Collaborative Filtering?, in: Proceedings of the third ACM conference on Recommender systems, RecSys '09, ACM, New York, NY, USA (2009) pp. 293-296

[46] N. Tomašev, J. Rupnik, D. Mladenić, The Role of Hubs in Cross-Lingual Supervised Document Retrieval, in: Proceedings of the PAKDD Conference, PAKDD (2013)

[47] J. Aucouturier, F. Pachet, Improving Timbre Similarity: How High is the Sky?, Journal of Negative Results in Speech and Audio Sciences 1 (2004)

[48] Flexer A., Schnitzer D., Schlüter, J., A Mirex Meta-Analysis of Hubness in Audio Music Similarity, in: Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR'12

[49] Schedl M., Flexer A., Putting the User in the Center of Music Information Retrieval, in: Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR'12

[50] D. Schnitzer, A. Flexer, M. Schedl, G. Widmer, Using Mutual Proximity to Improve Content-based Audio Similarity, in: ISMIR'11, pp. 79-84

[51] N. Tomašev, R. Brehar, D. Mladenić, S. Nedevschi, The Influence of Hubness on Nearest-Neighbor Methods in Object Recognition, in: Proceedings of the 7th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), pp. 367-374

[52] N. Tomašev, D. Mladenić, Image Hub Explorer: Evaluating Representations and Metrics for Content-based Image Retrieval and Object Recognition, in: Proceedings of the ECML/PKDD Conference, Springer (2013)

[53] N. Tomašev, D. Mladenić, Exploring the Hubness-related Properties of Oceanographic Sensor Data, in: SiKDD conference (2011)

[54] M. Radovanović, A. Nanopoulos, M. Ivanović, Nearest Neighbors in High-Dimensional Data: The Emergence and Influence of Hubs, in: Proceedings of the 26rd International Conf. on Machine Learning, ACM, pp. 865-872

[55] J. E. Keller, M. R. Gray, J. A. Givens, A Fuzzy k-nearest-neighbor Algorithm, in: IEEE Transactions on Systems, Man and Cybernetics, pp. 580-585

[56] I. Rish, An Empirical Study of the Naive Bayes Classifier, in: Proc. IJCAI Workshop on Empirical Methods in Artificial Intelligence (2001)

[57] N. Tomašev, M. Radovanović, D. Mladenić, M. Ivanović, The Role of Hubness in Clustering High-Dimensional Data, Advances in Knowledge Discovery and Data Mining, Lecture Notes in Computer Science 6634 (2011) pp. 183-195

[58] M. Poulos, M. Rangoussi, N. Alexandris, A. Evangelou, Person Identification from the Eeg using Nonlinear Signal Classification, Methods of Information in Medicine 41 (2002) pp. 64-75

[59] K. Napierala, J. Stefanowski, Identification of Different Types of Minority Class Examples in Imbalanced Data, in: In Proceedings of Hybrid Artificial Intelligence Systems Conference, Springer Berlin (2012) pp. 139-150

[60] N. Tomašev, D. Mladenić, Class Imbalance and the Curse of Minority Hubs, Knowledge-Based Systems 53 (2013) 157-172

[61] N. Tomašev, K. Buza, K. Marussy, P. B. Kis, Hubness-aware Classification, Instance Selection and Feature Construction: Survey and Extensions to Time-Series, Feature selection for data and pattern recognition (2015) pp. 231-262

[62] N. Tomašev, D. Mladenić, Hub co-occurrence Modeling for Robust High-Dimensional Knn Classification, Machine Learning and Knowledge Discovery in Databases. Springer Berlin Heidelberg (2013) pp. 643-659

Big Data Testbed for Network Attack Detection

Dániel Csubák, Katalin Szücs, Péter Vörös, Attila Kiss

Department of Information Systems, Eötvös Loránd University
Pázmány Péter sétány 1/C, H-1117 Budapest, Hungary
csuby@caesar.elte.hu, szucsk@caesar.elte.hu, vopraai@inf.elte.hu, kiss@inf.elte.hu

Abstract: Establishing an effective defense strategy in IT security is essential on one hand, but very challenging on the other hand. According to the 2014 Cyberthreat Defence Report [1] that involved more than 750 security decision makers and practitioners, more than 60% of organizations had been breached in 2013. Big data analytics in security provides the possibility to gather and analyse massive amounts of digital information in order to predict and prevent these attacks. However, since collecting the needed data in an efficient, complete and reliable fashion encounters problems, the industry is lacking and could truly benefit from a tool offering benchmark data, provided in a platform, which would allow gauging and improving the effectiveness of security defence algorithms. To this end in this paper we introduce a platform that allows one to generate large parametrized datasets of simulated Internet traffic consisting of the combination of attack-free and malicious network traffic patterns. For the simulations we use the ns3 discrete-event network simulator. To make the resulting dataset appropriate for intrusion detection system benchmarking purposes we investigate the statistical characteristics of normal and intrusive traffic patterns. Finally we present a use case in which we validate our results.

Keywords: Network Traffic Simulation; Intrusion Detection; DDOS; NS-3

1 Introduction and Background

Internet traffic simulation has long been important for network intrusion detection experiments. For testing the efficiency of a newly developed algorithm, researchers are in need of an environment in which tests of an intrusion detection system can be performed. The produced dataset should contain attack-free background traffic as well as intentionally inserted malicious traffic. Many of the intrusion detection evaluation experiments have been conducted on proprietary datasets that are hard to access (due to privacy concerns) and hinder reproducible research. A great effort has been made to reduce this problem in 1998 and 1999 by MIT Lincoln Laboratory, under Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory (AFRL/SNHS) sponsorship, when they created the IDEVAL benchmark datasets [2-3]. To generate a corpus they followed the approach of recreating normal and attack patterns on a private network using real hosts, live attacks, and live background traffic. The generated data flow is similar to what can be seen between a small Air Force base and the Internet. IDEVAL has been used extensively for many years as the largest publicly available benchmark for intrusion detection system (IDS) performance evaluation, although in 2000 McHugh [4] has reported some issues about it, such as the lack of comparison of the benchmark data and real data. His suspicion that the dataset's statistical characteristics differ from live network traffic was later confirmed by Mahoney et al. [5] in 2003.

Typically there are four approaches to use background traffic in IDS testing: using no background traffic, using real traffic, using sanitized traffic and using simulated traffic [6]. When the tests are conducted without using background traffic as a reference condition, the IDS's hit rate can be determined, but nothing can be said about the false positive rate. Another drawback of this scheme is the assumption that the presence or the absence of background traffic does not change the performance of the system being analysed. Injecting attacks in real background traffic can overcome this difficulty, although, usually these experiments use a small set of victim machines, the data may contain malicious traffic or anomalies specific to the network, and even privacy concerns may arise. Sanitizing the real traffic by removing any sensitive data (for example using only the TCP headers) can reduce privacy problems, but it also can lead to unrealistic scenarios if too much data is removed, or it can unintentionally cause privacy risk if sanitization fails.

Using simulated traffic can overcome many of the above mentioned problems. It can be freely distributed without privacy concerns and it surely does not contain any unexplored attack. Another advantage is that the generated traffic can be later replayed to repeat the experiment. However, providing a testbed environment that is able to preserve all the important characteristics of real life Internet traffic is a great challenge. In [7] Floyd and Paxon present a detailed description of these simulation difficulties that are mainly present due to the heterogeneity and the rapid change of the Internet.

In the literature we can find basically two ways of network traffic generation. One of them is trace-based generation, where the generated traffic is the replication of some previously recorded real traffic traces. Generators like this for example are TCPReplay [8] and TCPivo [9]. The other solution is the analytical model-based approach. It this scheme, the generation is based on statistical models. Some widely used solutions like this are:

Traffic Generator (TG) that is capable to generate constant, uniform, exponential on/off UDP or TCP traffic [10].

MGEN is both a command line and GUI traffic generator. It provides programs for sourcing/sinking real-time multicast/unicast UDP/IP traffic flows [11].

RUDE/CRUDE: RUDE stands for Real-time UDP Data Emitter and CRUDE for Collector for RUDE. RUDE is a small and flexible program that generates traffic to the network, which can be received and logged on the other side of the network with the CRUDE. Currently these programs can generate and measure only UDP traffic [12].

Distributed Internet Traffic Generator (D-ITG) is a platform capable to produce traffic that accurately adheres to patterns defined by the inter departure time between packets and the packet size stochastic processes [13].

Internet Traffic Generator (ITG) allows the reproduction of TCP and UDP traffic and to accurately replicate appropriate stochastic processes for both Inter Departure Time and Packet Size random processes. ITG achieves performance comparable to that of RUDE/CRUDE, but additionally it makes available a greater number of traffic source types [14].

2 Our Network Traffic Generator

Our goal was to build a parametrizable tool that is able to generate realistic attack free HTTP traffic combined with DDOS attack traffic. To this end, we used the widely known NS3 event based network traffic generation tool as the basic environment. Since there is no generally accepted definition about how HTTP traffic has to be like, different servers have different user habits (e.g. Facebook is checked several times a day, while news are usually read in work after lunch) we tried to make our solution as configurable as possible.

Our background traffic generation model relies on the work of Choi and Limb [15]. This is one of the most wildly used traffic generation models. According to their study, a web browsing user can be described by an on-off process. The “on” stage starts as soon as the user requests a web page. Upon a request the main object is downloaded that contains the basic structure of the web page and the links to inline objects. The main object is followed by the inline objects that actualize the embedded content of the page, these can be scripts, images, etc. The “on” stage ends when all the elements of the requested page are downloaded. After that, a silent “off” stage takes place while the user is reading the retrieved content. Although the basic concept is still viable, the exact measurements of HTTP traffic made by Choi and Limb are considered obsolete because since the time of their research the nature of web traffic went through a significant change. The appearance of social networks and multimedia streaming, the more complex structure and the new services of web pages required new measurements to better describe web browsing behaviour. The actual measurements we relied on for DDOS detection system evaluation in Section 3 were conducted by Pries et al. in [16]. The presented results are based on the top one million visited web pages. The settings that we used are presented in Table 1.

Table 1
HTTP model parameters

Parameter	Best fit
Main object size	Weibull (28242.8,0.814944)
Number of main objects	Lognormal μ = 0.473844; σ = 0.688471
Inline object size	Lognormal μ = 9.17979; σ = 1.24646
Number of inline objects	Exponential μ = 31.9291
Reading time	Lognormal μ = 0.495204; σ = 2.7731

Several calculations have been made about the content of the configuration, as we wanted to keep it simple, yet customisable enough to fulfil any need of HTTP traffic. The resulted bunch of options is detailed in Table 2.

Table 2
Simulation parameters

numOfNodes	The number of nodes in the simulation, this includes the server and the dos clients as well
numOfDosClients	The number of nodes that will generate DoS traffic instead of general client behaviour
startTime	This parameter shows in which simulated second the clients start their work (does not affect DoS clients)
endTime	Which simulated second the clients stop their work
dosStartTime	Which simulated second the DoS clients start their work
dosStopTime	Which simulated second the DoS clients stop their work
serverPort	Number of port which the server listens on, and clients connect to
dataRate	Global link speed
delay	Number of seconds required for the first bit of a packet to arrive at the destination host
packetLoss	The probability that a packet gets lost while transferring
inlineObjectSizeLogNormalVariable	Due to studies [16] the number of inline objects in a mainline object is a Lognormal random variable. This variable's two parameter can be set in.

npageRequestsLogNormalVariable	Due to studies [16] the time between two mainline object requests is a Lognormal random variable. This variable's two parameter can be set in.
dosClient/ pageRequestsLogNormalVariable	The same as above, but it refers to DoS clients
avgClicks sleepTimeBetweenClickings	These parameters work together. Legal clients after the set number of average clicks will not request any other object for an average of SleepTimeBetweenClickings seconds.

A key in our data generator is to make it easy to use. We integrated our solution into a minimalistic web service, which results in a user friendly interface where without installing anything, a click is enough to start the simulation, and be able to download your results. This solution also, disencumbers our developer computers and puts the load to dedicated servers.

Since ns3 compiles for quite a long time, we wanted to give a solution which does not require recompilation too often but also stays parametrizable. Not just to fasten the simulations but to provide an easy summation of possible parameters, we implemented a configuration parser, which works with an XML file, and therefore does not require recompilation after a parameter changes.

2.1 Case Study

To show that our solution is easily configurable to generate any type of network traffic we simulated our local web server, and generated simulated data with the attributions of the original. In this section we are going to discuss the details of this experiment.

To gather a fair amount of real network data we monitored all the packets received by our local web server, for about a month-long period. Figure 1 shows the frequency of page requests that arrived at our server during the monitored time interval.

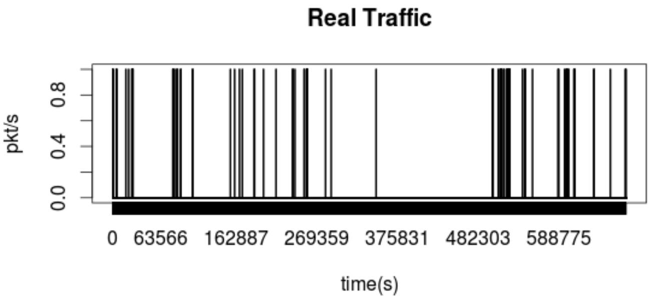


Figure 1

Main object requests per second in the real data

To estimate the necessary parameters for the simulation we filtered the data for the http requests. We used R's "fitdstrplus" library to give the best estimation for the distribution of the time between two main object requests, the average number of consecutive main object requests and for the "sleepTimeBetweenClickings" that corresponds to the time between two visits of the same client to the web page. The results are presented in Table 3.

Table 3
Parameter estimates

Parameter	Best fit
MainpageRequests	Exponential μ = 857.5256
avgClick	Pareto mean = 5; shape = 1.1; bound = 20
sleepTimeBetweenClickings	Exponential μ = 20696.83

Figure 2 shows the frequency of the page requests in the simulated traffic with the above configuration.

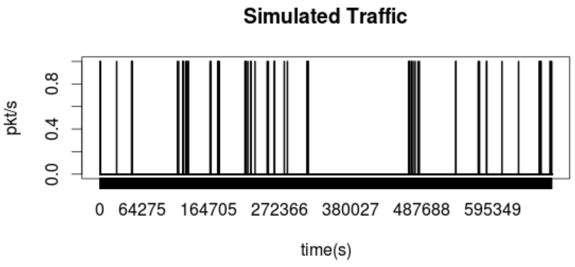


Figure 2
Main object requests per second in the simulated data

2.1.1 Self-Similarity and Long-Range Dependence

The distribution of traffic on the Internet commonly exhibits self-similarity. The first observation of the phenomena was made by Leland and Wilson [17]. They commented in detail on the presence of "burstiness" across an extremely large range of time scales in the data collected at Bellcore Morristown Research and Engineering Center on several Ethernet LANs. The first statistically rigorous analysis was made in 1994 by Leland et al. [18] on the same dataset. Their research pointed out that the Ethernet LAN traffic is self-similar, irrespective of where and when the data were collected in the network. They showed that the Hurst parameter better describes the fractal-like nature of the traffic and able to capture its "burstiness", when other methods (the index of dispersion, the peak-to-mean-ratio or the coefficient of variation for inter arrival times) fail to do so. Their observations have been supported by later research and their findings have led the research community to make significant efforts towards developing appropriate mathematical and statistical techniques that provide a network-related understanding of the observed self-similar scaling behaviour [19-26]. A mathematical description of self-similarity used by Bai and Shami in 2013 [27, 28] for network traffic simulation is the following:

Let X_i (i = 1, 2, …) be an increment process and $X_j^{(m)}$ (j = 1, 2, …) be another process, which is obtained by averaging the values in non-overlapped blocks of size m in X_i, i.e.:

$$X_j^{(m)} = \frac{1}{m}\left(X_{jm-m+1} + X_{jm-m+2} + \dots + X_{jm}\right). \tag{1}$$

The process X_i is said self-similar if $X_j^{(m)}$ is similar in distribution to $m^{H-1}X_i$, where m ($m \geq 1$) is the scale parameter and H is the Hurst exponent. In a more understandable form it implies:

$$\mathrm{Var}\left(X_j^{(m)}\right) = m^{2H-2}\,\mathrm{Var}\left(X_i\right). \tag{2}$$

Another wildly researched characteristic of network traffic is long-range dependence (LRD). Its discovery in network traffic, has fundamentally changed the conventional wisdom by stating that the correlation of packet inter arrivals decays slower than in traditional traffic (e.g., Markov) models. LRD and self-similarity are strongly related. From the definitions of [29] we use the inference that LDR characterizes a time series if it holds for the Hurst exponent H that $0.5 < H \leq 1$. As H approaches 1, the dependence becomes stronger.

During background traffic generation we found it important to test if the simulated data preserved the LRD characteristics of the real data. To test LRD of the packet arrivals we estimated the Hurst exponent using the aggregated variance method of the fArma R library. The estimation of the real and the simulated traffic's Hurst exponent satisfied our expectations since it resulted similar numbers, 0.57 and 0.52 respectively.

3 DDoS Detection

3.1 Brief introduction to DDoS Attacks

Denial-of-Service (DoS) and Distributed Denial-of-Service (DDoS) attacks are attempts to make machines, and network-related resources or services (e.g. Webserver) unavailable for its legal users.

DDoS attacks are launched by more than one hosts, but more often the attacker uses botnets (network of zombie hosts, infected by some kind malware) and thousands of hosts from all around the world. According to [30], in 2014 the frequency of recognized DDoS attacks had reached an average rate of 28 per hour.

In this paper we focus on HTTP GET flood attacks [31]. It is an application layer attack and in this case the attacker's hosts send seemingly legitimate HTTP GET requests to the webserver that they aim to attack. These attacks do not use malformed packets, or spoofing, and they require less bandwidth than other attacks to bring down the targeted servers, but they require some understanding of the targeted application. This type of attack is harder to detect than some others, mainly because it uses valid HTTP GET requests, and it usually doesn't generate significant network traffic. These types of measurements don't really affect DDoS detection, that's why we used the frequency of the page requests in our detector.

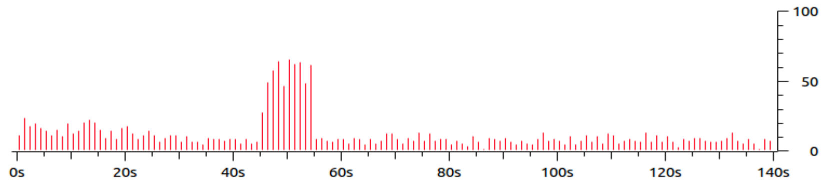


Figure 3

Packet per second rate of a generated DDoS attack

Figure 3 shows the packet per second rate of a generated DDoS attack. In this case, we used the recorded traffic (~100 MB) of the server that was attacked by some malicious clients between time 45 and 55, which is illustrated by higher lines in the plot. The traffic before, and after the DDoS is normal.

3.2 Detecting DDoS attacks with Snort

Snort [32] is widely used, free and open source network intrusion detection software. It is capable of real time traffic analysis and packet logging, and because of its huge community, it is one of the most widely developed intrusion detection systems in the world.

Snort has a set of rules defined by the user, and the network traffic is analysed against these rule-sets. After the detection special actions can take place based on what has been identified.

Snort has a built-in set of rules for DDoS detection, which could be used for validation, but these rules were very generic, so we tried another approach.

Our approach consists of two parts:

- Analysis of a training set of normal traffic data
- Use the parameters from phase 1 to detect DDoS attacks with Snort

In the 1st phase we analysed normal traffic data for parameters which will be used for detection. In this case we used only the packet/sec rate of the most active client from the most loaded moment of the server. To achieve this we used a python and the "dpkt" module (python-dpkt package on ubuntu 14.04) for "pcap" analysis.

In the 2nd phase we used the parameter from above as a base threshold for a rate limit-like approach. Furthermore we used a multiplier for the parameter, but its value was decided by empirical methods for every unique case. We generated rules for Snort, and after we started it with the given ruleset, Snort raises alerts if any of the clients reaches the packet rate.

In the case of datasets larger than many gigabytes, it is hard to check the generated traffic in the way we have done it in the case of Figure 3, so instead of wireshark, we used Snort to check for DDoS attacks. Snort was able to analyze these pcap files in reasonable time, with the method mentioned above, so we could easily validate our generator.

Conclusions

We deeply studied the different types of web traffic, with outstanding attention to the relation between HTTP and the many DoS/DDoS attacks. Simulations are much cheaper, quicker and easier to use than real systems, so we considered the need of a tool, which is able to generate valid and parametrisable HTTP traffic, with customisable DDoS attackers.

We extended the existing NS3 with our classes, and implemented our own XML configurable simulator and a webservice to make our simulator easy to use.

The simulator is perfect for generating hundreds of GB traffic. Since such volume of data cannot be verified manually, we implemented a python script that is able to generate Snort rules.

References

[1] CyberEdge: 2014 Cyberthreat Defence Report for North America & Europe, a CyberEdge report sponsored by ForeScout Technologies, Inc., 2014

[2] R. Lippmann, et al.: The 1999 DARPA Off-Line Intrusion Detection Evaluation, Computer Networks 34(4) 579-595, 2000. Data is available at http://www.ll.mit.edu/IST/ideval/

[3] Lippmann, Richard P., et al.: Evaluating Intrusion Detection Systems: The 1998 DARPA Off-Line Intrusion Detection Evaluation, DARPA Information Survivability Conference and Exposition, 2000, DISCEX'00, Proceedings. Vol. 2, IEEE, 2000

[4] McHugh, John: Testing Intrusion Detection Systems: a Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. ACM Transactions on Information and System Security 3.4 (2000): 262-294

[5] Mahoney, Matthew V., and Philip K. Chan: An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection. Recent Advances in Intrusion Detection. Springer Berlin Heidelberg, 2003

[6] Hu, Vincent, et al.: An Overview of Issues in Testing Intrusion Detection Systems (2003)

[7] Floyd, Sally, and Vern Paxson: Difficulties in Simulating the Internet. IEEE/ACM Transactions on Networking (TON) 9.4 (2001): 392-403

[8] URL http://tcpreplay.synfin.net/

[9] URL http://www.thefengs.com/wuchang/work/tcpivo/

[10] URL http://www.postel.org/tg/

[11] URL http://www.nrl.navy.mil/itd/ncs/products/mgen

[12] URL http://rude.sourceforge.net/

[13] Avallone, Stefano, et al.: D-ITG Distributed Internet Traffic Generator. Quantitative Evaluation of Systems, 2004, QEST 2004, Proceedings, First International Conference on the. IEEE, 2004

[14] Avallone, Stefano, Antonio Pescape, and Giorgio Ventre: Analysis and Experimentation of Internet Traffic Generator. Proc. of New2an (2004): 70-75

[15] Choi, Hyoung-Kee, and John O. Limb "A Behavioral Model of Web Traffic."Network Protocols, 1999 (ICNP'99) Proceedings Seventh International Conference on. IEEE, 1999

[16] Pries, Rastin, Zsolt Magyari, and Phuoc Tran-Gia "An HTTP Web Traffic Model based on the Top One Million Visited Web Pages." Next Generation Internet (NGI) 2012 8th EURO-NGI Conference on. IEEE, 2012

[17] Leland, Will E., and Daniel V. Wilson: High Time-Resolution Measurement and Analysis of LAN Traffic: Implications for LAN Interconnection, INFOCOM'91, Proceedings, Tenth Annual Joint Conference of the IEEE Computer and Communications Societies. Networking in the 90s, IEEE, 1991

[18] Leland, Will E., et al.: On the Self-Similar Nature of Ethernet Traffic (extended version) Networking, IEEE/ACM Transactions on 2.1 (1994): 1-15

[19] M. E. Crovella and A. Bestavros: Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes, ACM SIGMETRICS, Vol. 24, No. 1, pp. 160-169, 1996

[20] P. Barford and M. Crovella: Generating Representative Web Workloads for Network and Server Performance Evaluation, ACM SIGMETRICS, Vol. 26, pp. 151-160, 1998

[21] P. Barford, A. Bestavros, A. Bradley, and M. E. Crovella: Changes in Web Client Access Patterns: Characteristics and Caching Implications, World Wide Web, Special Issue on Characterization and Performance Evaluation, Vol. 2, pp. 15-28, 1999

[22] C. R. Cunha, A. Bestavros, and M. E. Crovella: Characteristics of WWW Client-based Traces, 1995, technical Report TR-95-010, Boston University Computer Science Department

[23] T. Ott, T. Lakshman, and L. Wong: SRED: Stabilized RED, in IEEE INFOCOM, 1999, pp. 1346-1355

[24] P. Barford and M. E. Crovella: A Performance Evaluation of HyperText Transfer Protocols, in ACM SIGMETRICS, 1999, pp. 188-197

[25] Paxson, Vern, and Sally Floyd. "Wide Area Traffic: the Failure of Poisson Modeling." IEEE/ACM Transactions on Networking (ToN) 3.3 (1995): 226-244

[26] Riedi, Rudolf H., and Walter Willinger.: Toward an Improved Understanding of Network Traffic Dynamics. Self-Similar Network Traffic and Performance Evaluation (2000): 507-530

[27] Bai, Xiaofeng, and Abdallah Shami.: Modeling Self-Similar Traffic for Network Simulation. arXiv Preprint arXiv:1308.3842 (2013)

[28] P. Orenstein, H. Kim and C. L. Lau: Bandwidth Allocation for Self-Similar Traffic Consisting of Multiple Traffic Classes with Distinct Characteristics, IEEE GLOBECOM 2001, Vol. 4, pp. 2576-2580, December 2001

[29] Karagiannis, Thomas, Mart Molle, and Michalis Faloutsos.: Long-Range Dependence Ten Years of Internet Traffic Modeling. Internet Computing, IEEE 8.5 (2004): 57-64

[30] Preimesberger, Chris (May 28, 2014) "DDoS Attack Volume Escalates as New Methods Emerge". *Eweek*

[31] Incapsula - DDoS Attack Glossary – HTTP Flood http://www.incapsula.com/ddos/attack-glossary/http-flood.html

[32] Snort: - URL: https://www.snort.org

A Formal Representation for Structured Data

János Demetrovics

MTA SZTAKI, Lágymányosi u. 11, H-1111 Budapest, Hungary
e-mail: demetrovics.janos@sztaki.mta.hu

Hua Nam Son, Ákos Gubán

Budapest Business School, Buzogány u. 11-13, H-1149 Budapest, Hungary
e-mail: Hua.NamSon@pszfb.bgf.hu, guban.akos@pszfb.bgf.hu

Abstract: Big Data is a technology developed for 3-V management of data by which large volumes and different varieties of data would be processed in optimal velocity. The data to be dealt with may be structured or unstructured. Relational databases (spreadsheets) are typical examples of structured data and the methods, as well as the techniques for researches of relational database management are well-known. In this paper, we describe a formalism, by which, structured data, can be considered as a directly generalized model of relational databases. A higher leveled structured data, in our generalization, are defined recursively as a set or a queue of lower leveled structured data. Consequently, our study proves that many concepts and results of relational database management can be transferred to structured data, accordingly to this generalization. The sub data, the components of structured data, the functional dependencies between structured data, as well as the keys data in structured data are defined and studied. Alternately, some concepts that are defined here for structured data can be applied for relational databases, as a special case. In this paper, some operations on structured data and the homomorphism between structured data are defined and studied that appear to be quite suitable for relational databases. In fact, the formalization introduced here, offers effective methods for further structural, algebraic researches of structured data.

Keywords: Data management; Big Data; Structured data; Relational database; Lattice; Partially ordered set

1 Introduction

Big Data storage and Big Data processing model design are essential problems of Big Data management (see, for example, [9]). Structured data in a common sense, are complex data, constructed by atomic data residing in fixed fields within a definite structure. In contrast to semi-structured data and unstructured data, structured data is taking new and special roles in the world of Big Data. Spreadsheets and relational databases are evident examples of structured data. A data table in relational databases as defined by Codd [4] in fact can be considered as a set of its columns (or its rows) that in their turn are the queues of atomic data. By using the characteristics of the data structure, lots of properties of relational databases were discovered and vigorously studied in 1990s ([1], [10]). The studies were focused on keys, functional dependencies between attributes and normalization of databases (see [2], [7]). The lattice-type properties of functional dependencies in relational databases were studied thoroughly in [8]. We can note that many properties of the functional dependencies between attributes are induced by the lattice structure of the data.

This remark inspires an idea: not only for relational databases but in general, it is the structure of data that determines the dependencies between their components and other structural properties of them. Thus the first question to be considered is the definition of the concept of structure. Structure is an indefinite concept and one can hardly give a sufficient definition that concerns all possible structures. This paper deals only with those structured data that are built up recursively as *sets,* or *queues* of other less complex data. Thus, the data can be defined in different orders of complexity: atomic data are structured data of lowest order, a set or a queue of atomic data is structured data of first order, while the relations in relational databases being sets of data columns (or data rows) are structured data of second order, etc. This definition does not cover all structured data, but it deals with rather wide range of data. The relational databases as sets of interconnected relations are in fact 4-order data.

The formulation of structured data proposes an efficient approach for structural studies. On one hand, the approach reveals the lattice characteristics of the well-known properties of relational databases. On the other hand, the approach enables the generalization of the concepts and properties, well-known for relational databases, into those of structured data.

In Section 2 we generalize the concept of relations in structured data as defined later. It is pointed out in this section, that there is a natural order between the relations and all relations can be represented in a linear form or by tree graphs. In Section 3, structured data are defined. In fact, structured data are generalized relations with all their participant relations. Structured data may be considered as algebraic objects in which the various operations and homomorphism should be studied. The concept of sub data and components of data, as well as the queries on data, are also defined here. In this generalized model we study the dependency between components of data. The key components of structured data are defined as those components, that all other components, depend. We show in this section that relational databases are really special cases of structured data, where the dependencies between attributes, are in fact, dependencies of partial order types. Some aspects for further research, as well as open problems are discussed in the Conclusions.

2 Relations

In this section we propose a generalization of the concept of a relation. Relations are defined recursively accordingly to their order. The first-order relation on a set is a collection of subsets or queues of elements of the given set, while the higher order relations are collection of subsets or queues of lower order relations. Thus relations are defined based on subsets or queues of elements.

2.1 Sets and Queues of Atomic Data

By traditional algebraic definition the relation of elements is a set of n-tuples of elements. In a more generalized sense, a relation of elements can be understood as a set of finite tuples of elements.

Definition 1: For a set V, let $V^\infty = \bigcup_{i=1}^{\infty} V^i$. V^∞ denotes the set of all finite queues of elements in V.

Remark:

1. Below, we should distinguish the sets and the queues of elements: the sets and the queues of elements are parenthesized by { } and by < >, respectively.
2. By Definition 1, in general, $\{u, v\} = \{v, u\}$ and $\{v, v\} = \{v\}$, while $\langle u, v\rangle \neq \langle v, u\rangle$ and $\langle v, v\rangle \neq \langle v\rangle$
3. We accept $\{v\} = \langle v\rangle$

Definition 2:

Let U, V be two sets of atomic data, $U \subseteq V$. For $s \subseteq V^\infty$ the *projection* of s denoted by $Pr_U(s)$ is defined as follows:

i. If $s = \langle v_1, v_2, \dots, v_k\rangle \in V^\infty$, then $Pr_U(s) = \langle v_i | v_i \in U\rangle$.
ii. If $s = \{s_1, s_2, \dots, s_q\} \subseteq V^\infty$ then $Pr_U(s) = \{Pr_U(s_1), Pr_U(s_2), \dots, Pr_U(s_q)\}$.
iii. If $s = \langle v_1, v_2, \dots, v_k\rangle \in V^\infty$ or $s = \{s_1, s_2, \dots, s_q\} \subseteq V^\infty$ then $A(s)$ denotes the set of all atomic data in V that appears in s.

Remark:

1. If $r = Pr_U(s)$ then r is a sub-queue of s in the case s is a queue, and r is a set of sub-queues of s in the case s is a set of sub-queues.
2. If $r = Pr_U(s)$ and $s = Pr_W(t)$ then $r = Pr_{U\cap W}(t)$.
3. If r, s are finite subsets of V^∞ and $r = Pr_U(s)$, $s = Pr_W(r)$ then $s = r$ and $A(s) = A(r) \subseteq U \cap W$.

   This is evident if s, r are queues in V^∞.

If $r, s \subseteq V^{\infty}$, $r = \{r_1, r_2, \dots, r_p\}, s = \{s_1, s_2, \dots, s_q\}$, then by definitions for all k there is i such that $r_k = Pr_U(s_i)$ and vice versa, for all i there is k such that $s_i = Pr_W(r_k)$. Thus, for all k there are $r_{k_1} = r_k, r_{k_2}, \dots$ and $s_{k_1}, s_{k_2}, \dots$ such that $r_{k_j} = Pr_U\left(s_{k_j}\right)$ and $s_{k_j} = Pr_W\left(r_{k_{j+1}}\right)$, $j = 1,2, \dots$. By previous remarks one can see that $r_{k_j} = Pr_{U \cap W}\left(r_{k_{j+1}}\right)$. Since $r_{k_{j+1}} = Pr_{U \cap W}\left(r_{k_{j+2}}\right)$, we have $r_{k_j} = r_{k_{j+1}} = Pr_{U \cap W}\left(r_{k_{j+2}}\right)$. Now it is easy to see that $r_k = r_{k_1} = r_{k_2} = \cdots = s_{k_1} = s_{k_2} = \cdots$, i.e. r_k is a queue in s. The similar explain proves that arbitrary queue s_{i} in s is queue in r. We have $r = s$. By the proof we see also that $A(s) = A(r) \subseteq U \cap W$.

4. By 2. and 3. if we define a relation on finite subsets of V^{∞} as follows:

   $r \lhd s \Leftrightarrow \exists U : r = Pr_U(s)$

   then $\lhd$ is a partial order on the finite subsets of V^{∞}.

2.2 m-Order Relations

The relations that we define below are a generalization of the relations (spreadsheets) in relational modeling.

Definition 3:

1. A *0-order relation* over V is V. The set of all 0-order relations over V is denoted by $\mathcal{R}^0(V)$. Thus $\mathcal{R}^0(V) = V$.
2. For $m \geq 0$ an *(m+1)-order relation* over V is a finite subset of $\mathcal{R}^{\mathrm{m}}(V)$ or a finite queue of elements of $\mathcal{R}^{\mathrm{m}}(V)$. The set of all ($m$+1)-order relations over V is denoted by $\mathcal{R}^{\mathrm{m}+1}(V)$.
3. $\mathcal{R}^{\infty}(V) = \bigcup_{\mathrm{m}=0}^{\infty} \mathcal{R}^{\mathrm{m}}(V)$.

In words, a relation of higher order over V is a finite subset or a finite queue of lower order relations. $\mathcal{R}^{\infty}(V)$ denotes the set of all relations over V.

Remark:

1. By Definition 1 we have $v = \{v\} = \langle v \rangle$, therefore $V \subseteq \mathcal{R}^1(V)$ and so on, $\mathcal{R}^m(V) \subseteq \mathcal{R}^{m+1}(V)$ for all $m \geq 0$.
2. Two relations of different order are different and are named differently, exceptionally, since $r = \{r\} = \langle r \rangle$ for all $r \in \mathcal{R}^m(V)$, we have also $r \in \mathcal{R}^{m+1}(V)$.

Definition 4:

Let $U \subseteq V$. For $s \in \mathcal{R}^{\infty}(V)$ the *projection* of s denoted by $Pr_U(s)$ is defined as follows:

i. If $s \in \mathcal{R}^1(V)$ then $Pr_U(s)$ is defined as in Definition 2.

ii. If $s = \langle v_1, v_2, \dots, v_k \rangle \in \mathcal{R}^{\mathrm{m}+1}(V)$, $v_i \in \mathcal{R}^{\mathrm{m}}(V)$, then $Pr_U(s) = \langle Pr_U(v_i) | \mathrm{i} = 1,2, \dots, \mathrm{k} \rangle$.

iii. If $s = \{s_1, s_2, \dots, s_q\} \in \mathcal{R}^{\mathrm{m}+1}(V)$ where $s_1, s_2, \dots, s_q$ are queues on $\mathcal{R}^{\mathrm{m}}(V)$, then $Pr_U(s) = \{Pr_U(s_1), Pr_U(s_2), \dots, Pr_U(s_q)\}$.

The projection of a relation on $U \subseteq V$ can be obtained from the given relation by deleting all atomic data that are not in U.

2.3 Partial Order on Relations

We show that on $\mathcal{R}^{\infty}(V)$ there exists a partial order between the relations.

Definition 5: For $r, s \in \mathcal{R}^{\infty}(V)$ we write $r \leq s$ if

i. $r = s$, or

ii. There exist $s_0, s_2, \dots, s_k \in \mathcal{R}^{\infty}(V)$, $s_0 = r$, $s_{\mathrm{k}} = s$, such that s_{i} is a finite subset or a finite queue of $s_{\mathrm{i}-1}$ and other elements of $\mathcal{R}^{\infty}(V)$, for all $i = 1,2, \dots, k$.

In words, $r \leq s$ if r appears in the presentation of s. We have a trivial theorem:

Theorem 1: $\leq$ is a partial order on $\mathcal{R}^{\infty}(V)$.

2.4 Representation of Relations

The relations can be represented by linear expressions and by tree graphs.

Definition 6 (linear representations of relations):

i. For $r \in \mathcal{R}^0(V)$, $r = v \in V$ the linear representation of r is the expression $l(r) = v$.

ii. For $r \in \mathcal{R}^{m+1}(V)$ the linear representation of r is the expression $l(r)$:

$$l(r) = \begin{cases} \langle l(r_1), l(r_2), \dots, l(r_{\mathrm{k}}) \rangle \text{ if } r = \langle r_1, r_2, \dots, r_{\mathrm{k}} \rangle, r_{\mathrm{i}} \in \mathcal{R}^m(V) \\ \{l(r_1), l(r_2), \dots, l(r_{\mathrm{k}})\} \text{ if } r = \{r_1, r_2, \dots, r_{\mathrm{k}}\}, r_{\mathrm{i}} \in \mathcal{R}^m(V) \end{cases}$$

The set of all representations of on r is denoted by $\mathcal{P}(r)$.

In fact, the representations on V are the expressions that can be defined, formally, as follows:

Definition 7:

i. If $v \in V$ then the expression v is a formal representation. The set of all expressions of this form is denoted by $\mathcal{P}^0(V)$.

ii. If $r_1, r_2, \dots, r_k \in \mathcal{P}^i(V), i \leq m$, then the expressions of the form $\{r_1, r_2, \dots, r_k\}$ and $\langle r_1, r_2, \dots, r_k \rangle$ are formal representations on V. The set of all expressions of this form is denoted by $\mathcal{P}^{m+1}(V)$.

iii. All formal representations on V are defined as in i. and ii.

The set of all formal representations on V is denoted by $\mathcal{P}(V)$, i.e.

$\mathcal{P}(V) = \bigcup_{m=1}^{\infty} \mathcal{P}^m(V)$.

Remark:

1. The representation of relations is not unique: each relation has many representations.
2. The linear representations of relations over V are formal representations on V and vice versa, each formal representation on V represents some relation over V.

For $p, q \in \mathcal{P}$ we write $p \sim q$ if:

i. $p = \{q\}$ or $p = \langle q \rangle$, or $q = \{p\}$ or $q = \langle p \rangle$,

ii. $p = \{u_1, u_2, \dots, u_k\}$, $q = \{v_1, v_2, \dots, v_k\}$, $u_{\mathrm{i}}, v_{\mathrm{i}} \in V$ and $v_1, v_2, \dots, v_k$ is a permutation of $u_1, u_2, \dots, u_k$, or

iii. $p = \langle u_1, u_2, \dots, u_k \rangle$, $u_{\mathrm{i}} \in V$ and $q = p$, or

iv. $p = \{r_1, r_2, \dots, r_m\}$, $q = \{s_1, s_2, \dots, s_m\}$, $r_{\mathrm{i}}, s_{\mathrm{i}}$ are formal representations on V and there exists a permutation $u_1, u_2, \dots, u_m$ of $r_1, r_2, \dots, r_m$ such that $u_i \sim s_i$ for all $i = 1, 2, .., m$.

v. $p = \langle r_1, r_2, \dots, r_m \rangle$, $q = \langle s_1, s_2, \dots, s_m \rangle$ and $r_i \sim s_i$ for all $i = 1, 2, .., m$.

Theorem 2:

1. $\sim$ is an equivalence on $\mathcal{P}$.
2. $p \sim q \Leftrightarrow \exists r \in \mathcal{R}^{\infty}(V)\colon p, q \in \mathcal{P}(r)$.
3. There exists an algorithm that constructs $r \in \mathcal{R}^{\infty}(V)$ such that $l(r) = p$ for $p \in \mathcal{P}$.
4. There exists an algorithm that decides if $p \sim q$ for $p, q \in \mathcal{P}$.

We define a partial order on $\mathcal{P}$: For $p, q \in \mathcal{P}$ we write $p \leq q$ if

i. $p = q$, or

ii. There exist $p_1, p_2, \dots, p_k \in \mathcal{P}(V)$, $p_1 = p$, $p_{\mathrm{k}} = q$, such that $p_{\mathrm{i}} \in \mathcal{P}^{m+i}(V)$ and $p_{\mathrm{i+1}} = p_{\mathrm{i}}$ or $p_{\mathrm{i+1}}$ is the expression of the form $\{\dots, p_{\mathrm{i}}, \dots\}$ or $\langle \dots, p_{\mathrm{i}}, \dots \rangle$ for all $i = 1, \dots, k-1$.

$\leq$ is a partial order on $\mathcal{P}$. We have:

Theorem 3: For all $r, s \in \mathcal{R}^{\infty}(V)$ we have:

$r \leq s \Leftrightarrow l(r) \leq \mathrm{l}(s)$

The relations can also be represented by tree graphs as follows:

Definition 8 (graphical representation of relations): To each relation r we associate a tree graph $t(r)$ as follows:

i. For $r = v \in \mathcal{R}^0(V)$ the graph $t(v)$ is the tree graph that contains single node labeled by v.

ii. For $r \in \mathcal{R}^{m+1}(V), r = \langle r_1, r_2, \dots, r_k \rangle, r_i \in \mathcal{R}^m(V)$ the graph $t(r)$ is the tree graph with the root r that is connected directly to the nodes to that we attach the trees $t(r_1), t(r_2), \dots, t(r_k)$ from left to right.

iii. For $r \in \mathcal{R}^{m+1}(V), r = \{r_1, r_2, \dots, r_k\}, r_i \in \mathcal{R}^m(V)$, the graph $t(r)$ is the tree graph with the root r that is connected directly to the nodes to that we attach the trees $t(r_1), t(r_2), \dots, t(r_k)$ for $i = 1,2 \dots k$.

The set of all graphs representing r is denoted by $\mathcal{T}(r)$. We have:

Theorem 4:

1. If $\mathcal{T} = \mathcal{T}(r)$ is a tree that represents r, then
   i. The leaves are labeled by elements of V.
   ii. Only the labels of the leaves may be repeated.
2. If $\mathcal{T}$ is a tree graph with labeled nodes that satisfies i, ii conditions, then there exists a relation r on V such that $\mathcal{T} = \mathcal{T}(r)$.
3. For two relations $r, s \in \mathcal{R}^\infty(V)$ $r \leq s$ if and only if $\mathcal{T}(r)$ is a sub-tree of $\mathcal{T}(s)$.

Example 1:

A data table, i.e. a relation in relational database, may be considered as a relation defined in Definition 3: If $r = \{A_1, A_2, \dots, A_m\}$ is a relation with $A_1, A_2, \dots, A_m$ columns, where $A_i = (a_{1i}, a_{2i}, \dots, a_{ni})$ then r is 2-order relation

$$r = \{\langle a_{11}, a_{21}, \dots, a_{n1} \rangle, \langle a_{12}, a_{22}, \dots, a_{n2} \rangle, \dots, \langle a_{1m}, a_{2m}, \dots, a_{nm} \rangle\}$$

The tree graph of r is

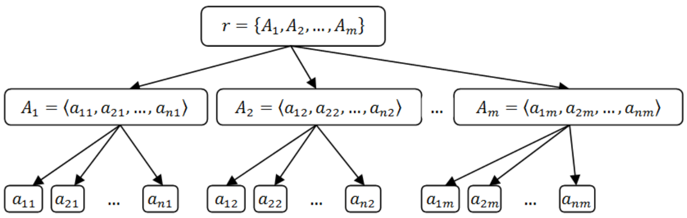

$r = \{A_1, A_2, \dots, A_m\}$
$A_1 = \langle a_{11}, a_{21}, \dots, a_{n1} \rangle$
$A_2 = \langle a_{12}, a_{22}, \dots, a_{n2} \rangle$
...
$A_m = \langle a_{1m}, a_{2m}, \dots, a_{nm} \rangle$
a_{11} a_{21} ... a_{n1}
a_{12} a_{22} ... a_{n2}
a_{1m} a_{2m} ... a_{nm}

Figure 1

Tree graph of a relation in a relational database

3 Structured Data

Structured data are sets of relations with specified systems of participant relations. Formally, we have:

Definition 9: Let V be a set of atomic data.

1. A structured data is a finite sequence of relations

$$\alpha = [r_0, r_1, \dots, r_m]$$

where

i. r_0 is finite subset of V,

ii. For all $i = 1, \dots, m$, r_i is a relation constructed by atomic data in r_0 and preceding data, i.e. r_i is a finite set or a finite queue of elements from $r_0 \cup \{r_1, r_2, \dots, r_{i-1}\}$.

The set of all structured data over V is denoted by S_V.

2. A structured data $\alpha = [r_0, r_1, \dots, r_m]$ is *simple* if

iii. $r_j \neq r_i$ for $i \neq j$ and r_j, r_i are not elements from r_0

In a simple structured data the condition iii. guarantees that only elements from r_0 may be repeated.

Definition 10: Let $\alpha = [r_0, r_1, \dots, r_m]$ be a structured data.

1. By $l(\alpha) = [l(r_0), l(r_1), \dots, l(r_m)]$ we denote the linear representation of α where $l(r_i)$ is some linear representation of r_i for all $i = 1, \dots, m$.

2. By $t(\alpha) = \{t(r_0), t(r_1), \dots, t(r_m)\}$ we denote the multi tree of α which is constructed as follows:

   i. $t(r_0)$ contains the nodes marked by elements in r_0,

   ii. $t(r_i)$ is the tree of r_i for all $i = 1, \dots, m$.

Example 2: In Table 1 we can see a linear representation of a structured data:

Table 1
Linear representation of a structured data

$\alpha = [r_0, r_1, r_2, r_3, r_4, r_5]$	$l(\alpha) = [l(r_0), l(r_1), l(r_2), l(r_3), l(r_4), l(r_5)]$
$r_0 = \{v_1, v_2, v_3, v_4\}$	$l(r_0) = \{v_1, v_2, v_3, v_4\}$
$r_1 = \langle v_1, v_2 \rangle$	$l(r_1) = \langle v_1, v_2 \rangle$
$r_2 = \langle v_2, v_3 \rangle$	$l(r_2) = \langle v_2, v_3 \rangle$
$r_3 = \{v_1, r_1, r_2\}$	$l(r_3) = \{v_1, \langle v_1, v_2 \rangle, \langle v_2, v_3 \rangle\}$
$r_4 = \{v_2, \langle v_4, r_3 \rangle\}$	$l(r_4) = \{v_2, \langle v_4, \{v_1, \langle v_1, v_2 \rangle, \langle v_2, v_3 \rangle\}\rangle\}$
$r_5 = \{r_4, \langle r_1, r_3, r_4 \rangle\}$	$l(r_5) = \{\{v_2, \langle v_4, \{v_1, \langle v_1, v_2 \rangle, \langle v_2, v_3 \rangle\}\rangle\},$ $\langle\langle v_1, v_2 \rangle, \{v_1, \langle v_1, v_2 \rangle, \langle v_2, v_3 \rangle\}, \{v_2, \langle v_4, \{v_1, \langle v_1, v_2 \rangle, \langle v_2, v$

Example 3: Let **Atomic data =** {N, BD, Na, NID, NV, NC, NM, T, C, NDL} where N, BD, Na, NID, NV, NC, NM, T, C, NDL is the abbreviation of Name, Birth date, Nationality, Number of ID card, Number of Vehicle registration card, Number of chassis, Number of motor, Type, Category of vehicle and Number of Driving license, respectively. Furthermore, let **ID card** = <NID, N, BD, Na>, **Vehicle registration card** = <NV, NC, NM, T, C>, **Driving license** = <NDL, N, BD, C>, **Personal Document** = {**ID card, Vehicle registration card, Driving license**}.

In fact, **Personal Document** may be considered as a structured data: **Personal Document** = [**Atomic data, ID card, Vehicle registration card, Driving license,** {**ID card, Vehicle registration card, Driving license**}].

The tree graph of **Personal Document** is t(**Personal Document**):

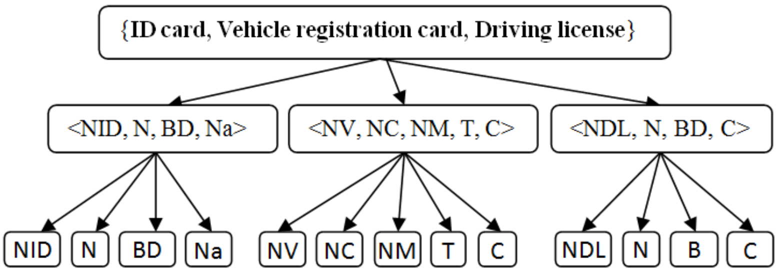


Figure 2
The tree graph of **Personal Document**

3.1 Operations of Structured Data

Let $\alpha = [r_0, r_1, \dots, r_m]$, $\beta = [s_0, s_1, \dots, s_n]$ be structured data over V. Then:

1. **Union:** The union of α, β is

   $\{\alpha, \beta\} = [r_0 \cup s_0, r_1, \dots, r_m, s_1, \dots, s_n, \{r_m, s_n\}]$

   The union of two structured data is also a structured data.

2. **Queuing :** The queuing of α, β is

   $\langle \alpha, \beta \rangle = [r_0 \cup s_0, r_1, \dots, r_m, s_1, \dots, s_n, \langle r_m, s_n \rangle]$

   The queuing of two structured data is also a structured data.

3. **Projection:** If $U \subseteq V$, then the projection of α on U is

   $Pr_U(\alpha) = [Pr_U(r_0), Pr_U(r_1), \dots, Pr_U(r_m)]$

   The projection of a structured data is also a structured data.

4. **Conjunction:** Let $\alpha_i = [r_0^i, r_1^i, \dots, r_{m_i}^i]$ be structured data over V for all $i = 1, \dots, k$, $r_0 = \bigcup_i r_0^i$. If r is a finite set of queues on $r_0 \cup \{r_j^i | j = 1, \dots, m_i, i = 1, \dots, k\}$, i.e. $r \in \mathcal{R}(r_0 \cup \{r_j^i | j = 1, \dots, m_i, i = 1, \dots, k\})$, then the *conjunction* of α_i's by r is

   $$r(\alpha_1, \alpha_2, \dots, \alpha_k) = [r_0, r_1^1, r_2^1, \dots, r_{m_1}^1, r_1^2, r_2^2, \dots, r_{m_2}^2, \dots, r_1^k, r_2^k, \dots, r_{m_k}^k, r]$$

   One can see that the conjunction of structured data is also a structured data. The union and the queuing operations are special cases of the conjunction.

3.2 Homomorphism, Isomorphism between Structured Data

Let $V,\ W$ be two sets of atomic data, $\varphi: V \to W$ and $r \in \mathcal{R}^{\mathrm{m}}(V)$. Then the homomorphic image of r is defined as follows:

Definition 11: Let $V,\ W$ be two sets of atomic data, $\varphi: V \to W$ and $r \in \mathcal{R}^{\mathrm{m}}(V)$.

i. If $r \in \mathcal{R}^1(\mathcal{A}) = \mathcal{R}(V)$, i.e. r is finite set of queues from V, $r = \{r_1, \dots, r_k\}$, where $r_i = \langle r_1^i, r_2^i, \dots, r_{m_i}^i \rangle$, $r_{\mathrm{j}}^{\mathrm{i}} \in V$, then

   $$\varphi(r) = \{\langle \varphi(r_1^1), \varphi(r_2^1), \dots, \varphi(r_{m_1}^1) \rangle, , \dots, \langle \varphi(r_1^k), \varphi(r_2^k), \dots, \varphi(r_{m_k}^k) \rangle\}$$

ii. Suppose that $\varphi(s)$ has been defined for all $s \in \mathcal{R}^{\mathrm{m}}(V)$. If $r \in \mathcal{R}^{\mathrm{m+1}}(V) = \mathcal{R}(\mathcal{R}^{\mathrm{m}}(V)), r = \{r_1, \dots, r_k\}$, where $r_i = \langle r_1^i, r_2^i, \dots, r_{m_i}^i \rangle$, $r_{\mathrm{j}}^{\mathrm{i}} \in \mathcal{R}^{\mathrm{m}}(V)$, then

   $$\varphi(r) = \{\langle \varphi(r_1^1), \varphi(r_2^1), \dots, \varphi(r_{m_1}^1) \rangle, , \dots, \langle \varphi(r_1^k), \varphi(r_2^k), \dots, \varphi(r_{m_k}^k) \rangle\}$$

   $\varphi(r)$ is the homomorphic image of r through φ.

Remark: Let $\varphi: V \to W$ and let $\alpha = [r_0, r_1, \dots, r_m] \in S_V$ be a structured data over V. It can be verified that $\varphi(\alpha) = [\varphi(r_0), \varphi(r_1), \dots, \varphi(r_m)]$ is also a structured data over W:

i. $\varphi(r_0) = \{\varphi(a) | a \in V\}$ is a finite set over W.

ii. For $i = 1, 2, \dots, m$ we have $r_{\mathrm{i}} \in \mathcal{R}(r_0 \cup \{r_1, r_2, \dots, r_{i-1}\})$, i.e. $r_i = \{u_1, u_2, \dots, u_k\}$, where $u_j = \langle u_1^j, u_2^j, \dots, u_{m_j}^j \rangle$, $u_{\mathrm{t}}^{\mathrm{j}} \in r_0 \cup \{r_1, r_2, \dots, r_{i-1}\}$. By Definition 11 $\varphi(r_i) = \{\varphi(u_1), \varphi(u_2), \dots, \varphi(u_k)\}$, where $\varphi(u_j) = \langle \varphi(u_1^j), \varphi(u_2^j), \dots, \varphi(u_{m_j}^j) \rangle$.

Since $\varphi(u_t^j) \in \varphi(r_0) \cup \{\varphi(r_1), \varphi(r_2), \dots, \varphi(r_{i-1})\}$, we have:

$\varphi(r_i) \in \mathcal{R}(\varphi(r_0) \cup \{\varphi(r_1), \varphi(r_2), \dots, \varphi(r_{i-1})\})$, i.e. $[\varphi(r_0), \varphi(r_1), \dots, \varphi(r_m)]$ is a structured data over W

Definition 12:

Let $\varphi: V \to W$ and $\alpha = [r_0, r_1, \dots, r_m] \in S_V$.

1. The *homomorphic image* of α by φ is $\varphi(\alpha) = [\varphi(r_0), \varphi(r_1), \dots, \varphi(r_m)]$.
2. If φ is bijective then $\varphi(\alpha)$ is the *isomorphic image* of α by φ.

By the previous remark we can see that the homomorphic image of a structured data is also a structured data. In other words, $\varphi: V \to W$ can be extended into $\varphi: S_V \to S_W$. We have:

Theorem 5: Let $\varphi: V \to W$. Then

1. For all $r \in \mathcal{R}^{\mathrm{m}}(V)$ we have $\varphi(l(r)) = l(\varphi(r))$
2. For all $\alpha \in S_V$ we have $\varphi(l(\alpha)) = l(\varphi(\alpha))$

Example 4:

Let $\varphi: V \to W$, where $V = \{x_1, x_2, x_3, x_4\}$, $W = \{a, b, c, d\}$ and $\varphi(x_1) = a$, $\varphi(x_2) = b$, $\varphi(x_3) = \varphi(x_4) = c$. Then

Table 2

Homomorphic image of a structured data

$\alpha = [r_0, r_1, r_2, r_3, r_4, r_5]$	$\varphi(\alpha) = [s_0, s_1, s_2, s_3, s_4, s_5]$
$r_0 = \{x_1, x_2, x_3, x_4\}$	$s_0 = \varphi(r_0) = \{a, b, c\}$
$r_1 = \langle x_1, x_2\rangle$	$s_1 = \varphi(r_1) = \langle a, b\rangle$
$r_2 = \langle x_2, x_3\rangle$	$s_2 = \varphi(r_2) = \langle b, c\rangle$
$r_3 = \{x_1, \quad r_1, \quad r_2\}$	$s_3 = \varphi(r_3) = \{a, s_1, s_2\}$
$r_4 = \{x_2, \langle x_4, r_3\rangle\}$	$s_4 = \varphi(r_4) = \{b, \langle c, s_3\rangle\} = \{b, \langle c, \{a, \langle a, b\rangle, \langle b, c\rangle\}\rangle\}$
$r_5 = \{r_4, \langle r_1, r_3, r_4\rangle\}$	$s_5 = \varphi(r_5) = \{s_4, \langle\langle a, b\rangle, s_3, s_4\rangle\}$

Remark: There exists $\varphi: V \to W, r, s \in \mathcal{R}^{\mathrm{m}}(V)$ such that:

1. $\varphi(Pr_U(s)) \neq Pr_{\varphi(U)}(\varphi(s))$
2. $r \lhd s$, but $\varphi(r) \ntriangleleft \varphi(s)$

Let $V = \{x, y, u\}$, $U = \{y, u\}$, $\varphi(x) = \varphi(y) = a$, $\varphi(u) = b$. Then $\varphi(U) = \{a, b\}$. For $r = \{u, \langle y, u\rangle\}$, $s = \{x, \langle x, u\rangle, \langle y, u\rangle\}$, we see $r = Pr_U(s)$, therefore $r \lhd s$. Moreover, $\varphi(r) = \varphi(Pr_U(s)) = \{b, \langle a, b\rangle\}$ and $\varphi(s) = Pr_{\varphi(U)}(\varphi(s)) = \{a, \langle a, b\rangle\}$. One can see $\varphi(Pr_U(s)) \neq Pr_{\varphi(U)}(\varphi(s))$ and $\varphi(r) \ntriangleleft \varphi(s)$.

3.3 Sub-Data

Below we define sub-data of the given structured data. In a sense, sub-data are not simply parts of structured data, but inherit the given structure.

Definition 13:

For two structured data $\alpha = [r_0, r_1, \dots, r_m]$, $\beta = [s_0, s_1, \dots, s_n]$ over V we say that α is a sub-data of β if:

i. $r_0 \subseteq s_0$, and

ii. $\forall i \geq 1\ \exists j \geq 1 : r_i = s_j$

Then we write $\alpha \sqsubseteq \beta$. The set of all sub-data of β is denoted by $SUBD(\beta)$.

Example 5:

Let $\alpha = [r_0, r_1, r_2, r_3], \beta = [s_0, s_1, s_2, s_3, s_4]$ where $s_0 = \{x_1, x_2, x_3, x_4\}$, $s_1 = \langle x_2, x_4 \rangle$, $s_2 = \{x_1, s_1\} = \{x_1, \langle x_2, x_4 \rangle\}$, $s_3 = \langle x_3, x_4 \rangle$, $s_4 = \{s_2, s_3\} = \{\{x_1, \langle x_2, x_4 \rangle\}, \langle x_3, x_4 \rangle\}$ and $r_0 = \{x_1, x_2, x_4\}$, $r_1 = s_1 = \langle x_2, x_4 \rangle$, $r_2 = s_2 = \{x_1, \langle x_2, x_4 \rangle\}$. One can see that $\alpha \sqsubseteq \beta$.

It is evident that the relation $\sqsubseteq$ defined in Definition 13 is a partial order on the set of structured data.

3.4 Components of Structured Data

Not all sub-data of structured data are its components. The components of structured data are all those sub-data that are maximal in a sense.

Definition 14:

1. Let $\alpha, \beta \in S_V$ be structured data. We say that α is a *component* of β if

   i. $\alpha \sqsubseteq \beta$,

   ii. There is no structured data $\gamma \in S_V$ such that $\alpha \sqsubseteq \gamma, \gamma \sqsubseteq \beta$, and $\gamma \neq \alpha, \gamma \neq \beta$.

The set of all components of a structured data β is denoted by $COMP(\beta)$.

2. Let $\alpha_1, \alpha_2, \dots, \alpha_n \in COMP(\beta)$. We say that $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ is an *adequate set of components* of β if $\beta = \alpha_1 \cup \alpha_1 \cup \dots \cup \alpha_n$.

3. We say that $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ is a *minimal adequate set of components* (MASC) of β if:

   i. $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ is an adequate set of components of β,

   ii. There is no real subset of $\{\alpha_1, \alpha_1, \dots, \alpha_n\}$ that is also adequate set of components of β.

In other words, a component of a structured data is some it's sub-data that is maximal in the partial order defined by $\sqsubseteq$.

Example 6:

Let $\beta = [\{a, b, c\}, \{b, c\}, \langle a, \{b, c\} \rangle, \{a, c\}, \langle b, \{c, a\} \rangle, \{\langle a, \{b, c\} \rangle, \langle b, \{c, a\} \rangle\}]$ and $\alpha = [\{a, b, c\}, \{b, c\}, \langle a, \{b, c\} \rangle]$. We can see that α is a component of β.

The following theorems are evident:

Theorem 6:

1. Let $\alpha, \beta \in S_V$ be structured data and $\alpha \sqsubseteq \beta$. Then there is $\gamma \in S_V$ such that $\alpha \sqsubseteq \gamma$ and γ is a component of β.
2. Every structured data has at least one component.
3. Every structured data has at least one MASC.

Theorem 7:

Let $\alpha \in S_X$ be structured data and $\varphi: \mathcal{A} \to \mathcal{B}$. If a set of structured data $\{\beta_1, \beta_1, \dots, \beta_n\}$ is a MASC of α, then $\{\varphi(\beta_1), \varphi(\beta_1), \dots, \varphi(\beta_n)\}$ is a MASC of $\varphi(\alpha)$.

3.5 Queries on Structured Data

Queries are operations that for a given set of data produce a set of data. In general, a simple query retrieves from a structured data some its sub-data. In this sense selections and projections in relational databases are such simple queries. Joins are queries, but are not simple queries.

Definition 15: Let $\mathcal{D} \subseteq S_V$ be a set of structured data over V.

1. A *query* over $\mathcal{D}$ is a mapping $q: \mathcal{D} \to \mathcal{D}$.
2. A query $q: \mathcal{D} \to \mathcal{D}$ is *proper* for $\alpha \in S_V$ if $q(\alpha) \sqsubseteq \alpha$.

   A query q is proper for $S \subseteq S_V$ if it is proper for all $\alpha \in S$.
3. Let Q be a set of queries over $\mathcal{D}$ and $\alpha \in S_V$ be a structured data over V. Then α is *minimal applicable data* for Q if:
   1. α is applicable for all query $q \in Q$.
   2. There is no $\beta \in S_V$ such that $\beta \sqsubseteq \alpha$ and β is applicable for all queries $q \in Q$.

3.6 Dependency Types and Keys

In this section we propose a concept of dependency types and the dependencies between the sub-data and components defined accordingly to the given dependency types are studied. The idea is simple: structured data and their sub-data, as well as their components are associated to the elements of a "sample set" where the "sample dependencies" have been well defined. Thus, the "sample dependencies" in the "sample set" induce, on the set of structured data, sample-like dependencies. Our study is focused on the dependency types defined by the lattices with partial order. We show here that functional dependencies in relational databases are, in fact, partial order type (or lattice-type) dependencies. This approach reveals that most of properties of functional dependencies are inherited from the properties of the partial order on the "sample" lattice.

Let $\mathcal{L}$ be a set with the partial order $\preccurlyeq$, then, as usual, for $L \subseteq \mathcal{L}$ we denote $Sup(L) = \{l \in \mathcal{L} |\ \forall l' \in L: l' \preccurlyeq l\}$ and $Sup^*(L) = \{l \in Sup(L) |\ \forall l' \in \mathcal{L}: l' \preccurlyeq l \Rightarrow l' \notin Sup(L)\}$. Similarly, we denote $Inf(L) = \{l \in \mathcal{L} |\ \forall l' \in L: l \preccurlyeq l'\}$ and $Inf^*(L) = \{l \in Inf(L) |\ \forall l' \in \mathcal{L}: l \preccurlyeq l' \Rightarrow l' \notin Inf(L)\}$.

Definition 16:

1. Let $\mathcal{R} \subseteq \mathcal{R}^\infty(V)$ be a set of relations over V. By *dependency type* on $\mathcal{R}$ we understand a couple $(\mathcal{L}, \varphi)$ where $\mathcal{L}$ is a lattice with the partial order $\preccurlyeq$, $\varphi: \mathcal{R} \to \mathcal{L}$ is a mapping that satisfies the following conditions:
   i. For $r, s \in \mathcal{R}$ if $\varphi(r)$ are determined and $s \leq r$ then $\varphi(s)$ is determined and $\varphi(s) \preccurlyeq \varphi(r)$
   ii. For $r, s \in \mathcal{R}$, if $s = \langle s_1, s_2, \dots, s_n \rangle, r = \langle r_1, r_2, \dots, r_n \rangle$; $\varphi(s_i), \varphi(r_i)$ are determined and $\varphi(s_i) \preccurlyeq \varphi(r_i)$ for all $i = 1,2,\dots,n$, then $\varphi(s)$, $\varphi(r)$ are determined and $\varphi(s) \preccurlyeq \varphi(r)$.
   iii. For $r, s \in \mathcal{R}$, if $s = \{s_1, s_2, \dots, s_n\}, r = \{r_1, r_2, \dots, r_m\}$; $\varphi(s_i)$, $\varphi(r_j)$ are determined and for all $i = 1,2, \dots, n$ there exists $j = 1,2,\dots,m$ such that $\varphi(s_i) \preccurlyeq \varphi(r_j)$, then $\varphi(s)$, $\varphi(r)$ are determined and $\varphi(s) \preccurlyeq \varphi(r)$
2. For two relations $r, s \in \mathcal{R}$ we say that s *depends on* r in dependency type $(\mathcal{L}, \varphi)$ if $\varphi(s) \preccurlyeq \varphi(r)$. Then we write $r \underset{\mathcal{L},\varphi}{\longrightarrow} s$ or for simplicity $r \to s$ if $\mathcal{L}, \varphi$ are well known.
3. For two structured data $\alpha, \beta \in S_V$, $\alpha = [r_1, r_2, \dots, r_m]$, $\beta = [s_1, s_2, \dots, s_n]$, we say that β *depends on* α in dependency type $(\mathcal{L}, \varphi)$ if $\{r_1, r_2, \dots, r_m\} \to s_i$ for all $i = 1,2,\dots,n$. Then we write $\alpha \underset{\mathcal{L},\varphi}{\longrightarrow} \beta$ or for simplicity $\alpha \to \beta$ if $\mathcal{L}, \varphi$ are well known.

The dependencies defined in the Definition 16 are called partial order dependencies or lattice-like dependencies.

Theorem 8: Let $(\mathcal{L}, \varphi)$ be a dependency type, where $\mathcal{L}$ is a set with the partial order $\preccurlyeq$, $\varphi: \mathcal{R} \to \mathcal{L}$.

1. If $s = \langle s_1, s_2, \dots, s_n \rangle$, $\varphi(s_i)$ is determined for all $i = 1,2,\dots,n$, then $\varphi(s)$ is determined and

   $\varphi(s) \in Sup^*(\varphi(s_1), \varphi(s_2), \dots, \varphi(s_n))$
2. If $s = \{s_1, s_2, \dots, s_n\}$, $\varphi(s_i)$ is determined for all $i = 1,2,\dots,n$, then $\varphi(s)$ is determined and

   $\varphi(s) \in Sup^*(\varphi(s_1), \varphi(s_2), \dots, \varphi(s_n))$

Proof:

If $s = \langle s_1, s_2, \dots, s_n \rangle$ or $s = \{s_1, s_2, \dots, s_n\}$, $\varphi(s_i)$ is determined for all $i = 1,2,\dots,n$, then $\varphi(s)$ is determined. Moreover, $s_i \leq s$, therefore $\varphi(s_i) \preccurlyeq \varphi(s)$ for all $i = 1,2,\dots,n$, i.e. $\varphi(s) \in Sup(\varphi(s_1), \varphi(s_2), \dots, \varphi(s_n))$.

Moreover, if $\varphi(s') \in Sup(\varphi(s_1), \varphi(s_2), \dots, \varphi(s_n))$, then $\varphi(s_i) \preccurlyeq \varphi(s')$, for all $i = 1,2,\dots,n$.

1. By ii. in Definition 16, if $s = \langle s_1, s_2, \dots, s_n \rangle$ then by $s' = \langle s', s', \dots, s' \rangle$, we have $\varphi(s) \preccurlyeq \varphi(s')$. This proves that $\varphi(s) \in Sup^*(\varphi(s_1), \varphi(s_2), \dots, \varphi(s_n))$.
2. By ii. in Definition 16, if $s = \{s_1, s_2, \dots, s_n\}$ then by $s' = \{s'\}$, we have $\varphi(s) \preccurlyeq \varphi(s')$. This proves that $\varphi(s) \in Sup^*(\varphi(s_1), \varphi(s_2), \dots, \varphi(s_n))$

Theorem 9: The functional dependencies in relations of relational database are partial order dependencies.

Proof: In the Example 1 we have shown that every relation $r = \{A_1, A_2, \dots, A_m\}$ in relational database with the columns $A_i = (a_{ij})$, $i = 1,2,\dots,m$, $j = 1,2,\dots,n$ can be considered as a 2-order relation:

$$r = \{\langle a_{11}, a_{12}, \dots, a_{1n} \rangle, \langle a_{21}, a_{22}, \dots, a_{2n} \rangle, \dots, \langle a_{m1}, a_{m2}, \dots, a_{mn} \rangle\}$$

In fact, r can be considered as a structured data

$$r = [r_0, r_1, r_2, \dots, r_m, r_{m+1}]$$

where $r_0 = \{a_{ij} | i = 1,2,\dots,m, j = 1,2,\dots,n\}$, $r_i = \langle a_{i1}, a_{i2}, \dots, a_{in} \rangle$ for $i = 1,2,\dots,m$ and $r_{m+1} = \{r_1, r_2, \dots, r_m\}$.

The functional dependency between the columns of r can be defined as follows: Let $\mathcal{L}$ the set of all partitions on the set $\{l_1, l_2, \dots, l_n\}$, where $l_1, l_2, \dots, l_n$ are the rows of r. We denote by $\preccurlyeq$ the following partial order on $\mathcal{L}$: for two partitions p, q on $l_1, l_2, \dots, l_n$ we write $p \preccurlyeq q$ if

$$l_j \sim_q l_k \Rightarrow l_j \sim_p l_k$$

To each a_{ij} we associate the partition $\varphi(a_{ij})$ over $\{l_1, l_2, \dots, l_n\}$ defined by the following equivalence: $l_k \sim_{a_{ij}} l_t$ if and only if both l_k, l_t contain a_{ij} or both l_k, l_t do not contain a_{ij}.

To the column A_i we associate the partition $\varphi(A_i)$ over $\{l_1, l_2, \dots, l_n\}$ defined by the following equivalence: $l_k \sim_{A_i} l_t$ if and only if $a_{ik} = a_{it}$.

To a set of columns $\mathcal{A}$ we associate the partition $\varphi(\mathcal{A})$ over $\{l_1, l_2, \dots, l_n\}$ defined by the following equivalence: $l_j \sim_{\mathcal{A}} l_k$ if and only if $l_j \sim_{A_i} l_k$ for all $A_i \in \mathcal{A}$.

One can verify that φ satisfies the conditions in Definition 16, thus $(\mathcal{L}, \preccurlyeq)$ is partial order dependency type. It is easy to see that for two set of columns $\mathcal{A}, \mathcal{B}$ in r, $\mathcal{B}$ functionally depends on $\mathcal{A}$, i.e. $\mathcal{A} \to \mathcal{B}$, if and only if $\varphi(\mathcal{B}) \preccurlyeq \varphi(\mathcal{A})$.

One can verify that most of rules that hold for functional dependency in fact hold also for generalized model, i.e. for partial order dependency. We have:

Theorem 10: Let $(\mathcal{L}, \varphi)$ be a dependency type on $\mathcal{R} \subseteq \mathcal{R}^{\infty}(V)$. Let α, β, γ be relations (or structured data) over V. We have:

1. (Reflexivity) If $\beta \leq \alpha$, then $\alpha \to \beta$.
2. (Augmentation) If $\alpha \to \beta$, then $\alpha \cup \gamma \to \beta \cup \gamma$ for any γ.

   In the case α, β, γ are relations by $\alpha \cup \gamma$, $\beta \cup \gamma$ we understand $\{\alpha, \gamma\}$ and $\{\beta, \gamma\}$, respectively.
3. (Transitivity) If $\alpha \to \beta$, $\beta \to \gamma$, then $\alpha \to \gamma$.

Definition 17: For a structured data $\alpha \in S_V$, $\alpha = [r_1, r_2, \dots, r_m]$, we say that a set $\beta = \{r_{i_1}, r_{i_2}, \dots, r_{i_k}\}$ is a *key set* of α in dependency type $(\mathcal{L}, \varphi)$ if $\{r_{i_1}, r_{i_2}, \dots, r_{i_k}\} \underset{\mathcal{L},\varphi}{\longrightarrow} r_i$ for all $i = 1,2, \dots, m$.

The following example shows how a relation in relational database can be considered as structured data and how the functional dependencies in it can be studied as partial order dependencies.

Example 7: Let $\boldsymbol{r}$ be the relation in the Figure x, where $\boldsymbol{l_i}$s and $\boldsymbol{c_2}$ are the rows and the columns of $\boldsymbol{r}$, respectively.

Table 3

Relation $\boldsymbol{r}$ in relational database

	$\boldsymbol{c_1}$	$\boldsymbol{c_2}$	$\boldsymbol{c_3}$	$\boldsymbol{c_4}$
$\boldsymbol{l_1}$	x_1	y_1	z_1	w_1
$\boldsymbol{l_2}$	x_1	y_2	z_2	w_1
$\boldsymbol{l_3}$	x_2	y_2	z_3	w_1

$\boldsymbol{r}$ can be considered as a structured data $[\boldsymbol{r_0}, \boldsymbol{c_1}, \boldsymbol{c_2}, \boldsymbol{c_3}, \boldsymbol{c_4}, \boldsymbol{r}]$ where $\boldsymbol{r_0} = \{x_1, x_2, y_1, y_2, z_1, z_2, z_3, w_1\}$ is the set of all atomic data, $\boldsymbol{c_1} = \langle x_1, x_1, x_2 \rangle$, $\boldsymbol{c_2} = \langle y_1, y_2, y_2 \rangle$, $\boldsymbol{c_3} = \langle z_1, z_2, z_3 \rangle$, $\boldsymbol{c_4} = \langle w_1, w_1, w_1 \rangle$ are the columns of the relation $\boldsymbol{r} = \{\boldsymbol{c_1}, \boldsymbol{c_2}, \boldsymbol{c_3}, \boldsymbol{c_4}\}$.

Denote the rows of $\boldsymbol{r}$ by $\boldsymbol{l_1}, \boldsymbol{l_2}, \boldsymbol{l_3}$**.** The set of all partitions on $\{\boldsymbol{l_1}, \boldsymbol{l_2}, \boldsymbol{l_3}\}$ is denoted by $\mathcal{L}$. Every partition $p \in \mathcal{L}$ can be determined by the equivalence $\sim_p$: $\boldsymbol{l_i}, \boldsymbol{l_j}$ belong to a same class in p if and only if $\boldsymbol{l_i} \sim_p \boldsymbol{l_j}$. $\mathcal{L}$ is a lattice where the partial order on $\mathcal{L}$ is defined as usual: $p \preccurlyeq q$ if and only if $\boldsymbol{l_i} \sim_q \boldsymbol{l_j} \Rightarrow \boldsymbol{l_i} \sim_p \boldsymbol{l_j}$. The partial order on $\mathcal{L}$ is described in the following diagram:

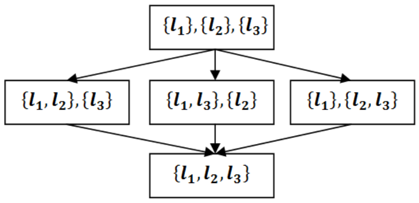


Figure 3
The partial order between the partitions

Each relation $\boldsymbol{x}$ in the structured data $\boldsymbol{r}$, $\boldsymbol{x} \in \{\boldsymbol{r_0}, \boldsymbol{c_1}, \boldsymbol{c_2}, \boldsymbol{c_3}, \boldsymbol{c_4}, \boldsymbol{r}\}$, accordingly to Theorem 9, can be associated to a partition $\varphi(\boldsymbol{x}) \in \mathcal{L}$ as follows:

Table 4
Relations in a structured data and their image in a partially ordered set

$\boldsymbol{x}$	$\varphi(\boldsymbol{x})$	$\boldsymbol{x}$	$\varphi(\boldsymbol{x})$	$\boldsymbol{x}$	$\varphi(\boldsymbol{x})$	$\boldsymbol{x}$	$\varphi(\boldsymbol{x})$
x_1	$\{l_1,l_2\},\{l_3\}$	y_2	$\{l_1\},\{l_2,l_3\}$	z_3	$\{l_1,l_2\},\{l_3\}$	$\boldsymbol{c_2}$	$\{l_1\},\{l_2,l_3\}$
x_2	$\{l_1,l_2\},\{l_3\}$	z_1	$\{l_1\},\{l_2,l_3\}$	w_1	$\{l_1,l_2,l_3\}$	$\boldsymbol{c_3}$	$\{l_1\},\{l_2\},\{l_3\}$
y_1	$\{l_1\},\{l_2,l_3\}$	z_2	$\{l_1,l_3\},\{l_2\}$	$\boldsymbol{c_1}$	$\{l_1,l_2\},\{l_3\}$	$\boldsymbol{c_4}$	$\{l_1,l_2,l_3\}$
						$\boldsymbol{r}$	$\{l_1\},\{l_2\},\{l_3\}$

Thus one can see the dependencies between the relations in the structured data $\boldsymbol{r}$:

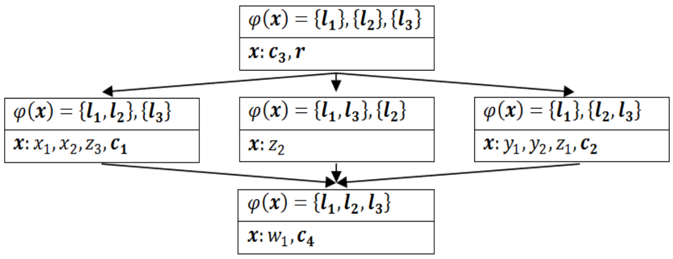

$\varphi(\boldsymbol{x}) = \{l_1\},\{l_2\},\{l_3\}$
$\boldsymbol{x}: \boldsymbol{c_3}, \boldsymbol{r}$
$\varphi(\boldsymbol{x}) = \{l_1,l_2\},\{l_3\}$
$\boldsymbol{x}: x_1, x_2, z_3, \boldsymbol{c_1}$
$\varphi(\boldsymbol{x}) = \{l_1,l_3\},\{l_2\}$
$\boldsymbol{x}: z_2$
$\varphi(\boldsymbol{x}) = \{l_1\},\{l_2,l_3\}$
$\boldsymbol{x}: y_1, y_2, z_1, \boldsymbol{c_2}$
$\varphi(\boldsymbol{x}) = \{l_1,l_2,l_3\}$
$\boldsymbol{x}: w_1, \boldsymbol{c_4}$

Figure 4
The dependencies between the relations in a structured data

Conclusions

In this paper we have proposed a formalization for structured data, in which data are constructed recursively by two basic structures, namely, by sets and queues, based upon atomic data. Although the approach may not deal with all structured data, it does touch on a large portion. The relations, relational databases can be handled in this formalization. We show that many well-known concepts and results in relational databases, such as keys and functional dependencies, can be studied in this generalized model of data. The generalization has certain advantages: the concepts and results in relational databases are quite clear in this formalization, the properties of keys and functional dependencies are inherited from the sample hierarchy in a lattice, etc. Moreover, the proposed approach also offers a unique method for managing different operations on structured data.

As one can see, the approach poses several problems that may be interesting topics for further studies. These problems are:

- The operations on structured data should be studied more thoroughly, including the composition and decomposition of structured data.
- The relational algebra should be developed for structured data.
- An optimization and normalization of structured data should be studied that guarantees the optimality and consistency of structured data management systems.

References

[1] Békéssy, J. Demetrovics: Contribution to the Theory of Data Base Relations, Discrete Math., 27, 1979, pp. 1-10

[2] C. Beeri, M. Dowd, R. Fagin, R. Statman: On the Structure of Armstrong Relations for Functional Dependencies, J. ACM, 31, 1984, pp. 30-46

[3] Brigitte Le Roux, Henry Rouanet: Geometric Data Analysis: From Correspondence Analysis to Structured Data Analysis, Kluwer Academic Publishers, ISBN 1402022352, 2004

[4] Codd, E. F. (1970). "A Relational Model of Data for Large Shared Data Banks". *Communications of the ACM* **13** (6): 377. doi:10.1145/362384.362685

[5] Dana Scott: Data Types as Lattices, SIAM Journal on Computing 1976, Vol. 5, No. 3, pp. 522-587, ISSN: 0097-5397

[6] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber Bigtable: A Distributed Storage System for Structured Data, J. ACM Transactions on Computer Systems (TOCS), Volume 26, Issue 2, June 2008, Article No. 4, ACM New York, NY, USA

[7] J. Demetrovics: Candidate Keys and Antichains, SIAM J. Algebraic Discrete Math., 1 (1980), p. 92

[8] János Demetrovics, Leonid Libkin, Ilya B. Muchnik: Functional Dependencies in Relational Databases: A Lattice Point of View, Discrete Applied Mathematics,Volume 40, Issue 2, 10 December 1992, pp. 155-185

[9] Karthik Kambatla, Giorgios Kollias, Vipin Kumar, Ananth Grama: Trends in Big Data analytics. J. Parallel Distrib. Comput. 74(7), 2014, pp. 2561-2573

[10] Paredaens, J., De Bra, P., Gyssens, M., Van Gucht, D.: The Structure of the Relational Database Model, EATCS Monographs on Computer Science, nr. 17, Springer Verlag, ISBN 3540137149, 1989

Performance Analysis of a Cluster Management System with Stress Cases

Gergő Gombos, Attila Kiss, Zoltán Zvara

Eötvös Loránd University, Faculty of Informatics
Pázmány Péter u. 1/c, H-1117 Budapest, Hungary
ggombos@inf.elte.hu, kiss@inf.elte.hu, dyin@inf.elte.hu

Abstract: Cluster computing frameworks are important in the "Big Data" world. The famous common framework is the MapReduce that was introduced by Google. This framework is used by many of companies. However, this technique doesn't effectively solve all analytical problems. Some cases need another framework and these frameworks can work in the cluster. In this case, the cluster needs a manager that manages the framework. Therefore, the performance analysis of cluster management systems will be important. In this paper, we compare the performance of two most well-known cluster management systems (Yarn, Mesos) with stress cases. We analyze the resource usage techniques of the management systems.

Keywords: cluster management; resource sharing; scheduling

1 Introduction

For years, Big Data was confined to a group of elite technicians working for companies like Google and Yahoo, but the databases and the tools used to manage the data at that scale have been constantly evolving. At that time, Big Data was only a synonym to the leading tool, the Apache Hadoop [1], a MapReduce [2] implementation that was used as a data-processing platform for many years, exclusively. As Big Data continued to evolve, researchers found that MapReduce – though is still powerful for a large number of applications – was not as effective at solving many problems. Technicians were working on new cluster computing frameworks, and it became clear that no framework would be optimal for all applications. Researchers have been developing a wide array of data-centric computing frameworks and the need for a major computing platform emerged, powering both the growing number of data-intensive scientific applications and large internet services. It has become essential to run multiple frameworks on the same cluster, so data scientists can pick the best for each application.

As new analytic engines began to cover the ever growing space of problems, sharing a cluster between these frameworks started to get complicated. At the enterprise level, along with the need of batch processing, the need of real-time event processing, human interactive SQL queries, machine learning and graphic analytics emerged.

In a cluster, data are distributed and stored on the same nodes that run computations shared by frameworks. When the cluster is shared, statically, by frameworks, unnecessary data replication will appear, along with utilization issues. When a framework, for example a web-service farm, would be able to scale down at late hours, the MapReduce framework would perform better if it were able to use the resources released by the web-service farm. Sharing improves cluster utilization, through statistical multiplexing and avoids per-framework data replication and leads to data consolidation.

A cluster management system acts as a cluster-wide operating system by sharing commodity clusters between multiple and diverse cluster computing frameworks. Because reading data remotely, is expensive on a distributed file system, it is necessary to schedule computations near their data. At each node, applications take turns running computations, executing long or short tasks, spawned by different frameworks. Locality in large clusters is crucial for performance, because network bisection bandwidth becomes a bottleneck. [2] A cluster management system should provide a tool or interface, to design and implement specialized, distributed frameworks targeted at special problem domains. While multiple frameworks are operating cluster-wide, the operating system should take care of difficult problems, like cluster health, fault monitoring, resource arbitration and isolation.

Energy efficiency also becomes a critical matter for data centers powering large numbers of clusters [6] [7], since energy costs are ever increasing and hardware costs are decreasing. Minimizing the total amount of resources consumed will directly reduce the total energy consumption of a job.

Scalability, resource- and energy-efficiency are key metrics for a cluster management system, their performance matters for data-center operators, as well as for end users. [3] [4] [5]

Driven by the need of a cluster-wide operating system to share data among frameworks, two solutions appeared from the ground of The Apache Software Foundation that circulated widely in the Big Data community, to provide a resource management substrate for analytic engines and their applications. One such solution was designed and presented at U.C. Berkeley, called Apache Mesos and another one, originated from the Hadoop architecture, named YARN (Yet Another Resource Negotiator).

In this work we will show and demonstrate the differences of these two, open-source cluster-wide operating systems, by presenting an infrastructure, resource management, scheduling overview and performance evaluations, on different scenarios together with load and stress testing. Both of these systems are used widely in production systems and by introducing different resource-management models, it is beneficial to analyze their performance. Using the performance evaluations we will demonstrate the advantages and disadvantages, of different configurations, use cases on both YARN and Mesos, with different analytical frameworks having diverse needs and routines on execution.

2 Design and Concepts

A cluster management system consists of two main components. A *master* entity, that manages resources, schedules framework's resource requirements and *slave* entities, which run on nodes to manage tasks and report to the master. These two components build up the *platform.* A *scheduler* is a singular or distributed component in the platform that schedules jobs (or applications) on the cluster expressed and written by end-users using a specific *framework* library. A cluster management system can be considered as a distributed operating system: it provides resources for frameworks and schedules their distributed applications.

Frameworks are more or less, independent entities, with their own scheduler and resource requirements, but there are dissimilarities among design philosophies on different systems. A live framework is expected to register itself with the cluster's master, by implementing a resource-negotiating API defined by the master. Apart from the global, cluster wide resource management, scheduling, other expectations, such as fault tolerance, job-level scheduling or logging are the framework's duty to provide.

The masters are made to be fault-tolerant on both Mesos and YARN by ZooKeeper [8]. In a cluster deployed with Mesos, a framework must be set up on a given node and it must register itself with the master to be able to negotiate for resources and run tasks on the nodes. YARN requires a *client* to submit the framework, as an *application* to the resource manager. The resource manager will eventually start the framework on a node, making it live, to be able to request resources and run tasks on the nodes.

2.1 Resource Management

As previously described, the master entity arbitrates all available cluster resources by working together with the per-node slaves and the frameworks or applications. The resource manager component of the master entity does not concern itself with framework or application state management. It schedules the overall resource profile for frameworks and it treats the cluster as a resource pool.

There are two methods for gathering resources from the cluster. Mesos *push*es, offers resources to frameworks, those implement a `Scheduler`, while applications, which implement the `YarnAppMasterListener` interface are expected to *pull*, request resources. Mesos offers resources to the `Scheduler` and it chooses to accept, or not, in contrast to the model used by YARN, where the `AppMaster` must request resources from the `ResourceManager` and it chooses to give resources or not.

Resource allocations in YARN are late binding, that is, the application or framework is obligated to use the resources provided by the container, but it does not have to apply them to a logical task on request. The framework or application can decide which task to run with its own, internal, second-level scheduler. In Mesos, task descriptions must be sent upon accepting a resource.

On Mesos and YARN the existing grammar of resource requests does not support specification of complex relationships between containers regarding co-location. Second-level schedulers must implement such relationships. Also, since Mesos offers resources to the framework it will hinder locality preferences, while YARN lets the framework request any node in the cluster, not only from a sub-cluster offered by the resource manager. To tailor and limit resource consumption of different frameworks, a pluggable allocation module in the master entity of Mesos can determine how many resources to offer each framework.

2.2 Scheduling

Given the limited resources in the cluster, when jobs cannot all be executed or resource requests cannot all be served, scheduling their executions becomes an important question, allocating resources to frameworks becomes crucial to the performance. A centerpiece of any cluster management system is the scheduler. Scheduler architecture design impacts elasticity, scalability and performance in many dimensions and data-localities within distributed operating systems.

2.2.1 Statically Partitioned

Statically partitioned schedulers lead to fragmentation and suboptimal utilization. It is not a viable architecture to achieve high throughput and performance, which is an elemental requirement amongst cluster management systems.

2.2.2 Monolithic

A monolithic scheduler uses a central algorithm for all jobs and it is not parallel, implements policies and specialized implementations, in one code base. In the high-performance computing world, this is a common approach, where each job must be scheduled by the same algorithm. The era of Hadoop on Demand (HoD), was a monolithic scheduler implementation. The problem with a monolithic architecture is that it puts too much strain on the scheduler from a certain cluster size and it becomes increasingly difficult to apply new policy goals, such as, failure-tolerance and scaling.

2.2.3 Two-Level

An approach used by many systems is to have a central scheduler, a coordinator that decides how many resources each sub-cluster will have. This two-level scheduling is used by Mesos, YARN, Corona [9] and HoD. An offer-based two-level scheduling mechanism provided by Mesos, works best when the tasks release resources frequently, meaning that job sizes are also small compared to the total available resources. Since the Mesos master does not have access to a global view of the cluster state, only the resources it has been offered, it cannot support preemption. This restricted visibility of cluster resources might lead to losing work when optimistic concurrency assumptions are not correct. Mesos uses resource hoarding to group (gang) schedule frameworks and this can lead to a deadlock in the system. Also, the parallelism introduced by two-level schedulers is limited, due to a pessimistic concurrency control.

YARN, is effectively, a monolithic architecture, since the application masters usually don't provide scheduling, but only job-management services, like the Spark [10] master entity. An `ApplicationMaster` can in fact implement a second level of scheduling and assign its containers to whichever task is part of its execution plan. The `MRAppMaster` is a great example of the dynamic two-level scheduler as it matches allocated containers against the set of pending map tasks by data locality.

2.2.4 Comparison

Design comparisons, simulations present the tradeoffs between the different scheduler architecture approaches [11]. Increasing the per-job scheduling overhead (the time needed to schedule a job) will increase the scheduler business in the monolithic, single-path baseline case, linearly. The job wait time will increase at a similar rate, until the scheduler is fully saturated. On a multi-path implementation, average job wait time and scheduler activity decreases, but batch jobs can still get stuck in a queue behind service jobs, which are slow to schedule.

Scheduling batch workloads will result in a busier scheduler when using a two-level (Mesos) architecture instead of a monolithic architecture, as a consequence of the interaction between the Mesos offer model and the second-level scheduler in the framework. Because Mesos achieves fairness by offering *all* available cluster resources to schedulers, a long second-level decision time means that nearly all the resources are locked too long a time, making them inaccessible to other schedulers. Mesos predicts by making quick scheduling decisions and having small jobs within a large resource pool, which can cause aforementioned mentioned problems in a different scenario.

4 Experimental Evaluations

In this section we will demonstrate the two cluster management systems in operation, regarding scheduling and execution performance in different scenarios using two popular frameworks, the Hadoop MapReduce implementation and Spark. We test single job execution concerning startup overhead and scheduling efficiency, throughput along with node performances.

These evaluations were run on 5 computers, each equipped with an Intel Core i5 CPU and 4GB RAM. One computer was set up as a dedicated master, resource manager for both YARN and Mesos, history server and proxy server for YARN, but also as a name node and secondary name node for HDFS. The other 4 nodes were set up as data nodes and slaves to run jobs. In the case of Mesos, the frameworks (for example Hadoop `JobTracker`, Hama `BSPMaster`) were deployed and activated on the master node.

In these experiments the following cluster and framework versions were used: Hadoop YARN version 2.5.2 [12], Mesos 0.21.0 [13], with the Hadoop on Mesos library version 0.0.8 [14] and Spark 1.3.0 [15]. We observed no measurable performance differences between MRv1 and MRv2, apart from the overhead originated from launching TaskTrackers.

In each cluster, a total 32 virtual CPUs and 32768 MB of virtual memory were available while running these tests. Both YARN and Mesos were only able to isolate CPU and memory as resources. Disk usage or network bandwidth were managed by the underlying operating system (Ubuntu 14.04). The tests ran 5 times and the results were aggregated to calculate averages. We considered the resource use as use of CPU and memory.

While each cluster management system provides a REST API to query for the system, framework, job, task data as well as metrics, the Mesos API seemed to be a more detailed and precise regarding system parameters. While each job runs as a separate application in YARN, from a cluster point of view, finer grained snapshot becomes visible. Mesos does not know and neither concerns of the job granularity, it really does not know how many map-reduce jobs were ran by the connected `JobTracker` framework entity.

The deployment of YARN with the different frameworks mentioned above worked more like an out-of-box application, compared to Mesos, where permission problems were met several times while running map-reduce jobs along with the frequent node failures in case of handling too much executors at once.

4.1 Single Batch Job Performance

To examine the performance on accepting, scheduling and preparing a certain task, we’ve ran a long, batch-like job on each platform, an IO and memory heavy map-reduce program on a 40 GB dataset that is stored in HDFS with a replication factor of 2. The cluster utilization was at 0% each time the program gets submitted. Since Mesos uses Hadoop MapReduce API and architecture of version 1 and YARN uses the next-generation version 2 API, differences are expected in map-reduce job execution schemes and performance.

While running the map-reduce job on Mesos, it completes in 1130 seconds with 374 maps and 1 reduce task and with total 358 data-local map tasks, that is 24 more maps ran, than in YARN. *Figure 1* shows that the first `TaskTrackers` of Hadoop version 1 to reach the staging status on an executor launched by Mesos required 8 seconds from the time the job submitted. Staging status refers to the state when the slot (container) is allocated and the setup of the executor has been started. To be able to set up `TaskTrackers`, the Hadoop v1 architecture needs to be distributed as a TAR throughout the slaves, stored on HDFS. After the executors get started, the Hadoop distribution gets downloaded from HDFS, so a working Hadoop must present on each slave with the capability to invoke the `hadoop` command and to communicate with the HDFS. If the Hadoop distribution in question, is not replicated at each node, the transfer time (of roughly 250 MB) heavily impacts the ramp-up time of the `TaskTrackers`.

In a scenario, where network bandwidth is a bottleneck, transferring Hadoop framework executors can keep many tasks on staging status for a long time. *Figure 1* shows that, with replication factor 4 it took 42 seconds to get the first few `TaskTrackers` to get to running state. Other mappers were also considered slow to start up. The reducer was created and launched in the 95th second, while map phase was at 7.6% completion. Reaching the maximum utilization, 1 virtual core and 1024 MB of memory was not used. Mesos set up a total of 11 `TaskTrackers` on the small cluster.

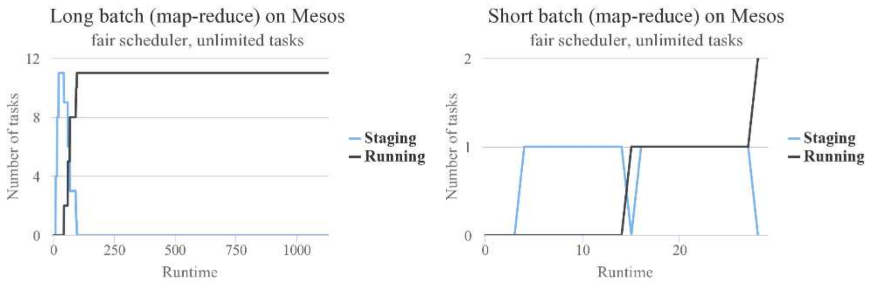


Figure 1
Number of staging and running tasks in case of long and short batch (map-reduce) jobs on Mesos

It became clear that `TaskTrackers`, executors are a huge deal to set up on first time and could mean a slow response from frameworks as elasticity is being harnessed. Burst-like jobs from different frameworks could mean too much work spent on setting up and launching executors.

A much smaller, micro map-reduce job on an 8 MB dataset was run for the same purpose. Mesos created two `TaskTrackers` on different nodes for 1 map and 1 reduce task, with 1 data-local map task. It took 29 seconds for the job to complete, while in contrast to the long batch job, the first task reached staging state on the 4th second and started to run on the 15 second mark. With smaller jobs, tasks get staged and run much faster.

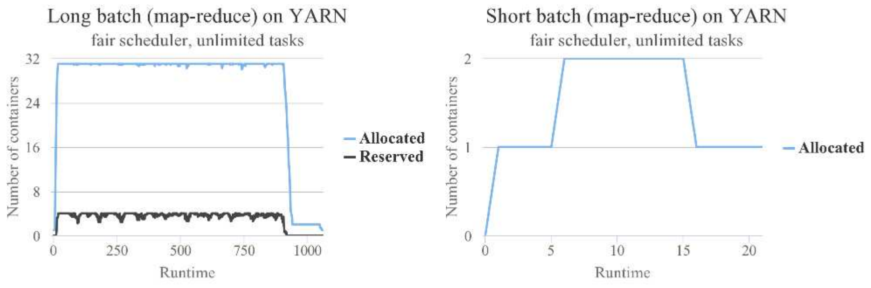


Figure 2

Number of containers allocated and reserved in case of long and short batch (map-reduce) job on YARN

YARN outperformed Mesos on both long and short term. As shown in *Figure 2*, YARN completed the short job in 21 seconds on average and the `MRAppMaster` was placed on container 0 in a second. Breaking with the Hadoop v1 design really pays off, as mappers and reducers are placed very fast on the designated nodes without the `TaskTrackers` to deploy. We found minimal differences in speed comparing the map-reduce implementation of Hadoop v1 and v2. As seen in *Figure 2*, YARN reserved containers to prevent starvation from smaller applications. This behavior appeared to be common on long tasks, but the `MRAppMaster` never reserved more than 4 containers, even on longer jobs, but one for each node.

YARN completed the long map-reduce job with 1 application master, 1 reduce and 360 map containers in total. The advantage of this granularity pays off, when new applications enter the scheduling phase and DRF wishes to free resources for them. Killing a few map tasks to be able to allocate cluster resources for new applications would not result in a major drawback for the map-reduce program running on YARN, since 360 mappers are reserved and used up linearly in the execution timeframe. On a long map-reduce job in case of Mesos, while `TaskTrackers` would allocate slots for a long time, an allocation module would kill a few of them, to place new frameworks' tasks on the cluster. Map-reduce is resilient to task failures, since work lost could be repeated, but on a long term this can hurt utilization and hinder completion time in a much greater aspect. Map-reduce on YARN, provides much better elasticity, along with, a faster execution.

Another issue to point out with Mesos, is that during the execution of the map-reduce long job it consumed relatively more resources on the execution timeframe than the MR implementation on YARN. As *Figure 2* shows after the 940th second, only one reducer was running on one container aggregating results from mappers, but in case of Mesos, until the very end of the job's last phase, all `TaskTrackers` were still running and the `JobTracker` freed them after the job was completed. As higher resource consumption directly affects money spent in a cloud environment, choosing Mesos might result in a higher bill than expected.

Running a micro Spark job on each system resulted in an average 4 second difference, in favor of Mesos. It worth mentioning, that the current Spark implementation does not support cluster deployment mode in for Mesos. Running a Spark job on YARN requires a Spark `ApplicationMaster` to be created on container 0, which impacts the startup time of the actual job. Spark jobs can run in client mode with YARN, but this setup did not yield a better result. The same job on Mesos was run by a Spark client on the master node, thus it was able to negotiate resources and launch containers without the time to deploy itself. Spark jobs on Mesos can run by using a predefined executor with the `spark.executor.uri` configuration parameter or by deploying packages manually to each node with the appropriate configuration.

It is evident, that the deploy-the-application approach introduced by YARN is more convenient from the client's point of view than the connect-the-framework concept. A client does not have to keep its instance of framework running and can disconnect from the cluster after the application got submitted. On the other hand, deploying a framework manually to a node could lead to uncontrolled resource consumption as the framework is not managed and isolated by the resource manager. Using Spark in client mode means that multiple Spark-framework instances will appear and act as tenants for the DRF scheduler, while one `JobTracker` runs multiple map-reduce programs. This approach will eventually make the tasks of the allocator module harder, when it tries to enforce organizational policies.

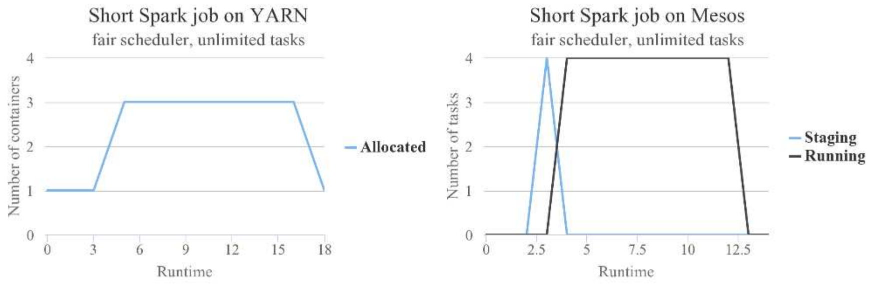


Figure 3

Number of containers allocated (YARN) and number of tasks staging and running (Mesos) in case of running the short Spark job

As seen in *Figure 3*, YARN completed the Spark job in 18 seconds using 3 containers (including the Spark master on container 0), while Mesos in 14 seconds using 4 containers. The running container 0, on YARN, required roughly 5 seconds to request, prepare and run the Spark executors, on the allocated containers. The executors were running for 11 seconds and the application master went offline after 2 seconds. The execution of Spark job on 4 containers resulted in a timeframe of 7 seconds.

Table 1
Summary of single job performances on YARN and Mesos

Case	Runtime	Maximum number of containers used
YARN, short map-reduce job	21 s	2 (including application master)
Mesos, short map-reduce job	28 s	2
YARN, long map-reduce job	1061 s	31 (including application master)
Mesos, long map-reduce job	1129 s	11
YARN, short Spark job	18 s	3 (including application master)
Mesos, short Spark job	14 s	4

4.2 Mixed Job and Framework Performance (Scenario 1)

To test the scheduling performance of Mesos and YARN, we've created a client that submits map-reduce and Spark jobs periodically. In this scenario, a micro map-reduce and a CPU Spark job was submitted every 10 seconds, and a long batch map-reduce job every 100 seconds. A total of 22 jobs were submitted.

Using YARN as a platform, with the fair scheduler and unlimited application preferences, we were able to encumber the system to a point, where the context switching and administration overhead turned each running application into a zombie as `NodeManagers` were overwhelmed. As seen in *Figure 4*, after completing 22 applications, the last 20 never reached a complete state, but actually did not make any progress in hours. The scenario became complicated for YARN, when the long-job entered the cluster and a huge portion of resources were allocated to it, rendering micro-job executions slower, causing them to pile up. It became evident that concurrent application limits are crucial for performance, after a certain threshold on YARN as context switching and parallelism overhead went out of control. For this system and with this scenario, it happened with 15 applications. As memory-intensive applications were still running and requesting a resource vector with memory greater than (1 CPU, 1 GB), in average, 12 virtual cores were never used. The reserved values on the dimension of memory were introduced by the long running map-reduce job. The characteristic leap of the allocated plot line refers to the time when the first long job appeared and started to acquire all available resources.

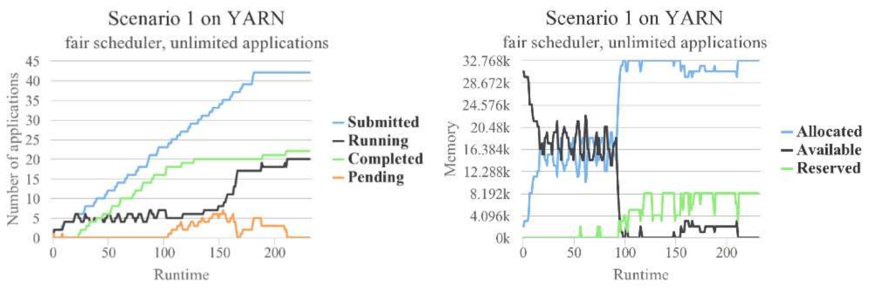


Figure 4
Evaluation of scenario 1 on YARN using fair scheduler with unlimited applications configuration in terms of number of applications and memory usage

Using YARN’s capacity scheduler with a limit of 4 concurrent applications, this scenario was completed in 2168 seconds and as seen in *Figure 5,* compared to fair scheduler with an application limit of 10, was slower. Fair scheduler completed applications in 2132 seconds, while also performed with a better response-time as smaller jobs were able to run earlier. For larger job sizes, capacity scheduler provided a better response-time with a lower application limit. In the case of capacity scheduler no reservations were made for new containers. By not reserving containers, it seems a few containers were unused and scheduled on-the-fly after they became available.

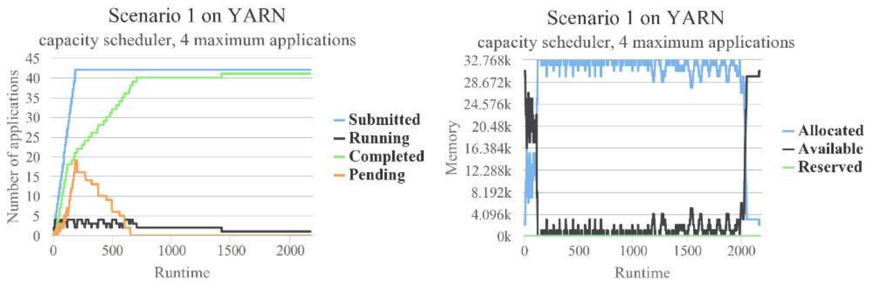


Figure 5
Evaluation of scenario 1 on YARN using capacity scheduler with maximum of 4 concurrent applications in terms of number of applications and memory usage

The FIFO scheduler with unlimited applications completed this scenario, on average 2153 seconds. This scheduling scheme can hurt smaller jobs and can cause starvation when a single, long job gets all resources as seen in *Figure 6*.

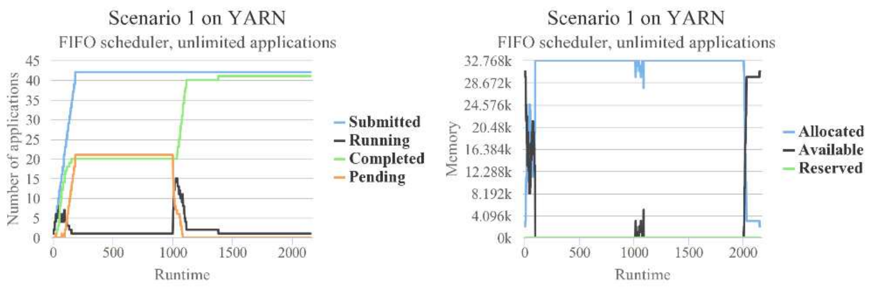


Figure 6

Evaluation of scenario 1 on YARN using FIFO scheduler with unlimited applications configuration in terms of number of applications and memory usage

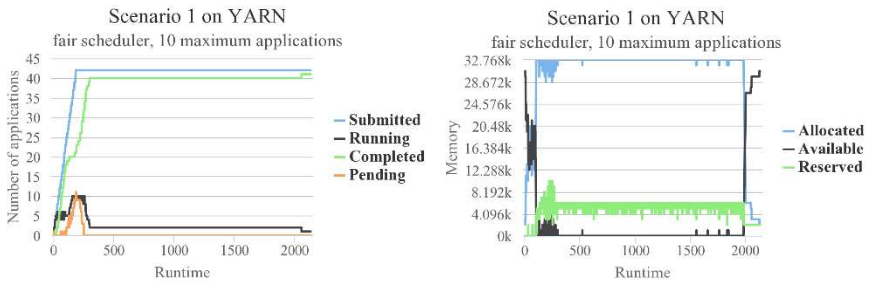


Figure 7

Evaluation of scenario 1 on YARN a maximumir scheduler with maximum of 10 concurrent applications in terms of number of applications and memory usage

Comparing the DRF implementation of Mesos to YARN's, YARN was able to perform much better and achieved a high utilization by allocating 100% of the available cluster resources for a long period of time, as shown on *Figure 7*. With the use of reserved amounts by the scheduler, containers were allocated and ran much faster achieving a higher utilization, than the capacity scheduler. Mesos, on the other hand, was not able to utilize all cluster resources. For a long time, the resource manager reported 4 CPUs and 6.1 GB memory idle, but the fine-grained, rapid tasks of Spark were utilizing the 4 CPUs as seen in *Figure 8*. Spark was set up in fine-grained mode in the first place, which means a separate Mesos task was launched for each Spark task. This allows frameworks to share cluster resources in a very fine granularity, but it comes with an additional overhead for managing task lifespans in a rapid rate. Focusing on the number of cores in the fine- and coarse-grained setup this behavior seems clear, as the fast task allocation pattern appears on the plot in the fine-grained case. A noticeable difference shows in the memory allocation pattern of different task-resolutions as (Spark) tasks with a lifespan measured in milliseconds allocated containers with <1 CPU, 128 MB> resource vectors instead of <1 CPU, 512 MB> or <1 CPU, 768 MB> as in the coarse-grained mode. In certain circumstances, it might be a good practice to place and force very fast tasks of different fine-grained setup frameworks next to memory-intensive jobs to improve utilization and fairness with Dominant Resource Fairness.

The map-reduce jobs were managed by a single `JobTracker` and Spark jobs were submitted by multiple Spark clients. By increasing the number of Spark clients, the utilization improved. *Figure 8* shows the utilization best achieved while 6 Spark frameworks were active.

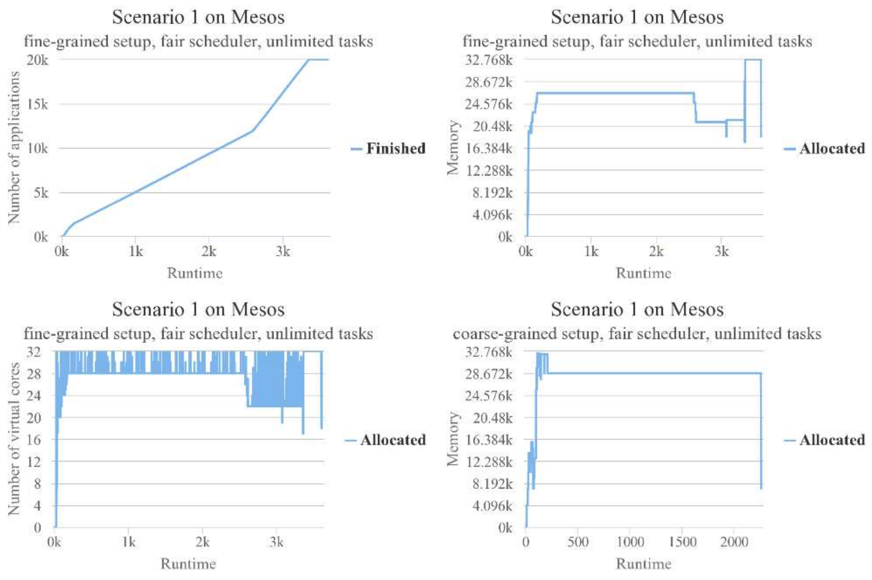


Figure 8

Evaluation of scenario 1 on Mesos using coarse- and fine-grained setup with fair scheduler and unlimited tasks configuration in terms of number of applications, virtual core and memory usage

In this scenario, multiple issues were found with Mesos. On some runs, an average of 24 `TaskTrackers` were lost and some of them were stuck in staging status, never reached running state. Also, one or two slaves were tended to disconnect from the master and froze in the first minutes of this scenario. The recorded and aggregated results were used, when Mesos did not lose a task.

The following table shows a summary of the results experienced from scenario 1.

Table 2
Summary of runtimes and average utilization experienced in scenario 1

Case	Runtime	Average utilization (CPU)	Average utilization (memory)
YARN, fair scheduler, unlimited	infinite	100%	100%
YARN, capacity scheduler, 4	2168 s	82.35%	89.87%
YARN, FIFO scheduler, unlimited	2153 s	83.27%	92.71%
YARN, fair scheduler, 10	2132 s	86.27%	92.90%
Mesos fine, fair scheduler, unlimited	3604 s	83.96%	78.46%
Mesos coarse, fair scheduler, unlimited	2256 s	90.81%	89.25%

The Spark implementation in fine-grained mode using Mesos spanned close to 20000 tasks in this scenario, it has put a strain worth mentioning on the master to schedule resources. While YARN performed about 1.7 times better than Mesos (with fine setup) with a relaxed (unlimited applications or tasks) DRF setting, due to the lightweight nature of Mesos it handled fine-grained tasks better as a first level scheduler. It has become clear that Mesos is more reliable and more suited for running large amounts of frameworks and tasks-per-framework with very fine-grained tasks.

4.3 Micro-Job Performance (Scenario 2)

In scenario 2, we've prepared a script, which submitted 4 micro-applications or jobs if you will, 2 map-reduce and 2 Spark job in each 10th second for 30 times. A total of 120 jobs reached the cluster. Our goal were to evaluate how fast short jobs can enter and leave the cluster on both systems and to see if there's any chance of overwhelming the slaves by increasing parallelism overhead to an undesirable level.

It has been shown in the demonstration of Mesos running a long batch job, `TaskTrackers` have a high startup overhead so our expectations were met about the difficulties these cases would produce. A standby TaskTracker might provide significant benefit regarding task start-ups, but it would also introduce data-locality problems, since a data might not be available where our TaskTracker has been deployed. Designing heuristics to keep TaskTrackers wisely on certain nodes, suggested by workload statistics, would not solve all of our problems on a long term. As seen on *Figure 9* YARN performed very well, by not letting pending tasks to reach 3 as applications were able to finish in a fast rate and were not interrupted by and overcrowded cluster. Applications were completed linearly with time and on average roughly 10 were running concurrently. In case of Mesos, as seen on the curve of staging tasks, in the first 60 seconds every Spark job entered the system were able to run and complete without unnecessary staging.

In the first 100 seconds only `TaskTrackers` were staging for a longer period of time, which meant that because of their startup time, those rapid map-reduce jobs arriving in every 10 seconds were not reached running status fast enough. Due to the fact that map-reduce programs were spawning on `TaskTrackers`, unnecessary parallelism appeared on slaves and about 20 map-reduce jobs were running concurrently along with the Spark jobs on the cluster.

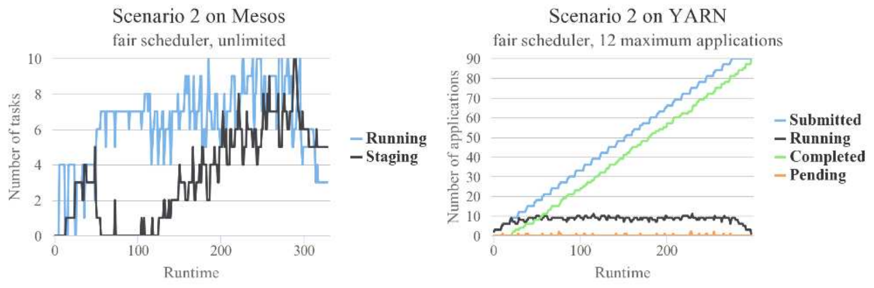


Figure 9
Evaluation of scenario 2 on Mesos (unlimited tasks) and YARN (maximum 12 applications) using fair scheduler in terms of number of tasks and application

On the memory footprint produced by tasks running on Mesos as shown in *Figure 10*, the `TaskTrackers` crowding the cluster are visible on the 30th to 110th second interval. After that point a few of them were broken down to be able to offer resources to Spark programs. YARN, in contrast, kept resource consumption constant as applications were not able to encumber the cluster.

The container reservations used by YARN's fair scheduler helps applications to receive and utilize containers faster than the resource offer approach introduced by Mesos. Mesos completed this scenario in 328 seconds, while YARN in 297 seconds. Again, issues were found, but this time with the `JobTracker` (Hadoop v1): in some cases map-reduce jobs were stuck and never reached running state on `TaskTrackers`.

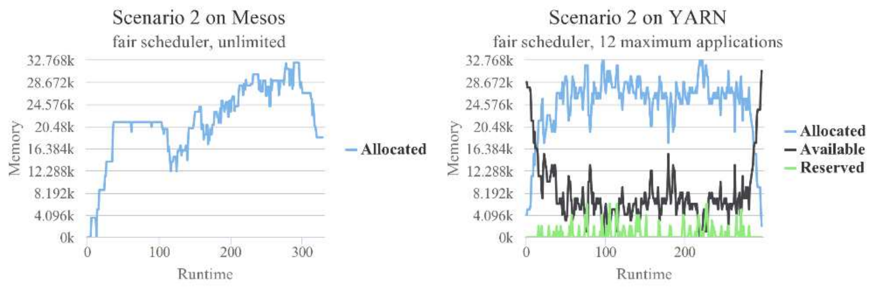


Figure 10
Evaluation of scenario 2 on Mesos (unlimited tasks) and YARN (maximum 12 applications) using fair scheduler in terms of memory consumption

4.4 Micro-Job Interruptions (Scenario 3)

To evaluate the problems related to granting fairness and job interruptions, we've prepared a scenario, where micro Spark programs were submitted on a long running map-reduce batch job. After the submission of the same long job, that was previously evaluated, for every 100 seconds a Spark CPU-heavy program was submitted, nine times longer overall.

Interrupting the map-reduce job's execution with micro Spark jobs on Mesos added, on average 9 seconds to the overall completion time, which became 1138 seconds. Recall the results of the same long job performance of YARN and Mesos from *Figure 1* and *Figure 2*. On YARN, the same scenario stretched the completion time of the map-reduce batch job, from 1061 to 1092 seconds.

On Mesos, 1 CPU was available with 1 GB RAM and the Spark client was able to initiate a start on a free container, where it completed in 48 seconds on average. Since YARN were utilizing all cluster resources during the execution of the map-reduce job most of the time, the Spark programs needed to wait on average 13.3 seconds to be able to progress to running state from pending state. Theoretically, every 2.6 seconds, a mapper finish (from the length map phase and number of maps) and its resources <1 CPU, 1 GB> frees up. On average, 3 containers were reserved by the `MRAppMaster` and the Spark job needs 2 containers (including the application master) to be able to run. When the Spark job reached the pending state, on average, 5 containers needed to free up, to be able to reach running state, which is roughly 13 seconds.

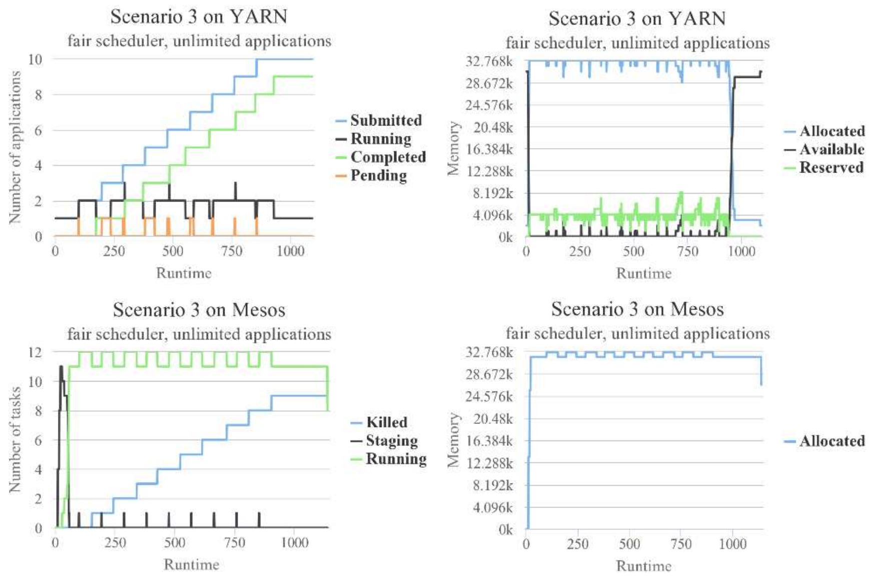


Figure 11
Evaluation of scenario 3 on Mesos (unlimited tasks) and YARN (unlimited applications) using fair scheduler in terms of number of tasks, application and memory usage

As seen in *Figure 11*, the same Spark job on YARN needed more time to start, mainly because the application master is necessary to be placed on a free container, which set back and slowed subsequent pending mappers. Utilization of YARN still proved to be better than in the case of Mesos.

Conclusions

Cluster management systems are the backbone of any Big Data analytical toolsets that are used by an Enterprise, their performance is determined significantly, by their design and greatly affects energy consumption for a data center. Evolution of Hadoop had the greatest impact in the motivation, concept and birth of these systems. We focused on the main two-level, open-source schedulers available, YARN and Mesos.

YARN is perfect for ad-hoc application deployment, as it ships the application to the requested node by carefully setting up the process in all cases. YARN has been made for an environment with higher security demands, as it protects the cluster from malicious clients and code in many cases. The differences in resource allocation methods showed that the push-method used by Mesos might hinder utilization and locality preferences in some cases, but proved to be faster than YARN's, which provides agility by using late-binding in opposite fashion. Applications running on YARN have the benefit of making better second-level scheduling decisions, because they have a global view of the cluster, whereas a framework have sight of only a subset of the cluster on Mesos. In a consequence of the resource allocation method, YARN supports preemption to prevent starvation. Restricted visibility of cluster resources might lead to losing work and resource hoarding used by Mesos can lead to a deadlock within the system. Mesos predicts outcomes to make quick scheduling decisions.

The functionality of YARN proved to be richer by providing convenient services for applications, but also supports more scheduling methods and algorithms. Capacity schedulers can work effectively when the workloads are well known. Fair scheduler introduces several problems with head-of-line jobs, but Delay scheduling addresses them and improves locality. HaSTE is a good alternative on YARN, when the goal is to minimize makespan in the cluster.

Regarding system parameters, the API provided by Mesos is more detailed and precise, but YARN gives a finer grained snapshot as each job runs as a separate application. Mesos does not know the job granularity of a connected framework, which can cause several problems. Deployment of YARN is usually more convenient, due to the higher level and more comprehensive interfaces available. It works more like an out-of-box product. During the evaluations, several issues were found with Mesos regarding permissions and node failures.

YARN has a wider and more diverse analytical toolset (frameworks) available than Mesos, but a practical decision between these platforms might include special requirements. The mainstream frameworks are mostly available on both systems.

As single job performance evaluations show, executors are difficult to set up the first time and can mean a slow response from frameworks like Hadoop MapReduce. YARN deploys mappers and reducers, much faster on the designated nodes and can provide better locality, also, this task-granularity provides better elasticity along with a faster execution. As TaskTrackers are expensive to deploy and they are long running, killing them to provide fairness is usually a significant drawback. Cached or standby TaskTrackers might provide significant improvements in task start-ups, but it would introduce other problems, for instance, a hindered data locality. Single job benchmarks also showed, that map-reduce jobs on Mesos run longer and consume more resources, which directly affects money spent in a cloud environment. YARN is 1.06 (in case of short map-reduce) and 1.33 (in case of long map-reduce) times faster than Mesos, but Mesos runs a micro-Spark job 1.28 times faster than YARN. It must be taken to account that YARN has to deploy the submitted application each time, while the framework's master runs separately on Mesos.

Multiple scenarios showed that the "concurrent application limit" is crucial for performance after a certain threshold on YARN. Using preemption, YARN performed about 1.7 times better than Mesos with the fine-grained setup, but Mesos handles large amounts of tasks better as a first level scheduler. Mesos is more reliable and more suited for running large amounts of fine-grained tasks. With the same setup, YARN provides a 1.05 times faster execution and 4.54% less CPU consumption. It is evident that the container reservations, used by YARN's fair scheduler, can utilize and provide containers to applications faster than the resource offer approach introduced by Mesos. Other scenarios showed that overall utilization on a YARN cluster is better, along with a 1.10 times faster execution.

References

[1] WHITE, Tom. Hadoop: The definitive guide. O'Reilly Media, Inc., 2012

[2] DEAN, Jeffrey; GHEMAWAT, Sanjay. MapReduce: simplified data processing on large clusters. Communications of the ACM, 2008, 51.1: 107-113

[3] LIANG, Fan, et al. Performance benefits of DataMPI: a case study with BigDataBench. In: Big Data Benchmarks, Performance Optimization, and Emerging Hardware. Springer International Publishing, 2014, pp. 111-123

[4] JIA, Zhen, et al. Characterizing and subsetting big data workloads. arXiv preprint arXiv:1409.0792, 2014

[5] TAN, Jian; MENG, Xiaoqiao; ZHANG, Li. Performance analysis of coupling scheduler for mapreduce/hadoop. In: INFOCOM, 2012 Proceedings IEEE. IEEE, 2012, pp. 2586-2590

[6] CHEN, Yanpei, et al. Energy efficiency for large-scale mapreduce workloads with significant interactive analysis. In: Proceedings of the 7th ACM european conference on Computer Systems. ACM, 2012, pp. 43-56

[7] KUMAR, K. Ashwin; DESHPANDE, Amol; KHULLER, Samir. Data placement and replica selection for improving co-location in distributed environments. arXiv preprint arXiv:1302.4168, 2013

[8] HUNT, Patrick, et al. ZooKeeper: Wait-free Coordination for Internet-scale Systems. In: USENIX Annual Technical Conference. 2010, p. 9

[9] "Corona" 2013 [Online] Available: https://github.com/facebookarchive /hadoop-20/tree/master/src/contrib/corona

[10] ZAHARIA, Matei, et al. Spark: cluster computing with working sets. In: Proceedings of the 2nd USENIX conference on Hot topics in cloud computing. 2010, p. 10-10

[11] SCHWARZKOPF, Malte, et al. Omega: flexible, scalable schedulers for large compute clusters. In: Proceedings of the 8th ACM European Conference on Computer Systems. ACM, 2013, pp. 351-364

[12] "Apache Hadoop 2.5.2," 2014 [Online] Available: http://hadoop.apache.org/docs/r2.5.2/

[13] "Apache Mesos" 2014 [Online] Available: https://github.com/apache/mesos

[14] "Hadoop on Mesos" 2014 [Online] Available: https://github.com/mesos/hadoop

[15] "Spark" 2014 [Online] Available: https://github.com/apache/spark

Incremental Ensemble Learning for Electricity Load Forecasting

Gabriela Grmanová, Peter Laurinec, Viera Rozinajová, Anna Bou Ezzeddine, Mária Lucká, Peter Lacko, Petra Vrablecová, Pavol Návrat

Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, Ilkovičova 2, 842 16 Bratislava, Slovak Republic
{gabriela.grmanova, peter.laurinec, viera.rozinajova, anna.bou.ezzeddine, maria.lucka, peter.lacko, petra.vrablecova, pavol.navrat}@stuba.sk

Abstract: The efforts of the European Union (EU) in the energy supply domain aim to introduce intelligent grid management across the whole of the EU. The target intelligent grid is planned to contain 80% of all meters to be smart meters generating data every 15 minutes. Thus, the energy data of EU will grow rapidly in the very near future. Smart meters are successively installed in a phased roll-out, and the first smart meter data samples are ready for different types of analysis in order to understand the data, to make precise predictions and to support intelligent grid control. In this paper, we propose an incremental heterogeneous ensemble model for time series prediction. The model was designed to make predictions for electricity load time series taking into account their inherent characteristics, such as seasonal dependency and concept drift. The proposed ensemble model characteristics – robustness, natural ability to parallelize and the ability to incrementally train the model – make the presented ensemble suitable for processing streams of data in a "big data" environment.

Keywords: big data; time series prediction; incremental learning; ensemble learning

1 Introduction

Generating large amounts of data has become part of our everyday life. In reality, human activities produce data that in recent years have rapidly increased, e.g. as measured through various sensors, regulation systems and due to the rapid development of information technologies [3]. "Big data" significantly changes the nature of data management as it introduces a new model describing the most significant properties of the data -volume, velocity and variety. Volume refers to the vast amounts of data requiring management, and it may not stem from the number of different objects, but from the accumulation of observations about these objects in time or in space. Velocity can be determined by the rate of acquisition of streams of new data, but also by application requirements, where it is necessary to make a very fast prediction, as the result of a particular user's request. This will comprise research of methods and models for big data analysis, whether with low latency, or even in real time.

In our work, we focus on stream data coming from smart metering. The smart meters are able to send measurements of power consumption every 15 minutes thus, providing new possibilities for its modelling and prediction. The most useful aspect of having this vast amount of data is the ability to forecast the power demand more precisely. This is particularly important when viewed with regard to the fact that the possibility to store electricity is very limited. With accurate predictions, the distributor can reliably deliver electricity and fulfil the power authorities' regulations, which protect the distribution network from being at too high or too low a voltage. It also helps to flexibly react to different unexpected situations like large-scale blackouts.

The number of smart meters increases rapidly every day which results in production of large amount of data. Classical methods can fail to process such amount of data in reasonable amount of time; therefore it is necessary to focus on parallel and distributed architectures and design methods and applications that are able to automatically scale up depending on the growing volume of data.

The classical prediction methods of electricity consumption are: regression analysis and time series analysis models. These approaches will not be sufficient in the near future, as the European Union's efforts are aimed at introducing an intelligent network across the whole of the European Union. This fact raises new perspectives in modelling and predicting power demand.

A significant feature of many real-world data streams is *concept drift*, which can be characterized as the arbitrary changes of data characteristics. The occurrence of concept drift in a data stream can make classical predictive techniques less appropriate therefore, new methods must be developed. The typical example of concept drift is a change of workload profile in a system for controlling the load redistribution in computer clusters [48] or a change of user's interests in information filtering and recommendation modelling [14], [26]. In power engineering are concept drifts caused by change of consumers' behaviour during holidays, social events, different weather conditions, or summer leaves in big enterprises – the biggest electricity consumers.

This paper introduces a new approach to electrical load forecasting. It takes into consideration the aspect of concept drift, and is based on the principle of ensemble learning. It is organized as follows: the second chapter is devoted to the characteristics of the problem and the third contains the summary of the related work. In the next chapter we describe our proposed approach (the incremental heterogeneous ensemble model for time series prediction). The experimental evaluation is presented in Chapter 5 and the overall evaluation and discussion is given in Chapter 6.

2 Characteristics of the Problem

As mentioned earlier, after the widespread introduction of smart meters, the data provided will satisfy the first characteristic of Big Data based on volume. To analyse these data, it is appropriate to propose parallel methods that can be solved in distributed environments. Because electricity loads can be seen as a stream of incoming data, it is necessary to focus on adaptive methods that are able to learn incrementally.

There are also additional important characteristics that must be taken into account – the presence of concept drifts and strong seasonal dependence. The values of any variable evolving in time, such as the electricity load, often change their behaviour over time. These changes may be sudden or gradual. In the literature, both types of changes are termed *concept drift*. Narasimhamurthy and Kuncheva [35] define the term *concept* as the whole distribution of the problem and represent it by joined distribution of data and model parameters. Then, concept drift may be represented by the change of this distribution [15]. Besides cases of concept drift when the change is permanent, one can often observe changes that are temporary. They are caused by the change of some conditions and after some time, these conditions can again change back. Moreover, seasonal changes may be considered to be concept drift, too.

In electricity load measurement, two types of concept drift can appear. The first one is permanent or temporary change that may be caused by the change of economical or environmental factors. The second type of concept drift is seasonal, caused by seasonal changes of weather and the amount of daylight. Seasonal dependency can be observed as daily, weekly and yearly levels. That is why it is necessary to consider these two possible sources of concept drift in any model proposal.

3 Literature Review

In this section we present methods used to compute time series predictions – classical, incremental and ensemble approaches.

3.1 Classical Approaches

Classical approaches to time series prediction are represented mainly by regression and time series analysis. Regression approaches model the dependencies of target variables on independent variables. For electricity load prediction, the independent variables can be the day of the week, the hour of the day, the temperature, etc. Plenty of different regression models were presented in the literature, such as a step-wise regression model, a neural network and a decision tree [45].

Because of strong seasonal periodicities in electricity load data, time series models are often used to make predictions. Mainly, Box-Jenkins methodology [5] with AR, MA, ARMA, ARIMA and derived models are applied.

However, the classical approaches are not able to adapt to incoming streams of data and thus, are not suitable for electricity load demand forecasting.

3.2 Incremental Learning

Incremental learning algorithms are able to adapt to new emerging data. They process new data in chunks of appropriate size. They can possibly process the data chunks by off-line algorithms.

Polikar et al. [39] defined the four desired properties of an incremental learning algorithm – the ability to learn new information from arriving data, the capability of working independently on historical data, the storage of previously learned knowledge, and the accommodation of new classes of data on their arrival. Minku [32] extends this definition and emphasizes that, in changing environments, where the target variable might change over time, only the useful knowledge will be stored.

Most of the incremental learning algorithms we encountered in the literature are based on machine learning, e.g. incremental support vector machines [51] and extreme learning machines [17]. Recently, the incremental ARIMA algorithm was proposed for time series prediction [34].

Usually, the incremental learning algorithms alone cannot sufficiently treat changes in the target variable. In order to cope with a changing environment, groups of predictors, i.e. ensemble models, are used to achieve better predictions.

3.3 Ensemble Learning

Ensemble learning is an approach that uses a set of base models, where each model provides an estimate of a target variable – a real number for a regression task. The estimates are then combined to make a single ensemble estimate. The combination of base estimates is usually made by taking a weighted sum of base estimates. The idea behind it is that the combination of several models has the potential to provide much more accurate estimates than single models. In addition, they have several more advantages over single models, namely the scalability, the natural ability to parallelize and the ability to quickly adapt to concept drift [52]. A great introduction to ensemble learning can be found in [32].

Several empirical studies showed that the accuracy of the ensemble depends on the accuracy of base models and on the diversity among them [11], [12], [28]. The diversity of base models may be accomplished by two different approaches – *homogeneous* and *heterogeneous ensemble learning* [52]. In homogeneous learning, the ensemble is formed by models of the same type that are learned on different subsets of available data. The heterogeneous learning process applies different types of models. The combination of homogeneous and heterogeneous approaches was also presented in the literature.

The best known methods for homogeneous ensemble learning are bagging [6] and boosting [13]. These approaches have been shown to be very effective in improving the accuracy of base models. To accomplish adaptive ensemble learning for online stream environments, two approaches are known from the literature. The first one – *incremental ensemble learning* – learns the base methods from different chunks of data. The second one – *the ensemble of online/incremental methods* – uses adaptive base methods, that are updated in an online (after each example) or incremental (after a chunk of data is available) manner. *Incremental ensemble learning* employs non-incremental algorithms to provide incremental learning. Wang et al. [46] proposed a general framework for mining data streams using weighted ensemble classifiers. The proposed algorithm adapts to changes in data by assigning weights to classifiers proportional to their accuracy over the most recent data chunk. Another approach was published by Kolter and Maloof [27]. They developed a dynamic weighted majority algorithm, which creates and removes weighted base models dynamically based on changes in performance.

Ensemble of online/incremental methods employ online/incremental base models. These approaches include online versions of well-established approaches such as online bagging and online boosting incorporating online base models [37], [38]. Another approach proposed by Ikonomovska et al. [24] introduces an ensemble of online regression and option trees.

Heterogeneous ensemble learning represents a different way of introducing the diversity of base models into ensemble, with the aim of combining the advantages of base algorithms and to solve problems of concept drift [16], [54], [40]. Different models are trained on the same training dataset; in the case of stream data on the up-to-date data chunk.

From the literature several combinations are also known of heterogeneous and homogeneous learning. Zhang et al. [55] present aggregate ensemble learning, where different types of classifiers are learned from different chunks of data.

The essential part of the ensemble learning approach is the method that is used to combine estimates of base models. For regression problems, this is done by a linear combination of the predictions. The sum of the weights which are used in the combination is 1. The weights are computed by different methods, such as basic or general ensemble methods, linear regression models, gradient descent or by evolutionary or biologically inspired algorithms, e.g. particle swarm optimization or “cuckoo search” [31], [30], [50].

Ensemble learning was also used to predict values of time series. Shen et al. [42] apply an ensemble of clustering methods to cluster 24-hour segments. Based on cluster labels, the segments are converted to sequences. Each testing sequence is matched to the training subsequences, and matching subsequences are averaged to make the prediction for a subsequent segment of the testing sequence. The predictions based on 5 different clustering methods are combined in the ensemble, where the weights are iteratively updated. Chitra and Uma [7] present an ensemble of RBF-network, k-nearest neighbour and self-organizing maps for a time series prediction. Wichard and Ogorzałek [47] describe the use of an ensemble method for their time series prediction. They use an ensemble of linear and polynomial models, k-nearest neighbour, nearest trajectory models and neural networks, with an RBF-network for "one-step- ahead" prediction.

All of these approaches use ensembles of regression models for generating time series predictions. They do not take explicitly any seasonal dependence into account and do not use time series analysis methods to make predictions.

4 The Incremental Heterogeneous Ensemble Model for Time Series Prediction

In this section we propose the *incremental heterogeneous ensemble model for time series prediction*. The ensemble approach was chosen for its ability to adapt quickly to changes in the distribution of a target variable and its potential to be more accurate than a single method. Since we focus on time series with strong seasonal dependence, in ensemble models we take into account different types of seasonal dependencies. Models for yearly seasonal dependence need to be computed based on one year of data. These models can be recomputed once a year. The models coping with daily seasonal dependence need only data from one to several days and can be computed in an incremental manner. The potential of the proposed ensemble is its ability to deal with the scalability problems of big data. Predictions of base models can be computed in parallel or in distributed environment in order to reduce computation time and to scale up to incoming amount of data which makes the proposed ensemble suitable for big streams of data.

The base models used in the ensemble are of two types – regression models and models for time series analysis. Regression models can potentially incorporate additional dependencies, such as temperature. Time series analysis models are suitable to capture seasonal effects.

4.1 Incorporating Different Types of Models

The proposed ensemble model incorporates several types of models for capturing different seasonal dependencies. The models differ in *algorithm*, *size of data chunk* and *training period* (see Figure 1). Different algorithms are assumed in order to increase the diversity of the models. The size of each data chunk is chosen in order to capture particular seasonal variation, e.g. data from the last 4 days for daily seasonal dependence. However, the model that is trained on a data chunk of 4 days' data, can be trained again as soon as the data from the next day (using a 1-day training period) are available. The new data chunk overlaps with the previous one in 3 days.

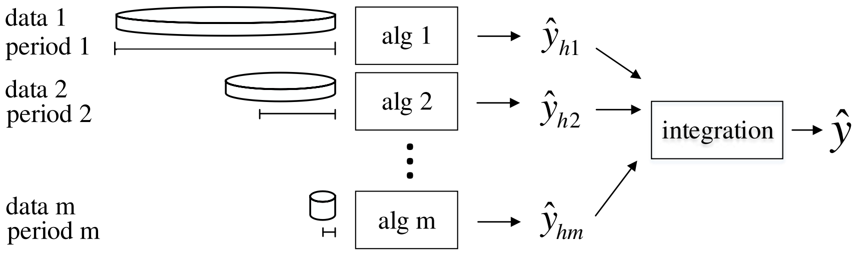


Figure 1
Schematic of ensemble learning

The ensemble model is used to make one-day predictions. Let h be the number of observations that are daily available. At day t, the ensemble makes h predictions by the weighted average of predictions made by m base models. After the observations for the current day are available, the prediction errors are computed. Based on computed errors, the weights are updated and each base model $i=1, ..., m$, for which t fits its training period p_i, is retrained on a data chunk of size s_i.

Let $\hat{Y}^t$be the matrix of predictions of m base methods for the next h observations at day t:

$$\hat{Y}^t = \begin{pmatrix} \hat{y}_{11}^t & \cdots & \hat{y}_{1m}^t \\ \vdots & \ddots & \vdots \\ \hat{y}_{h1}^t & \cdots & \hat{y}_{\mathrm{hm}}^t \end{pmatrix} = (\hat{y}_1^t \quad ... \quad \hat{y}_m^t)$$

and $w^t = (w_1^t \quad ... \quad w_m^t)^T$is a vector of weights for m base methods at day t before observations of day t are available. Weights w_j^t are initially set to 1. Weights and particular predictions are combined to make an ensemble prediction $\hat{y}^t = (\hat{y}_1^t \quad ... \quad \hat{y}_m^t)^T$. The ensemble prediction for k-th ($k = 1, ..., h$) observation is calculated by:

$$\hat{y}_k^t = \frac{\sum_{j=1}^{m} \hat{y}_{\mathrm{kj}}^t \bar{w}_j^t}{\sum_{j=1}^{m} \bar{w}_j^t}$$

where $\bar{w}^t$is a vector consisting of weights rescaled to interval $\langle 1, 10 \rangle$.

After observations of day t are available, the weights vector can be recomputed. From the prediction matrix $\hat{Y}^t$ and the vector of h current observations $y^t = (y_1^t \; \dots \; y_h^t)^T$, the vector $e^t = (e_1^t \; \dots \; e_h^t)^T$ of errors for m methods is computed. The error of each method is given by $e_j^t = \text{median}\left(\left|\hat{y}_j^t - y^t\right|\right)$. A vector of errors e^t is used to update the weights vector of base models in the ensemble. The weight for j-th method is calculated by:

$w_j^{t+1} = w_j^t \frac{\text{median}(e^t)}{e_j^t}$The advantages of the presented type of weighting is its robustness and the ability to recover the impact of base methods. The weighting and integration method is robust since it uses the median absolute error and the median of errors. In contrast to the average, the median is not sensitive to large fluctuations and abnormal prediction errors. A rescaling method, one that does not let the particular weights drop to zero, enables the ensemble to recover the impact of particular base methods in the presence of concept drift.

4.2 Base Models

In heterogeneous ensemble models, it is important to integrate the results of diverse base methods. We used 11 different algorithms. The methods are of different complexity, from very simple, e.g. a naïve average long-term model, to complex, such as support vector regression. They assume different seasonal dependencies, from daily to yearly. The presented base methods can be divided into the set of methods based on regression analysis and those based on time series analysis.

4.2.1 Regression Algorithms

Multiple linear regression (MLR) attempts to model the relationship between two or more explanatory variables and a response variable by fitting a linear equation to observed data. Rather than modelling the mean response as a straight line, as it is in simple regression, the model is expressed as a function of several explanatory variables.

Support Vector Machines are an excellent tool for classification, novelty detection, and regression (SVR). It is one of the most often used models for electricity load forecasting. SVM is a powerful technique used in solving the main learning problems. We have used the method based on epsilon-regression based on the radial basis Gaussian kernel, and also tested it in combination with the wavelet transform ($\varepsilon = 0.08$ for deterministic part and 0.05 for fluctuation part) [53].

4.2.2 Time Series Algorithms

The autoregressive model (AR) expresses the current value of electricity load as a linear combination of previous electricity load values and a random white noise

[33]. The current value of the modelled function is expressed as a function of its previous *n* values on which it is regressed.

Feed-forward neural networks (NNE) are biologically inspired universal approximation routines [22]. They were successfully used for solving prediction problems [43]. R package *forecast* [23] contains the *nnetar* method, which is a feed-forward neural network with a single hidden layer and lagged inputs, for forecasting univariate time series. It provides the capability to train a set of neural networks on lagged values for one-step forecasting. The prediction is an average of those neural networks predictions. Number of neurons in hidden layer was determined as a half of the number of input nodes plus one.

The *Holt-Winters exponential smoothing* (HW) [20], [49] is a prediction method applied to a time series, whereby past observations are not weighted equally, as it is in ARMA models, but the weights decrease exponentially with time. So the data that are closer in time can influence the modelling more strongly. We have considered seasonal changes with and without any trend in triple exponential smoothing (we have chosen the smoothing parameters $\alpha = 0.15$, $\beta = 0$, $\gamma = 0.95$), and combined this model with the wavelet transform (shrinkage method was chosen soft thresholding and threshold estimation was universal). The original load data series were decomposed into two parts - deterministic and fluctuation components, and then the regression of both parts was calculated separately. The resulting series were obtained with suitable wavelet coefficient thresholds and the application of the wavelet reconstruction method.

Seasonal decomposition of time series by loess (STL) is a method [8] that decomposes a seasonal time series into three parts: trend, seasonal and remaining. The seasonal component is found by *loess (local regression)* smoothing the seasonal sub-series, whereby smoothing can be effectively replaced by taking the mean. The seasonal values are removed, and the remainder is smoothed to find the trend. The overall level is removed from the seasonal component and added to the trend component. This process is iterated a few times. The remaining component represents the residuals from the seasonal plus trend fit.

STL decomposition works similarly to wavelet transform. For the resulting three time series (seasonal, trend and remainder) the result is used separately for prediction with Holt-Winters exponential smoothing and ARIMA model.

The ARIMA model has been introduced by Box and Jenkins [5] and is one of the most popular approaches in forecasting [21]. It is composed of three parts: autoregressive (AR), moving average (MA), and the differencing processes. In the case of non-stationary processes, it is important to transform the series into a stationary one and that is usually done by differentiation of the original series.

Seasonal naïve method-Random walk (SNaive) is only appropriate for time series data. All forecasts are simply set to be the value of the last observation. It means that the forecasts of all future values are set to be equal to the last observed value.

A similar method is also useful for highly seasonal data, where each forecast value is set to be equal to the last observed value from the same season of the year (e.g., the same month of the previous year).

Double seasonal exponential smoothing (TBATS) forecasting is based on a new state space modelling framework [10], incorporating Box-Cox transformations, Fourier series with time varying coefficients and ARMA error correction. It was introduced for forecasting complex seasonal time series that cannot be handled using existing forecasting models. These types of complex time series include time series with multiple seasonal periods, high frequency seasonality, non-integer seasonality and other effects. The modelling is an alternative to existing exponential smoothing models, and has many advantages.

Naïve average long-term method is based on the assumption that non-seasonal patterns [36] and trends can be extrapolated by means of a moving-average or smoothing model. It is supposed, that the time series is locally stationary and has a slowly varying mean. The moving (local) average is taken for the estimation of the current value of the mean and used as the forecast for the near future. The simple moving average model predicts the next value as a mean of several values. This is a compromise between the mean model and the random-walk-without-drift-model.

Naïve In median long-term method is an alternative to the previous method. The use of a moving average is not able to react in the case of rapid shocks or other abnormalities. In such cases a better choice is to take a simple moving median over the last *n* time series' items. A moving average is statistically optimal for recovering the underlying trend of the time series when the fluctuations about the trend are normally distributed. It can be shown that if the fluctuations are Laplace distributed, then the moving median is statistically optimal [2].

5 Experimental Evaluation

In this section we describe how data is used for the evaluation of the ensemble method; we provide details of the experiments and then we present the results.

5.1 Data

An experimental sample of data comes from smart meters installed in Slovakia that perform measurements every 15 minutes. Currently, the smart meters operate in around 20,000 consumers' premises. Based on legislation, this number will soon be higher and the amount of incoming data will significantly increase. The data has the potential to become "big" and "fast", because of its incremental and stream character. The consumers are small and medium enterprises. The data are anonymized and only postal codes are available. We created 10 samples by grouping customers according to regions. By doing this we simulate electricity load values at secondary distribution substations. We summed the quarter-hourly data to predict the load of the regions. Table 1 describes the data samples. The studied data samples show collected values from July 1st, 2013 to February 15th, 2015 (596 days, see Figure 2). The sudden changes in load were observed during holidays (e.g., summer leave, and Christmas).

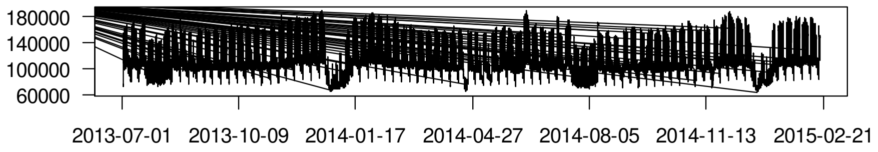


Figure 2

Electricity consumption for Bratislava region over period of 596 days (in kW per 15 minutes)

Table 1

Description of ten data samples and their electricity loads (in kW per 15 minutes)

postal code	region	no of delivery points	average	average per delivery point
04	Košice	722	35,501.854	49.172
05	Poprad	471	17,135.133	36.380
07	Trebišov	382	11,571.184	30.291
08	Prešov	580	18,671.795	32.193
8	Bratislava	1314	119,691.911	91.090
90	Záhorie	773	41,402.715	53.561
92	Piešťany	706	74,340.781	105.296
93	Dunajská Streda	594	28,196.959	47.470
95	Partizánske	584	34,298.912	58.731
99	Veľký Krtíš	114	2,124.887	18.639

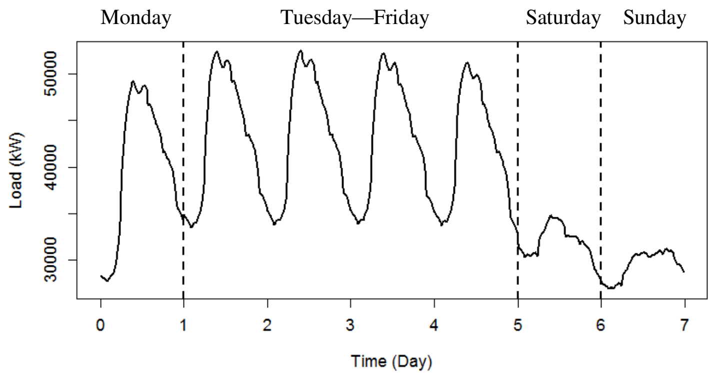


Figure 3

Average weekly electricity load (without holidays)

The load during the typical week (see Figure 3) consists of the four segments – Mon, Tue—Fri, Sat and Sun. To minimize the noise in the data and to improve our predictors we considered only the Tue—Fri segment, i.e. the days with similar behaviour.

5.2 Measures of Prediction Accuracy

To measure prediction accuracy we utilize three measures. *Mean absolute error* (MAE) and *root mean squared error* (RMSE) are commonly used measures of prediction error in time series analysis. The main difference between RMSE and MAE is that RMSE amplifies large errors. *Symmetric mean absolute percentage error* (sMAPE) is an accuracy measure based on relative (percentage) errors that enables us to compare percentage errors for any time series with different absolute values:

$$\text{sMAPE} = \frac{1}{n}\sum_{t=1}^{n}\frac{|\hat{y}_t - y_t|}{(\hat{y}_t + y_t)}$$

5.3 Experiments

To design the experiments, the best data chunk sizes for particular models were found experimentally. The most precise predictions for regression methods (MLR and SVR) were for days. Time series analysis models (AR, HW, STL+EXP, STL+ARIMA) coping with daily seasonality and NNE performed the best with data chunk size equal to 10 days. Based on its nature, SNaive needed only a 1-day long data chunk. Long-term double seasonal exponential smoothing (TBATS), incorporating two seasonal dependencies with 1 and -day periods, used data chunks the size of 41days – 1/3 of days of the test set. Naïve average and median log-term models use 1-year data chunks. In fact, in our experiments we had only 116 days in the test set for which observations from the previous year were available. Thus, 116-days long data chunk was used.

The training period for methods working with short-term seasonal dependency was 1 day. Models coping with yearly seasonal dependency have a 1 year period and since we had less than 2 years of data available, they were trained only once and were not further retrained.

Since there were only available data for training (both previous 10 days and previous 1 year observations) for the period July 1st, 2014 – February 15th, 2015, these were used as a test set. Namely: only non-holiday Tue-Fri days were assumed. The test set consisted of 116 days each having 96 observations. Initially, models were trained on respective chunks from a training set with equal weights in the ensemble. Then, the models were incrementally retrained according to their periods, while subsequently adding new data from the test set and ignoring the old ones.

The experiments were provided in an R environment. We used methods from a standard *stats* package and from *forecast* [23] (STL+EXP, STL+ARIMA, NNE, SNaive and TBATS), *wmtsa* [9] (wavelets) and kernlab [25] (SVR) packages.

5.4 Results

Figures 4 and 5 illustrate the incremental training process for a single region. It presents predicted and measured electricity loads (Figure 4), history of weights (Figure 5 top) and errors (Figure 5 bottom). An interesting observation of concept drift can be seen at *t*=10 and *t*=22, when errors sharply rise. In the history of weights, sharp changes can be seen, too. The concept drift was caused by the summer leave in bigger enterprises, which consume most of the electricity.

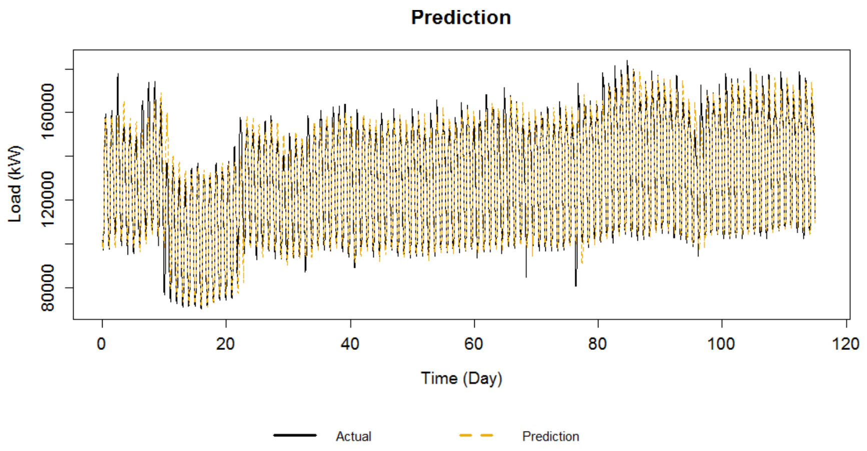


Figure 4

Results of prediction for Bratislava region. Concept drift at times t=10 and t=22 was caused by the summer leave in bigger enterprises, which consume the most of the electricity.

Tables 3 and 4 contain average errors of predictions and their standard deviations measured by sMAPE for every region and every base method plus the ensemble method. Tables show that there is no superior base method, which gives justification for the ensemble method, where errors are, in all tested cases, smaller.

We used the Wilcoxon rank sum test [19] to evaluate the *incremental heterogeneous ensemble model for time series prediction* against the best base method. The Wilcoxon rank sum test tests the statistical hypothesis whether errors of the ensemble are significantly lower than errors of the best base method used in the ensemble. The test used is a nonparametric alternative to the two-sample t-test. We used this nonparametric test because errors of predictions are not normally distributed (tested with Shapiro-Wilk test [41] and Q-Q plot). The Base method with the highest weight value at the end of the testing process is considered to be the best base method in the ensemble. A Statistical test on significance level α= 0.05 showed that in 9 out of 10 regions the ensemble method was significantly better than the best base methods in that region (see Table 5). The p-value exceeds the significance level for all but one region with errors measured by MAE, RMSE and sMAPE. Only for the Trebišov region, evaluated by RMSE measure, was the ensemble evaluated as smaller, but not significantly so.

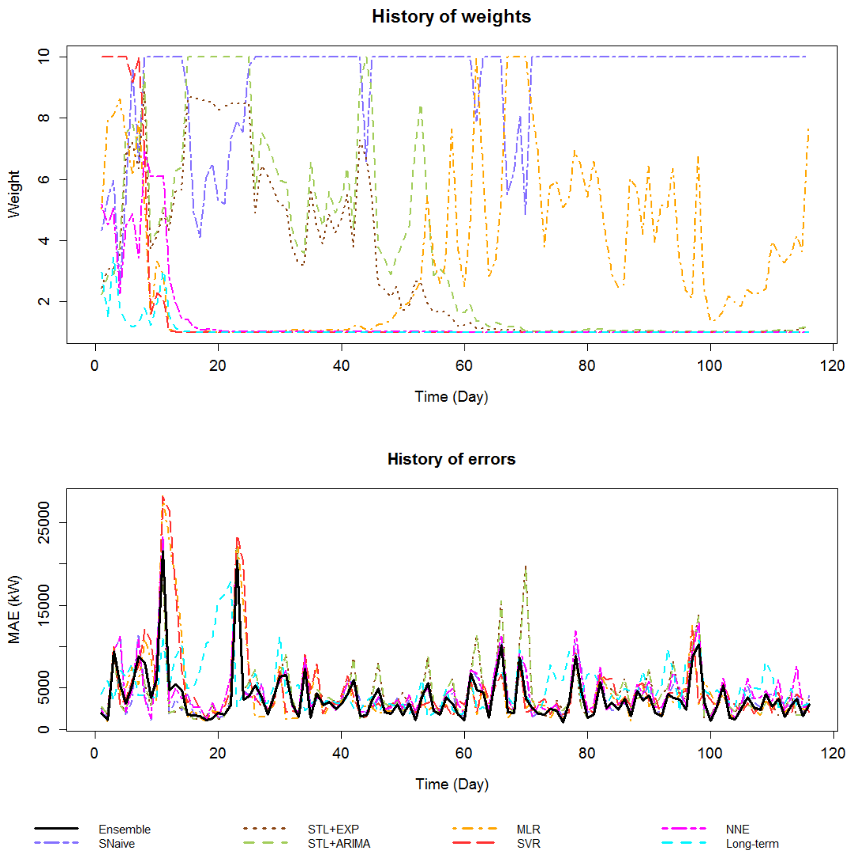


Figure 5

History of ensemble weights and prediction errors for Bratislava region. The legend belongs to both plots. The results of ensemble and following models are shown: seasonal naïve method-random walk (SNaive), seasonal decomposition of time series by loess plus Holt-Winters exponential smoothing (STL+EXP), seasonal decomposition of time series by loess plus ARIMA (STL+ARIMA), multiple linear regression (MLR), support vector regression (SVR), feed-forward neural networks (NNE) and naïve average long-term method (Long-term).

Table 3
Average and standard deviation sMAPE (%) of methods, part 1.
The best base method and the ensemble are highlighted.

Method	Bratislava	Záhorie	Košice	Piešťany	Dunajská Streda
AR	2.328 ± 1.45	2.484 ± 1.75	2.195 ± 1.30	1.876 ± 1.67	2.485 ± 1.58
HW	1.901 ± 1.46	2.744 ± 2.35	2.145 ± 1.59	1.892 ± 2.06	2.342 ± 1.83
STL+EXP	1.663 ± 1.54	2.537 ± 2.71	1.905 ± 1.62	1.759 ± 2.09	2.159 ± 1.86
STL+ARIMA	1.653 ± 1.53	2.433 ± 2.66	**1.833** ± 1.60	1.666 ± 2.17	**2.111** ± 1.79
NNE	1.695 ± 1.36	2.136 ± 1.88	1.985 ± 1.24	1.582 ± 1.73	2.325 ± 1.58
SNaive	**1.561** ± 1.36	2.090 ± 1.92	1.912 ± 1.27	**1.554** ± 1.72	2.299 ± 1.59
MLR	1.652 ± 1.85	1.994 ± 1.79	1.845 ± 1.34	1.696 ± 1.72	2.166 ± 1.69
SVR	1.773 ± 1.92	**1.948** ± 1.80	1.902 ± 1.37	1.656 ± 1.63	2.278 ± 1.76
TBATS	2.581 ± 2.43	2.724 ± 2.73	2.157 ± 1.56	2.791 ± 2.46	5.091 ± 4.84
Float mean	1.939 ± 1.30	1.789 ± 1.34	1.806 ± 1.17	1.730 ± 1.47	2.377 ± 1.56
Float med	2.627 ± 1.46	1.912 ± 1.42	1.907 ± 1.20	1.807 ± 1.52	2.441 ± 1.55
Ensemble	**1.417** ± 1.26	**1.796** ± 1.64	**1.643** ± 1.30	**1.446** ± 1.65	**1.899** ± 1.50

Table 4
Average and standard deviation sMAPE (%) of methods, part 2.
The best base method and the ensemble are highlighted.

Method	Partizánske	Prešov	Poprad	Trebišov	Veľký Krtíš
AR	2.351 ± 1.21	2.512 ± 0.74	3.005 ± 2.03	2.221 ± 1.07	5.453 ± 2.46
HW	2.837 ± 2.10	2.145 ± 1.21	2.927 ± 2.22	2.316 ± 1.56	6.983 ± 4.00
STL+EXP	2.584 ± 2.63	1.969 ± 1.28	2.729 ± 2.50	2.158 ± 1.63	5.962 ± 3.78
STL+ARIMA	2.367 ± 2.40	1.853 ± 1.15	2.487 ± 2.31	2.054 ± 1.52	**5.712** ± 3.49
NNE	2.004 ± 1.43	2.077 ± 0.91	2.402 ± 1.75	2.079 ± 1.20	6.709 ± 3.30

SNaive	1.941 ± 1.49	1.870 ± 0.95	**2.076** ± 1.74	2.006 ± 1.23	6.781 ± 3.36
MLR	1.766 ± 1.33	**1.568** ± 0.73	2.301 ± 2.53	**1.745** ± 0.93	5.658 ± 2.47
SVR	**1.806** ± 1.35	1.742 ± 0.75	2.408 ± 2.68	1.823 ± 0.92	5.523 ± 2.72
TBATS	5.017 ± 2.70	2.009 ± 0.90	3.544 ± 4.27	2.641 ± 1.86	5.621 ± 3.36
Float mean	1.723 ± 1.16	1.978 ± 0.75	2.565 ± 1.27	2.250 ± 1.82	6.769 ± 2.48
Float med	1.794 ± 1.18	2.060 ± 0.79	3.263 ± 1.73	2.296 ± 1.68	6.600 ± 3.17
Ensemble	**1.704** ± 1.43	**1.483** ± 0.73	**1.973** ± 1.74	**1.656** ± 1.05	**5.224** ± 2.67

Table 5

P-values for each region. The best base method compared to the ensemble method is in parentheses

Region	MAE	sMAPE	RMSE
Bratislava (SNaive)	$1.4*10^{-6}$	$1.5*10^{-6}$	$1.3*10^{-7}$
Záhorie (SVR)	0.0284	0.0267	0.0021
Košice (STL+ARIMA)	$2.8*10^{-7}$	$8.0*10^{-7}$	$3.8*10^{-8}$
Piešťany (SNaive)	$1.6*10^{-4}$	$1.0*10^{-4}$	$1.6*10^{-6}$
Dunajská Streda (STL+ARIMA)	0.0001	0.0001	0.0001
Partizánske (SVR)	0.0205	0.0132	0.0015
Prešov (MLR)	0.0046	0.0018	0.0153
Poprad (SNaive)	$2.2*10^{-4}$	$2.0*10^{-4}$	$2.7*10^{-6}$
Trebišov (MLR)	0.0416	0.0324	0.0565
Veľký Krtíš (STL+ARIMA)	0.0388	0.0411	0.0110

We have used sMAPE measure because we tested our methods for single delivery point predictions, too. Single delivery points, in general, have day parts with zero consumption where MAPE evaluation fails. Comparison with other works dealing with power consumption prediction is difficult because in our work we were strongly focused on predictions during concept drifts. This is why direct error evaluation comparison is not possible. Despite it, we present some recent works of load forecasting, and try to compare them to our method.

He et al. [18] used SARIMA models to forecast the electricity demand in China. They forecasted hourly and quarter-hourly demand for next few days ahead. The MAPE error of the models was about 1.5 %. Trained models were validated on real data.

Xiao et al. [50] presented ensemble learning method for a day-ahead consumption prediction. A cuckoo search algorithm was used to find the optimal weights for combining four forecasting models. Models were based on different types of neural networks (namely BPNN, GABPNN, GRNN and RBFNN). Half-hourly load data of February 2006 – 2009 in New South Wales in Australia were used for verification. The forecasts of the ensemble model were significantly better in comparison with the results of the individual models. The average MAPE was approximately 1.3%.

Taylor and McSharry [44] presented an empirical study of various short-term load forecasting methods, i.e. ARIMA model; periodic AR model; an extension for double seasonality of Holt-Winters exponential smoothing; an alternative exponential smoothing formulation; and a method based on the principal component analysis (PCA) of the daily demand profiles. Selected methods were evaluated on half-hourly and hourly load data from 10 European countries. The evaluation showed that the Holt-Winters smoothing provided the best average daily MAPE (ca 1.5%).

Presented works reach MAPE around 1.5%, some of them are using forecasting methods, which are used as base methods in our ensemble model. Forasmuch as our ensemble model delivers better results than single base methods, we can assume that it would deliver better results on presented works' datasets.

Conclusion

In this paper, we propose the *incremental heterogeneous ensemble model for time series prediction*. The model was designed to make predictions for time series with specific properties (strong seasonal dependence and concept drift) in the domain of energy consumption. Its characteristics – robustness, natural ability to parallelize and the ability to incrementally train the model – make the presented ensemble suitable for processing streams of data in a "big data" environment. The achieved results lead us to assume that the presented approach could be a prospective direction in the choice of prediction models for time series with particular characteristics.

In future work, we plan to incorporate dependencies into the model with external factors such as meteorological data and information about holidays in big enterprises in the different regions. Another interesting idea is to investigate possible correlations between different regions. These aspects should also improve the predictions.

Acknowledgement

This work was partially supported by the Research and Development Operational Programme as part of the project "International Centre of Excellence for Research of Intelligent and Secure Information-Communication Technologies and Systems", ITMS 26240120039, co-funded by the ERDF and the Scientific Grant Agency of The Slovak Republic, grant No. VG 1/0752/14.

References

[1] A. S. Alfuhaid and M. A. El-Sayed, "Cascaded Artificial Neural Network for Short-Term Load Forecasting," *IEEE Trans. Power Syst.*, Vol. 12, No. 4, pp. 1524-1529, 1997

[2] G. R. Arce, *Nonlinear Signal Processing: A Statistical Approach*. New Jersey, USA: Wiley, 2005

[3] A. Benczúr, "The Evolution of Human Communication and the Information Revolution — A Mathematical Perspective," *Mathematical and Computer Modelling*, Vol. 38, No. 7-9, pp. 691-708, 2003

[4] G. E. P. Box and D. R. Cox, "An Analysis of Transformations," *J. Roy. Statistical Soc. Series B*, Vol. 26, No. 2, pp. 211-252, 1964

[5] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco, CA: Holden-Day, 1970

[6] L. Breiman, "Bagging Predictors," *Mach. Learning*, Vol. 24, No. 2, pp. 123-140, 1996

[7] A. Chitra and S. Uma, "An Ensemble Model of Multiple Classifiers for Time Series Prediction," *Int. J. Comput. Theory and Eng.*, Vol. 2, No. 3, pp. 454-458, 2010

[8] R. B. Cleveland *et al.*: "Seasonal-Trend Decomposition Procedure based on LOESS," *J. Official Stat.*, Vol. 6, pp. 3-73, 1990

[9] W. Constantine and D. Percival. (2015, February 20). *Package 'wmtsa'* [Online]. Available: http://cran.r-project.org/web/packages/wmtsa/

[10] A. M. De Livera *et al.*, "Forecasting Time Series with Complex Seasonal Patterns Using Exponential Smoothing," *J. American Statistical Assoc.*, Vol. 106, No. 496, pp. 1513-1527, 2011

[11] T. G. Dietterich, "Machine Learning Research: Four Current Directions," *Artificial Intell.*, Vol. 18, No. 4, pp. 97-136, 1997

[12] T. G. Dietterich, "An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization," *Mach. Learning*, Vol. 40, No. 2, pp. 139-157, 2000

[13] Y. Freund and R. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *J. Comput. and System Sciences*, Vol. 55, No. 1, pp. 119-139, 1997

[14] J. Gama, I. *et al.*, "A Survey on Concept Drift Adaptation," *ACM Comput. Surv.*, Vol. 46, No. 4, pp. 1-37, Mar. 2014

[15] J. Gama *et al.*, "Learning with Drift Detection," in *Advances in Artificial Intelligence – SBIA 2004*, LNCS 3171, Springer, pp. 286-295, 2004

[16] J. Gao *et al.*, “On Appropriate Assumptions to Mine Data Streams: Analysis and Practice,” in *7th IEEE Int. Conf. Data Mining*, 2007, pp. 143-152

[17] L. Guo *et al.*, “An Incremental Extreme Learning Machine for Online Sequential Learning Problems,” *Neurocomputing*, Vol. 128, pp. 50-58, 2014

[18] H. He, T. Liu, R. Chen, Y. Xiao, and J. Yang, “High Frequency Short-Term Demand Forecasting Model for Distribution Power Grid based on ARIMA,” in *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, 2012, Vol. 3, pp. 293-297

[19] M. Hollander *et al.*, *Nonparametric Statistical Methods*. Hoboken, NJ: J. Wiley & Sons, 2014

[20] C. C. Holt, “Forecasting Trends and Seasonals by Exponentially Weighted Moving Averages,” *Office of Naval Research Memorandum*, Vol. 52, 1957

[21] W. C. Hong, *Intelligent Energy Demand Forecasting*. London: Springer-Verlag, 2013

[22] K. Hornik *et al.*, “Multilayer Feedforward Networks are Universal Approximators,” *Neural Networks*, Vol. 2, No. 5, pp. 359-366, 1989

[23] R. J. Hyndman *et al.* (2015, February 26). *Package ‘forecast’* [Online]. Available: http://cran.r-project.org/web/packages/forecast/forecast.pdf

[24] E. Ikonomovska *et al.*, “Learning Model Trees from Evolving Data Streams,” *Data Mining and Knowledge Discovery*, Vol. 23, No. 1, pp. 128-168, 2011

[25] A. Karatzoglou *et al.*, “kernlab - An S4 Package for Kernel Methods in R,” *J. Statistical Software*, Vol. 11, No. 9, pp. 1-20, 2004

[26] R. Klinkenberg and T. Joachims, “Detecting Concept Drift with Support Vector Machines,” pp. 487-494, Jun. 2000

[27] J. Z. Kolter and M. A. Maloof, “Dynamic Weighted Majority: An Ensemble Method for Drifting Concepts,” *J. Mach. Learning Research*, Vol. 8, pp. 2755-2790, 2007

[28] L. I. Kuncheva and C. J. Whitaker, “Measures of Diversity in Classifier Ensembles and their Relationship with the Ensemble Accuracy,” *Mach. Learning*, Vol. 51, No. 2, pp. 181-207, 2003

[29] N. Liu *et al.*, “Short-Term Forecasting of Temperature driven Electricity Load using Time Series and Neural Network Model,” *J. Clean Energy Technologies*, Vol. 2, No. 4, pp. 327-331, 2014

[30] J. Mendes-Moreira *et al.*, “Ensemble Approaches for Regression: A Survey,” *ACM Computing Surveys*, Vol. 45, No. 1, Article 10, 2012

[31] C. J. Merz, “Classification and Regression by Combining Models,” Ph.D. dissertation, University of California, USA, 1998

[32] L. L. Minku, “Online Ensemble Learning in the Presence of Concept Drift,” Ph.D. dissertation, University of Birmingham, UK, 2011

[33] I. Moghram and S. Rahman, “Analysis and Evaluation of Five Short-Term Load Forecasting Techniques,” *IEEE Trans. Power Syst.*, Vol. 4, No. 4, pp. 1484-1491, 1989

[34] L. Moreira-Matias *et al.*, “On Predicting the Taxi-Passenger Demand: A Real-Time Approach,” in *Progress in Artificial Intelligence*, LNCS 8154, Springer, pp. 54-65, 2013

[35] A. Narasimhamurthy and L. I. Kuncheva, “A Framework for Generating Data to Simulate Changing Environments,” in *25th IASTED Int. Multi-Conf. Artificial Intell. and Applicat.*, Innsbruck, Austria, 2007, pp. 384-389

[36] R. Nau (2015, February 28) *Moving Average and Exponential Smoothing Models* [Online] Available: http://people.duke.edu/~rnau/411avg.htm

[37] N. C. Oza, and S. Russell, “Experimental Comparisons of Online and Batch Versions of Bagging and Boosting,” in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Disc. and Data Mining*, San Francisco, CA, USA, 2001, pp. 359-364

[38] N. C. Oza and S. Russell, “Online Bagging and Boosting,” in *IEEE Int. Conf. Syst., Man and Cybern.*, New Jersey, USA, 2005, pp. 2340-2345

[39] R. Polikar *et al.*, “Learn++: An Incremental Learning Algorithm for Supervised Neural Networks,” *IEEE Trans. Syst., Man, and Cybern. Part C*, Vol. 31, No. 4, pp. 497-508, 2001

[40] S. Reid, *A Review of Heterogeneous Ensemble Methods*. University of Colorado at Boulder: Department of Computer Science, 2007

[41] S. S. Shapiro and M. B. Wilk, “An Analysis of Variance Test for Normality (complete samples),” *Biometrika*, Vol. 52, No. 3-4, pp. 591-611, 1965

[42] W. Shen *et al.*, “Ensemble Model for Day-Ahead Electricity Demand Time Series Forecasting,” in *Proc. 4th Int. Conf. Future Energy Syst.*, Berkeley, CA, USA, 2013, pp. 51-62

[43] Md. Shiblee *et al.*, “Time Series Prediction with Multilayer Perceptron (MLP): A New Generalized Error-based Approach,” *Advances in Neuro-Information Processing*, LNCS 5507, Springer, pp. 37-44, 2009

[44] J. W. Taylor and P. E. McSharry, “Short-Term Load Forecasting Methods: An Evaluation Based on European Data,” *IEEE Trans. Power Syst.*, Vol. 22, No. 4, pp. 2213-2219, Nov. 2007

[45] G. K. F. Tso and K. K. W. Yau, “Predicting Electricity Energy Consumption: A Comparison of Regression Analysis, Decision Tree and Neural Networks,” *Energy*, Vol. 32, No. 9, pp. 1761-1768, 2007

[46] H. Wang *et al.*, “Mining Concept-Drifting Data Streams using Ensemble Classifiers,” in *Proc. 9th ACM Int. Conf. Knowledge Discovery and Data Mining (KDD’03)*, Washington, DC, USA, 2003, pp. 226-235

[47] J. Wichard and M. Ogorzałek, “Time Series Prediction with Ensemble Models,” in *Proc. Int. Joined Conf. Neural Networks*, Budapest, Hungary, 2004, pp. 1625-1630

[48] G. Widmer and M. Kubat, “Learning in the Presence of Concept Drift and Hidden Contexts,” *Mach. Learn.*, Vol. 23, No. 1, pp. 69-101, Apr. 1996

[49] P. R. Winters, “Forecasting Sales by Exponentially Weighted Moving Averages,” *Management Science*, Vol. 6, No. 3, pp. 324-342, 1960

[50] L. Xiao *et al.*, “A Combined Model based on Data Pre-Analysis and Weight Coefficients Optimization for Electrical Load Forecasting,” *Energy*, Vol. 82, pp. 524-549, 2015

[51] W. Xie *et al.*, “Incremental Learning with Support Vector Data Description,” in *2014 22nd Int. Conf. Pattern Recognition (ICPR)*, 2014, pp. 3904-3909

[52] W. Zang *et al.*, “Comparative Study between Incremental and Ensemble Learning on Data Streams: Case Study,” *J. Big Data*, Vol. 1, No. 1, 2014

[53] F. Zhang *et al.*, “Conjunction Method of Wavelet Transform-Particle Swarm Optimization-Support Vector Machine for Streamflow Forecasting,” J. Appl. Math., Vol. 2014, article ID 910196, 2014

[54] P. Zhang *et al.*, “Categorizing and Mining Concept Drifting Data Streams,” in *Proc. 14th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Las Vegas, NV, USA, 2008, pp. 820-821

[55] P. Zhang *et al.*, “Robust Ensemble Learning for Mining Noisy Data Streams,” *Decision Support Syst.*, Vol. 50, No. 2, pp. 469-479, 2011

Adaptive Flow QoS Management Model for Wireless Communication in Mobile Environments

Taeyoung Kim[1], Youngshin Han[2], Jaekwon Kim[1], Jongsik Lee[1]

[1]Department of Computer and Information Engineering, INHA University, Seoul 402-751, Inha-ro 100, Nam-gu, Incheon, Republic of Korea, e-mail: taeyoung.kim@selab.inha.ac.kr, jaekwonkorea@selab.inha.ac.kr, jslee@inha.ac.kr

[2]Department of Computer Engineering, Sungkyul University, Seoul 430-742, Syunkyuldaehak-ro 53, Manan-gu, Anyang, Kyungi-do, Republic of Korea, e-mail: hanys@sungkyul.ac.kr

Abstract: Mobile environments are based on wireless communication, and wireless networks that provide communication services via radio signals. Although both mobile and wireless systems may be free from space constraints, they suffer from certain unstable characteristics. These problems can be compensated by applying some countermeasures to the communication environment. This paper presents an adaptive communication management model based on fuzzy logic. The proposed model includes an estimation module to control the flow throughput, and adopts a policy of providing greater benefits to better links. In addition, the model includes tuned snooping and retransmission schemes to ensure the quality-of-service of wireless communication. Simulation results verify the efficiency of the proposed model.

Keywords: Adaptive Flow Management; Fuzzy Logic; Mobile Wireless Network

1 Introduction

Information technology has advanced rapidly over the past two decades, with the most notable advances in the field of mobile and wireless technology [1]. In the past, a handheld device was a simple tool that performed a specific function. In contrast, current devices are equivalent to a small computer, and mobile users are able to access a range of services at any time [2]. Furthermore, wireless technology can maximize the functionality of mobile devices. A wireless network uses radio signals for communication, and devices can access the network from

[*] Corresponding Author. E-mail: hanys@sungkyul.ac.kr, jslee@inha.ac.kr

anywhere provided they can receive a sufficient signal. Thus, mobile and wireless technologies enable users to overcome the limitations of time and space [2, 3]. However, there are several problems that users cannot recognize. Most of these problems are related to wireless communication. A wireless signal is a radio wave transmitted from a source to the atmosphere. The quality of the received signal varies with the distance from the source. Moreover, the signal quality is significantly influenced by the surroundings. To address these problems, a supplementary method is required.

Many studies have attempted to ensure quality-of-service (QoS) in wireless networks. Most of these studies have focused on improving the communication protocol because conventional protocols are designed for wired environments. For example, the Transport Communication Protocol (TCP) has been used to ensure QoS in communication networks. However, some TCP functions may cause performance degradation in a wireless environment. This problem arises from the differences between wired and wireless environments [4]. Thus, various studies have attempted to improve the protocol from the viewpoint of wireless networks. A number of researchers have focused on the protocol itself, whereas others have attempted to develop a supplementary method [5-9]. Although existing methods have attempted to resolve the QoS issue for wireless networks, further investigation is required to improve wireless QoS.

In this paper, we propose an adaptive QoS management model for wireless communication that is based on fuzzy logic [10] and packet snooping [8, 9]. The snooping method includes packet storage and local retransmission schemes. Both schemes supplement the communication protocol to ensure packet delivery in a wireless network. However, the conventional method always applies the snooping scheme under the same conditions. Such fairness leads to a waste of resources while providing average performance. Therefore, we add a fuzzy-based scheme to estimate the link state. The state estimation serves as the basis for determining flow QoS management. When a link is in a good state, it consumes additional resources in the process of applying the scheme. In other words, the proposed model performs adaptive flow management according to the link state. This adaptive scheme allows for more efficient resource utilization in the snooping process.

The remainder of this paper is organized as follows. Section 2 briefly reviews related studies, before Section 3 describes the key concept of our fuzzy-based adaptive scheme. Section 4 discusses the simulation design and presents experimental simulation results. Finally, Section 5 summarizes our findings and concludes the paper.

2 Related Work

Reliability and QoS are the biggest problems in wireless communication, and many studies have proposed techniques and methods to overcome these issues. The focus of most studies is to improve the communication scheme in the protocol via protocol-based supplements or supporting methods. The former studies focus on the problem of conventional TCP in wireless communication, whereas the latter emphasize supplementary measures to control the wireless communication or network. In this section, we survey existing studies on wireless TCP and fuzzy-based QoS management.

2.1 TCP in Wireless Networks

TCP enables reliable communication over a wired network. However, the error occurrence probability in a wireless environment is higher than that in a wired environment. In a wireless environment, the TCP scheme degrades the communication QoS, because it repeats the entire transmission flow to recover from an error. Most communication lines are based on wired networks, whereas wireless communication only occurs between a device and an access point in a wired network. Therefore, most existing studies have focused on improving the TCP scheme for wireless networks.

The simplest solution is to use the TCP control packet. A mobile device is connected to a wired network via a base station. Thus, the base station can recognize the communication state of the mobile device. Therefore, it is possible to deliver a failure notification to the fixed host in the control scheme. This approach is referred to as mobile TCP (M-TCP). M-TCP uses a control packet to notify the network of a transmission failure. The use of control packets leads to increased network traffic [4]. To overcome this problem, a method that divides the flow control scheme has been proposed. In this approach, which is referred to as an indirect TCP strategy, the base station separately communicates with a fixed host and a mobile host. A failure in the wireless network can be resolved between the base station and the mobile host. The fixed host is only involved in the recovery process in the event of total transmission failure at the base station [6]. However, this indirect strategy may reduce efficiency at the base station because the base station is responsible for controlling the flow toward mobile devices. Therefore, an efficient control strategy should be established at the base station.

A snooping TCP mechanism is a control technique that can be adopted by indirect strategies to monitor packets in the network. For this purpose, the snooping module can capture and analyze packets to supplement wireless flow control. The base station can determine a packet's destination using the snooping module. Moreover, the snooping module creates the opportunity of saving the packet through the capture function. Thus, the base station gains the ability to control the packet flow. In the event of a transmission failure, the base station may locally attempt to resend the stored packet. The base station may also adjust the transmission rate of packets according to the communication state. Therefore, the design of the snooping policy is the most important factor in improving the QoS [8, 9].

Existing studies have focused on improving the snooping policy. The objective of the present study is to establish an adaptive snooping policy based on fuzzy logic in order to impart greater flexibility to the snooping technique. Consequently, we can improve wireless QoS.

2.2 QoS Management with Fuzzy Logic

In engineering, classical logic uses values of 0 (false) or 1 (true). With fuzzy logic, however, the truth is a multi-valued state ranging from 0 to 1 [11]. Parameters in fuzzy logic are indicated as a degree by a linguistic expression. The fuzzy system includes specific functions to manage these linguistic values, and also contains a rule-based engine to estimate the output result. These features are useful in determining the system control with complicated and uncertain conditions [10]. Thus, most researchers have applied fuzzy-based methods to QoS management in wireless environments [11-15].

In wireless sensor networks (WSNs), QoS is influenced by various factors [11]. Thus, the QoS management system should identify essential factors and deal with them. WSNs are distributed networks that consist of a number of battery-powered sensors. Energy efficiency is the most essential issue in WSNs. Thus, QoS management has focused on improving throughput and traffic control, and fuzzy logic has been used for congestion estimation [12], output QoS determination [11], and traffic estimation [13]. Each method handles QoS by regulating the packet generation rate, node transmission power, and traffic congestion [11-13].

Mobile (or wireless) ad hoc networks (MANETs) must consider routing and scheduling issues, because these are closely related to QoS requirements. Alternative methods that control the packet rate and flow admission also exist. Under fuzzy logic, rate control allows every node to regulate the traffic, which may have a positive effect on QoS management. The role of fuzzy logic is to determine the regulation rate based on measurements of traffic delay [14].

The consideration of QoS in wireless networks (WNs) is similar to that in both MANETs and WSNs. However, the presence of access points (APs) makes QoS management for WNs slightly different. In WNs, QoS is managed by whichever approach is associated with the AP, because the AP supervises all communication between the mobile and fixed nodes. Load balancing can help improve the QoS in WNs. The fuzzy logic may be notified that the mobile node's AP has changed using measured factors such as signal quality or transmission failure [15].

As described above, fuzzy-based approaches are widely used to overcome QoS management issues. Therefore, the present study adopts fuzzy logic to regulate the snooping scheme.

3 Adaptive Flow QoS Management

In this paper, we propose an adaptive management system to ensure the flow QoS in WNs. The proposed model is based on the snooping mechanism for TCP communication, and adopts fuzzy logic to control the communication flow. Figure 1 shows a schematic design of the proposed model.

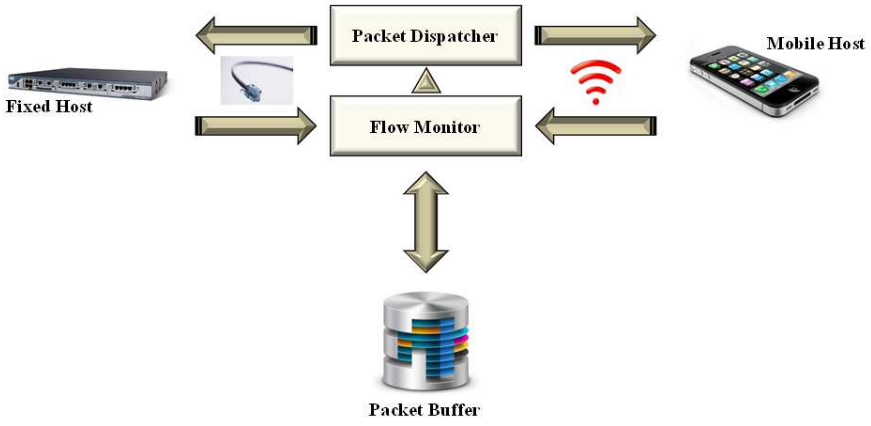


Figure 1
Schematic design of the proposed model

Packets from each host are collected by a flow monitor in the base station. The flow monitor includes two schemes for adaptive control.

The first scheme estimates the link state using fuzzy logic. The flow monitor must obtain input values for this scheme. This is relatively simple because all packets are directed to their destination via the flow monitor. Thus, the flow monitor can measure statistical information for all links in real time. However, fuzzy-based estimation is performed at regular intervals, and real-time estimation can adversely affect the overall efficiency. For instance, assume that the environment of a link changes rapidly. Under real-time estimation, the link state will be updated to reflect these changes. Both estimation and state management are service features of a base station, and so frequent state updates will increase the base station’s overhead. Thus, the state estimation should only identify trends in the link state.

The second scheme adjusts the snooping and transmission cycles on the basis of the estimated link state. The snooping TCP mechanism uses a buffer to store intercepted packets. The snooping module sends the packet to the mobile host, and stores it for local retransmission in the event of a communication failure. The proposed model uses this behavior to handle the transmission cycle. Figure 2 illustrates the logic of the snooping module in the proposed model.

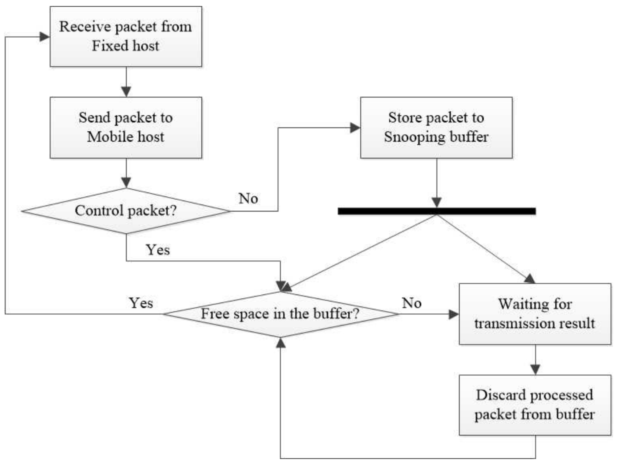

Receive packet from Fixed host
Send packet to Mobile host
Store packet to Snooping buffer
Control packet?
No
Yes
Yes
Free space in the buffer?
No
Waiting for transmission result
Discard processed packet from buffer

Figure 2
Process diagram of the snooping module in the proposed model

The key factor in this process is the presence of the free space in the buffer. Extra buffer space can provide additional benefits to the snooping process because the module can request packets to fill the buffer. Using this behavior, the proposed model can adjust the transmission cycle. The main concept of the adjustment method is to control the probability of free space existing in the allocated buffer. This probability is directly proportional to both the degree of the link state and the buffer size. The link state is related to the speed with which packets are discarded from the buffer. Thus, links in a better state are likely to have more free space in the allocated buffer. The buffer size is associated with the number of packets that can be imported from the fixed host. If packets continue to be discarded from the buffer, the increased capacity can provide more space to the link. Ideally, the link state and the buffer size will be regulated at the same time. However, the link state has variable characteristics that cannot be controlled directly by this module. Therefore, the proposed model adjusts the buffer size according to the link state (Figure 3).

The link state is estimated using fuzzy logic. An increase in the buffer size indicates an improvement in the link state. Thus, the flow module may request additional packets owing to the increased capacity. In contrast, a decrease in the buffer size indicates some deterioration in the link state. In this case, the buffer may overflow. In general, the buffer discards packets that overflow from the memory. However, in our design, all packets are retained regardless of the overflow condition of the buffer. In the proposed model, the buffer is an area for storing packets that have been intercepted in the transmission flow. Packets in the buffer may be discarded in accordance with the results of wireless communication. Because of this design, the buffer size must be adjusted carefully.

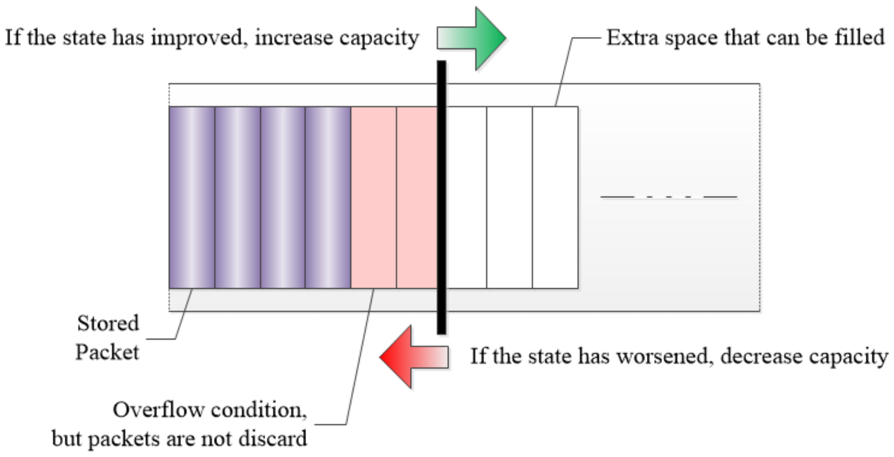

If the state has improved, increase capacity
Extra space that can be filled
Stored Packet
If the state has worsened, decrease capacity
Overflow condition, but packets are not discard

Figure 3
Adaptive buffer management based on link state

Further details of both schemes are provided in the following subsections.

3.1 Estimation of Link State using Fuzzy Logic

Various components of a communication link can be measured. However, it is very difficult to define the link state using the measured values. The link state can be quantified by a performance evaluation. For instance, it is easy to understand the link speed in terms of values such as 10 Mbps. However, whether this speed is fast or slow depends on the evaluation criteria. The measured values provide clarity in the form of crisp values, but the evaluation is subject to bias. Therefore, the control module requires a proper method to estimate the link state. Fuzzy logic has been widely used to overcome such uncertainty, as discussed in Section 2.2.

Figure 4 shows the design of our fuzzy module for estimating the link state. We have a total of four input variables: distance and signal strength, which are physical factors, and loss rate and timeout, which are logical factors. Physical factors are significantly influenced by user behavior and user circumstances. In contrast, the logical factors are influenced by the type and manner of communication. Each input element is assumed to be measured by the base station. The fuzzy module converts the received input parameters into fuzzy values, and the inference engine then determines the output according to the rules on the basis of the converted input. Finally, the output is converted into a crisp value via defuzzification.

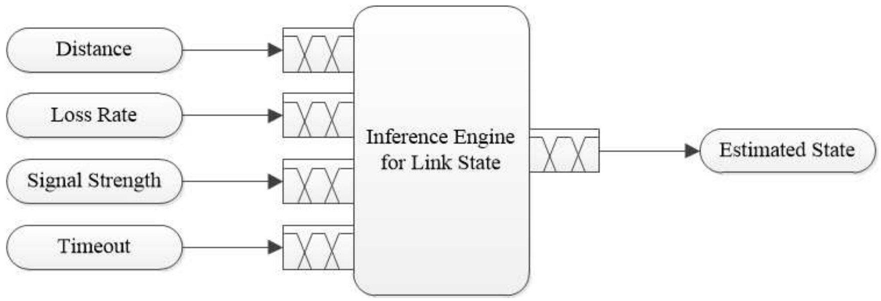


Figure 4
Fuzzy module for estimating link state

Table 1
Fuzzy input and output parameters

Parameter Name	Fuzzy Set
Distance	{Near, Middle, Far }
Loss Rate	{Very Small, Small, Large, Very Large}
Signal Strength	{Negative, Unstable, Normal, Stable}
Timeout	{Very Short, Short, Long, Very Long}
Estimated State	{Disappointing, Questionable, Acceptable, Excellent}

Table 1, summarizes the fuzzy input and output parameters. Each input parameter is converted into terms defined in the corresponding fuzzy set by a membership function. The inference engine deduces the output term using defined rules. The fuzzy rule table contains combinations of all input and output terms. Therefore, our inference logic is based on a total of 192 individual rules. Some characteristic samples of these rules are listed in Table 2. The samples show the effect of each input variable on the output. In the proposed design, the logical factors have a greater effect on the output than the physical factors. This design is based on the influence that each factor has on the communication QoS. Physical factors are relatively easy to control, i.e., users can move closer to the base station if required, and the signal strength can be adjusted (not recommended for power saving and safety purposes). However, it is difficult to control the logical factors. The result generated by the inference engine is converted into a crisp value by a membership function. This output is the estimated link state. The membership functions used in the above-mentioned processes are illustrated in Figure 5. There are various methods for obtaining the inferred result and its crisp output. We apply the most popular method used in fuzzy-based studies, i.e., the rule-based reasoning of Mamdani's min-max method. The defuzzification result is obtained using the center of gravity [16].

Table 2

Samples of defined rules

Rule #	uA	uB	uC	uD	uE
#5	Near	Very Small	Unstable	Very Short	Excellent
#21	Near	Small	Unstable	Very Short	Acceptable
#24	Near	Large	Unstable	Long	Questionable
#65	Middle	Very Small	Negative	Very Short	Acceptable
#69	Middle	Very Small	Unstable	Very Short	Excellent
#89	Middle	Very Small	Normal	Very Short	Acceptable
#120	Middle	Very Large	Unstable	Very Long	Questionable
#125	Middle	Very Large	Stable	Very Short	Acceptable
#184	Far	Very Large	Unstable	Very Long	Disappointing
#185	Far	Very Large	Normal	Very Short	Acceptable

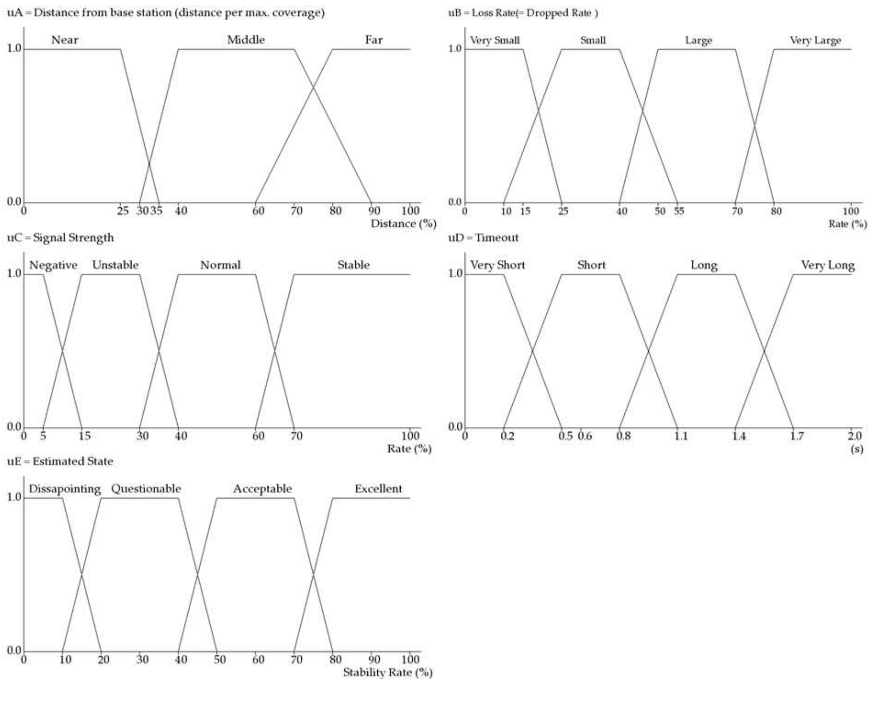


Figure 5

Membership functions for input and output parameters

3.2 Adaptive Control of Snooping Buffer Size

As discussed in Section 2.1, TCP is the most widely used protocol in communication environments. However, conventional TCP is optimized to support communication over-wired networks. A number of researches have suggested variants to extend its applicability, with snooping-based TCP suggested for wireless networks. The key features of the snooping mechanism are the packet capture and analysis functions. Thus, the buffer is the most important resource for the snooping module. This study adopts an adaptive scaling policy for the size of the snooping buffer of each flow. Figure 5 describes the process of buffer size rescaling.

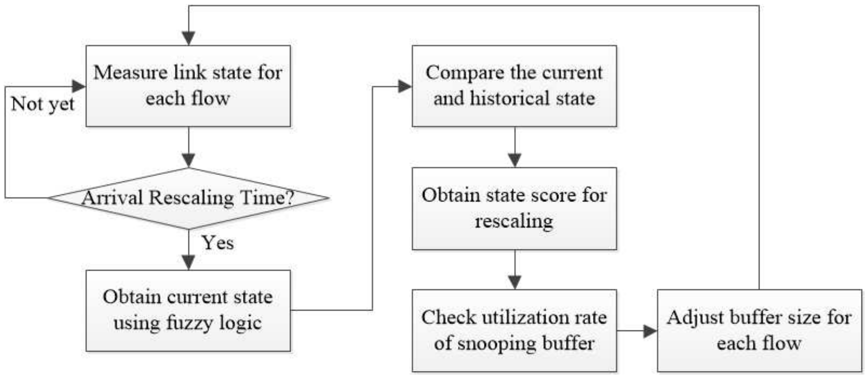
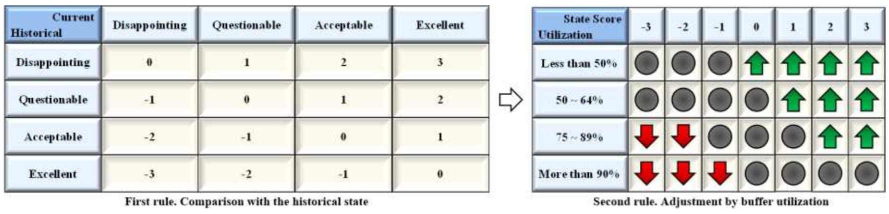

Measure link state for each flow
Not yet
Arrival Rescaling Time?
Yes
Obtain current state using fuzzy logic
Compare the current and historical state
Obtain state score for rescaling
Check utilization rate of snooping buffer
Adjust buffer size for each flow

Figure 5
Process of adaptive buffer rescaling

The process between measuring the link state and obtaining the current state corresponds to the fuzzy-based estimation described in Section 3.1. The rescaling process starts by comparing the current state with the historical state. The historical state is used to identify the trend in state changes. Thus, a range of historical states can be included. In this study, the historical state indicates the center of gravity of the previous three phases. The comparison result is converted to the state score, and this is utilized in the rescaling policy. The policy works to reduce the size of the allocated buffer when the condition continues to worsen, and vice versa.

The snooping buffer is a finite resource with a limited capacity for storage. Thus, the rescaling policy must consider the available capacity and utilization of the snooping buffer. If the buffer utilization is low, the utilization must be increased by allocating more space for flows that are improving. In contrast, additional allocation should be avoided if the buffer has a high utilization rate. Figure 6 presents these rescaling rules in two tables. Each table illustrates the decision-making process of the buffer rescaling for the given conditions.

In Figure 6, we do not specify the magnitude of the change in buffer size, as this can be changed according to the features, circumstances, and conditions of the network. In this study, we apply a small differential to increase or decrease the size depending on the state score and buffer utilization. However, the difference is only a few packets.



Historical \ Current	Disappointing	Questionable	Acceptable	Excellent
Disappointing	0	1	2	3
Questionable	-1	0	1	2
Acceptable	-2	-1	0	1
Excellent	-3	-2	-1	0

First rule. Comparison with the historical state

Utilization \ State Score	-3	-2	-1	0	1	2	3
Less than 50%	●	●	●	⬆	⬆	⬆	⬆
50 ~ 64%	●	●	●	●	⬆	⬆	⬆
75 ~ 89%	⬇	⬇	●	●	●	⬆	⬆
More than 90%	⬇	⬇	⬇	●	●	●	●

Second rule. Adjustment by buffer utilization

Figure 6
Rules for adjusting buffer size

This rescaling policy increases the circulation speed of the buffer. Thus, this scheme provides more space to links that can discard packets rapidly. It is difficult to predict and formalize the change in the link state, so we assume that the pattern of change is similar to a mathematical curve such as a sine or cosine wave. This curve is assumed to reflect the changes in transmission rate. The fixed buffer size can then be represented by the horizontal line. When the curve is located above the buffer size, there has been a failure to take full advantage of the transmission rate. The opposite situation indicates a transmission delay in the buffer, which is referred to as a bottleneck. Delayed packets can be transferred from the buffer when the transmission rate has been restored. This situation leads to an overall delay in transmission. The proposed scheme regulates the buffer size in accordance with this curve. Although these adjustments cannot exactly match the curve, they assist in minimizing inefficient movements.

4 Simulation Design and Results

We conducted simulations to evaluate the effectiveness of the proposed model. The operation of communication can be expressed as a discrete event, and each operation and device has a specific state corresponding to that event. Therefore, we adopted the DEVS methodology [17, 18] for our analysis.

4.1 Simulation Design

Figure 7, shows a schematic of our simulation model. The model was designed according to the configuration shown in Figure 1. Most of the modules shown in Figure 7 were used to simulate the physical device; only the wireless link module was used to simulate the environment. In addition, there was no module for wired links in the overall structure because our study is focused on problems in wireless communication. Thus, we included a separate module to simulate errors occurring in wireless links.

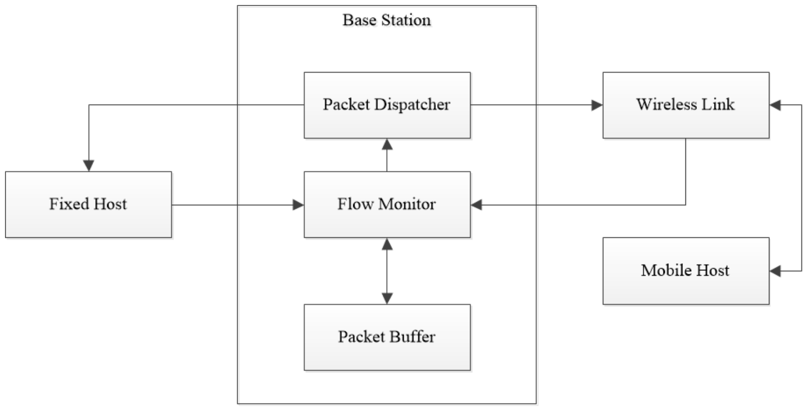


Figure 7
Schematic of wireless flow simulation

The fixed host transmits a packet when the mobile host requests data. In this process, the base station controls the overall flow for packet transmission. The flow monitor plays the most important role in the base station, i.e., managing the packet flow for communication links. For this purpose, the flow monitor receives incoming packets and analyzes the flow. This analysis is based on fuzzy logic, as discussed in Section 3. The estimated state determines the snooping mode to be applied to the link. The base station sends the received packet to the destination host. If the snooping mode is activated, the flow monitor temporarily stores the packet in the buffer. The stored packet is used for local retransmission to correct transmission errors. Therefore, if the acknowledgment packet is received, the base station discards the packet stored in the buffer. The flow monitor then requests the next packet to fill the empty space in the buffer. The mobile host simulates the movement and operation of a handheld device using a map.

Figure 8 shows a virtual load map for simulating the node mobility of the mobile host. Originally, we planned to conduct simulations using a real roadmap. However, we decided to use a virtual map because of problems with the road and base station locations. The map shows a virtual terrain in 7 × 7 grid form, with four base stations. The map has an infinite loop structure, i.e., its opposite ends are connected. All edges represent movement paths of nodes in the grid map. We designed the node mobility model on the basis of the Manhattan model [19, 20]. The dotted circles in the figure represent the coverage area of each base station. The nodes in overlapping coverage areas have to select a base station for communication. A mobile device generally connects to a base station depending on the signal strength. Thus, the handover situation arises when the base station communicating with the node changes. However, this simulation does not consider the handover situation. Therefore, we simplified the event generated at the point of handover. Every node selects a base station that has a clear advantage in terms of signal strength.

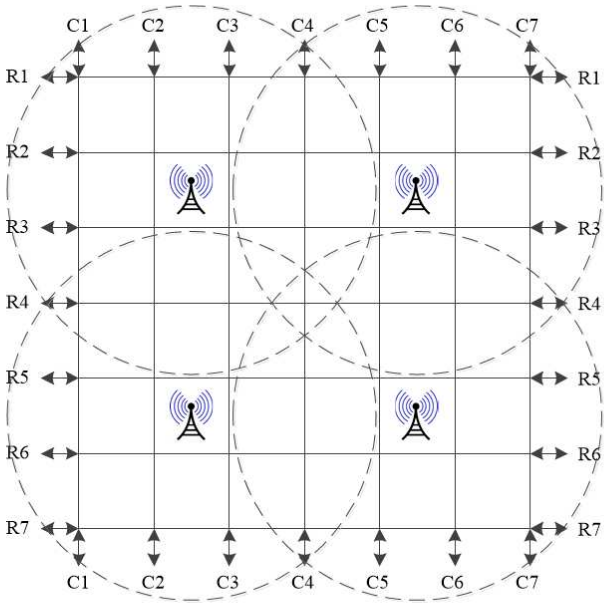


Figure 8
Virtual map for node mobility simulation

The simulation settings were as follows. The fixed host provided all data services from a single module. The base station and the mobile host constituted a single physical module. However, these modules acted as multiple devices through internal objects. The base station module set up each virtual device using four objects. The behavior of the base station was implemented through the method and operation of the objects, whereas handover was implemented as a data exchange between the objects. The mobile host module represented individual nodes through 128 objects. Every node sent a packet and updated its current position. We applied the Manhattan model to implement node mobility. The behavior design for communication was based on data collected from wireless routers. In this process, we also referred to the Wikipedia page traffic statistics [21].

4.2 Results and Discussion

Through our simulations, we compared three metrics for three models. The first model represents normal TCP communication without the snooping scheme (N-TCP) [22, 23]. The second model represents indirect TCP communication with the typical snooping scheme (S-TCP) [8], and the third model represents the snooping TCP mechanism for adaptive flow QoS control (AFQ-TCP). A description and the measurement results of each model are presented in the following subsections.

4.2.1 Total Number of Packets Generated

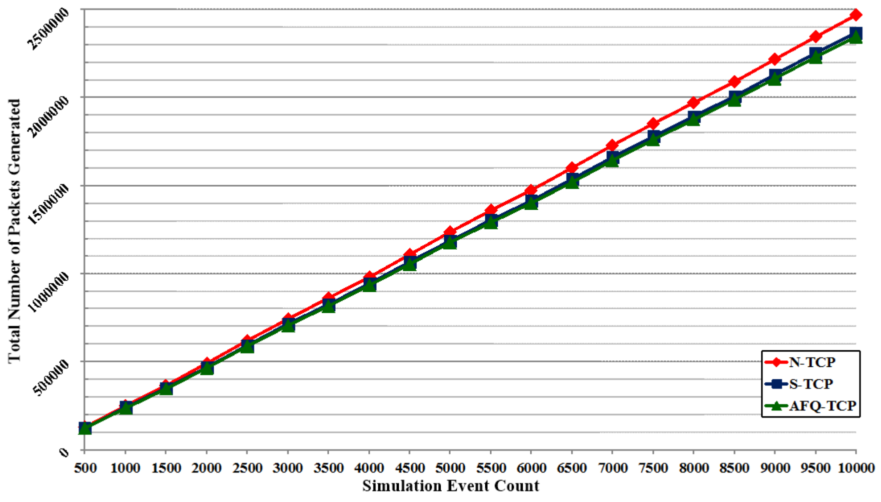


Figure 9
Total number of packets generated

The first metric is the total number of packets generated. A packet is a transmission unit for exchanging data and control messages. The number of packets generated from the same data is closely related to the frequency of error recovery. Therefore, an increase in the number of packets indicates inefficiency in the error recovery scheme. This also degrades the communication efficiency owing to the increase in network traffic. Figure 9 shows the number of packets generated for each model for 10,000 discrete events. The generated packets may include both data and control packets from each model.

As shown in Figure 9, N-TCP produced the largest number of packets because it does not include any scheme for wireless environments. Both snooping-based models used fewer packets than N-TCP. However, the proposed model displayed a slight advantage over S-TCP. Although the proposed model and S-TCP are based on the same snooping technique, the former includes adaptive schemes with fuzzy logic and buffer management; in addition, AFQ-TCP controls the packet flow according to the link state.

A bad link receives fewer opportunities to control packet flow based on buffer size management. Thus, this adaptive scheme is responsible for the difference in performance.

4.2.2 Total Communication Time

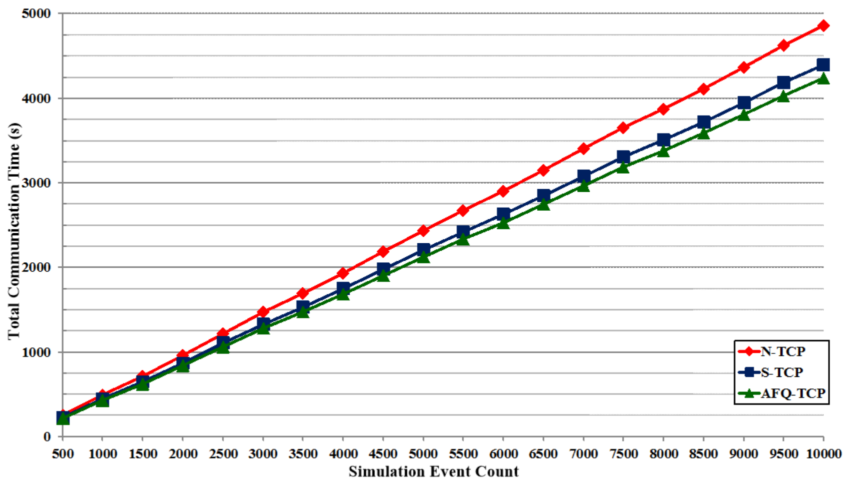


Figure 10
Total communication time

The second metric is the total communication time, i.e., the time taken for data transmission. The total communication time is a measure of the simulation time for each model. We deployed the same dataset and communication behavior for this measurement. The main factors that affect the communication time are packet error and the processing time of the control schemes. Therefore, an increase in the communication time indicates inefficiency in the control scheme. Figure 10 shows the total communication time for each model for 10,000 discrete events. For this measurement, we adjusted the simulation time ratio to be as high as possible. This is because we faced some problems in measuring the runtime without scaling the ratio.

Figure 10 indicates there was a notable gap between the runtimes of the three models as the simulation progressed. This gap is associated with delays due to transmission errors. N-TCP performs a recovery operation from a fixed host to the mobile host to correct transmission errors. In contrast, snooping-based techniques can correct packet errors at the base station. Thus, snooping-based TCP performs faster retransmission than N-TCP.

Our model provides additional communication cycles and resources according to the link state. The base station has limited resources, and S-TCP utilizes the resources evenly for each link in the snooping cycle. Our model does not guarantee a uniform distribution of resources to links. Instead, our scheme speeds up the transfer by focusing on and selecting excellent links. As a result, the communication time of the proposed model was observed to be faster than that of S-TCP.

4.2.3 Throughput

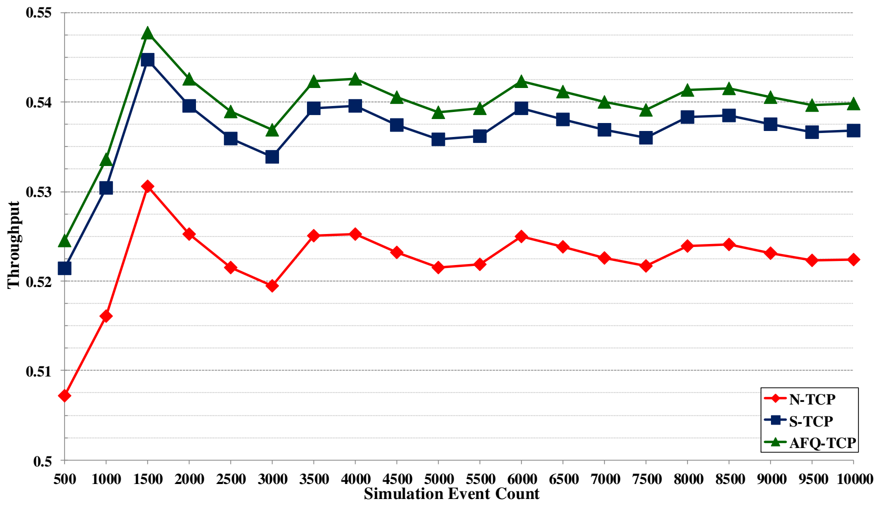


Figure 11
Throughput

$$T_x = \frac{\sum P_a(X)}{\sum P_s(X)}$$
$$T = \frac{\sum P_a}{\sum P_s} \tag{1}$$

The third metric is the throughput. This was calculated using equation (1), where T denotes the throughput, T_x denotes the throughput of the *x*-th node, P_s denotes the total number of packets generated, $P_s(x)$ denotes the P_s of the *x*-th node, P_a denotes the number of successfully delivered packets, and $P_a(x)$ denotes the P_a of the *x*-th node. Thus, the throughput is obtained by dividing the number of successfully delivered packets by the total number of packets generated. Higher throughput can be achieved by using fewer packets for packet transmission, or by reducing packet loss. In other words, high throughput indicates an efficient flow control scheme. Figure 11 shows the throughput for each model for 10,000 discrete events.

As shown in Figure 9, N-TCP generated a greater number of packets than the other models. Therefore, N-TCP exhibited the lowest throughput among the three models. The difference between S-TCP and our model is attributable to their respective snooping policies. The snooping module of S-TCP only supports the enabled or disabled protocol for each link. However, the proposed model regulates the snooping cycle in accordance with the link state. This minor difference has a significant impact on the throughput. Thus, the proposed model displayed a higher throughput.

Conclusions

We have presented a model for adaptive flow QoS management to overcome the problems suffered by conventional networking in wireless environments. Specifically, we have developed a link state estimation method based on fuzzy logic, and designed an adaptive flow control method based on the estimated link state. This approach allows the proposed scheme to increase the communication efficiency by providing greater opportunities to better links. We conducted simulations to evaluate the performance of the proposed model. The simulation results showed that the proposed model improves the efficiency of wireless communication, as it requires fewer packets and less communication time and provides high throughput.

In future work, we will attempt to regulate the proposed scheme. The short-term objective is to design a method for sharing the collected packets between base stations. The present study does not address this issue in detail. Instead, we have simply assumed that all base stations use the virtual shared repository within the simulation constraints. The long-term objective is to conduct experiments using realistic terrain and base station placements.

Acknowledgement

This work was supported by Defense Acquisition Program Administration and Agency for Defense Development under the contract UD140022PD, Korea, and funded by the Ministry of Science, ICT and Future Planning (NRF-2013R1A1A3A04007527).

References

[1] Raychaudhuri, D. & Mandayam, N. B.: Frontiers of Wireless and Mobile Communications, Proceedings of the IEEE, Vol. 100, No. 4, pp. 824-840, 2012

[2] Agrawal, D. & Zeng, Q. A.: Introduction to Wireless and Mobile Systems, Cengage Learning, 2015

[3] Avestimehr, A. S., Diggavi, S. N., & Tse, D. N.: Wireless Network Information Flow: A Deterministic Approach, Information Theory, IEEE Transactions on, Vol. 57, No. 4, pp. 1872-1905, 2011

[4] Maisuria, J. V. & Patel, R. M.: Overview of Techniques for Improving QoS of TCP over Wireless Links, In Communication Systems and Network Technologies (CSNT), 2012 International Conference on. IEEE, pp. 366-370, 2012

[5] Dalal, P., Kothari, N., & Dasgupta, K. S.: Improving TCP Performance over Wireless Network with Frequent Disconnections, International Journal of Computer Networks & Communications, Vol. 3, No. 6, pp. 169-184, 2011

[6] Liu, C. P.: Adaptive Splitting TCP for Wireless Sensor Networks, International Journal of Engineering and Industries (IJEI), Vol. 2, No. 2, pp. 88-94, 2011

[7] Le, D., Fu, X., & Hogrefe, D.: A Cross-Layer Approach for Improving TCP Performance in Mobile Environments, Wireless Personal Communications, Vol. 52, No. 3, pp. 669-692, 2010

[8] Nguyen, T. H., Park, M., Youn, Y., & Jung, S.: An Improvement of TCP Performance over Wireless Networks, In Ubiquitous and Future Networks (ICUFN), 2013 Fifth International Conference on. IEEE, pp. 214-219, 2013

[9] Tiyyagura, S., Nutangi, R., & Reddy, P. C.: An Improved Snoop for TCP Reno and TCP Sack in Wired-Cum-Wireless Networks, Indian Journal of Computer Science and Engineering (IJCSE), 2, pp. 455-460, 2011

[10] Rajasekaran, S. & Pai, G. V.: Neural Networks, Fuzzy Logic and Genetic Algorithms, PHI Learning Private Limited, 2011

[11] Sethis, K., Kole, A., & Bhattacharya, P. P.: Adaptive Fuzzy Logic-based QoS Management in Wireless Sensor Network, Advanced in Electronics and Electrical Engineering (AEEE), 2013 International Conference on, pp. 72-76, 2013

[12] Munir, S. A., Bin, Y. W., Biao, R., & Jian, M.: Fuzzy Logic-based Congestion Estimation for QoS in Wireless Sensor Network, In Wireless Communications and Networking Conference (WCNC), 2007 International Conference on IEEE, pp. 4336-4341, 2007

[13] Teppala, K. & Kumar, K.: A Fuzzy Logic Control in Distributed Traffic Management for Efficient Networks, International Journal of Scientific Engineering and Technology Research, Vol. 3, No. 48, pp. 9788-9793, 2014

[14] Khoukhi, L. & Cherkaoui, S.: Experimenting with Fuzzy Logic for QoS Management in Mobile ad hoc Networks. International Journal of Computer Science and Network Security, Vol. 8, No. 8, pp. 372-386, 2008

[15] Collotta, M. & Scatà, G.: Fuzzy Load Balancing for IEEE 802.11 Wireless Networks, IERI Procedia, Vol. 7, pp. 55-61, 2014

[16] Lee, C. C.: Fuzzy Logic in Control Systems: Fuzzy Logic Controller. II. Systems, Man and Cybernetics, IEEE Transactions on, Vol. 20, No. 2, pp. 419-435, 1990

[17] Zeigler, B. P.: Hierarchical, Modular Discrete-Event Modeling in an Object-oriented Environment, Simulation, Vol. 49, No. 5, pp. 219-230, 1987

[18] Zeigler, B. P., Moon, Y., Kim, D., & Ball, G.: The DEVS Environment for High-Performance Modeling and Simulation, Computing in Science and Engineering, Vol. 4, No. 3, pp. 61-71, 1997

[19] Lim, S., Yu, C., & Das, C. R.: Clustered Mobility Model for Scale-Free Wireless Networks, In Local Computer Networks, Proceedings 2006 31st IEEE Conference on. IEEE, pp. 231-238, 2006

[20] Lenders, V., Wagner, J., Heimlicher, S., May, M., & Plattner, B.: An Empirical Study of the Impact of Mobility on Link Failures in an 802.11 ad hoc network, Wireless Communications, IEEE, Vol. 15, No. 6, pp. 16-21, 2008

[21] Wikipedia Page Traffic Statics, https://aws.amazon.com/items/2596, 2009

[22] Padhye, J., Firoiu, V., Towsley, D. F., & Kurose, J. F.: Modeling TCP Reno Performance: a Simple Model and its Empirical Validation, IEEE/ACM Transactions on Networking (ToN), Vol. 8, No. 2, pp.133-145, 2000

[23] Khan, M. N. I., Ahmed, R., & Aziz, M.: A Survey of TCP Reno, New Reno and Sack over Mobile ad-hoc Network, International Journal of Distributed and Parallel Systems (IJDPS), Vol. 3, No. 1, pp.49-63, 2012

Conceptualization with Incremental Bron-Kerbosch Algorithm in Big Data Architecture

László Kovács[1], Gábor Szabó[2]

[1]University of Miskolc, Institute of Information Technology, H-3515 Miskolc-Egyetemváros, Hungary, e-mail: kovacs@iit.uni-miskolc.hu

[2]University of Miskolc, Hatvany József Doctoral School of Information Sciences, H-3515 Miskolc-Egyetemváros, Hungary, e-mail: szabo84@iit.uni-miskolc.hu

Abstract: The paper introduces a novel conceptualization algorithm optimized for a distributed, Big Data environment. The proposed method uses a concept generation module based on clique detection in the context graph. The presented work proposes a novel incremental version of the Bron-Kerbosch maximal clique detection method. The efficiency of the method is evaluated with random context tests. The presented incremental model is even comparable with the usual batch methods. The analysis of the clique detection algorithm in MapReduce architecture provides efficiency comparison for large scale contexts.

Keywords: ontologization; clique detection; incremental clique generation; mapreduce architecture

1 Introduction

One of the big challenges of current information technology is the efficient information management and knowledge engineering in Big Data environment. With the spread of new technologies like the Internet of Things, the amount of gathered data steadily increases and new data repository techniques are needed to provide an efficient data management. Another trend to be witnessed is the increased demand on intelligent smart applications. The adaption of knowledge engineering methods on Big Data collection is a real challenge for the IT community. The term 'ontology' in information science is defined as "a formal, explicit specification of a shared conceptualization" [10]. The term conceptualization refers to determination of the concept classes and concept relationships at an abstract model level for the phenomenon in the domain world. The ontology models cover not only concepts, but they also include the corresponding constraints on their usage, too. The ontologies emphasize aspects, such as, inter-agent communication and interoperability [25]. From the viewpoint of engineering, the term 'concept' is used as an identifier or a descriptor for a cluster of objects. In this sense, the concept describes besides the naming also the properties of the cluster. In the history of knowledge engineering, a great variety of data structures was developed to represent the meaning of concepts. Nowadays, these models exist parallel and are used for different purposes.

According to [19], ontology models can be classified into very different model categories, based on the purpose, specificity and expressiveness. Application ontology is used to control some computational applications and has a methodological emphasis on fidelity. Reference ontology has a theoretical focus on representation and it is used primarily to reduce terminology ambiguity among members of a community. Based on the abstraction level, generic (upper level) and core and domain ontologies can be distinguished. According to [22], the main representation levels of ontologies are *f*

- Taxonomy: Objects are hierarchically classified, e.g. A is child of B.
- Thesaurus: Objects are related (e.g. A is a B; A is related with B). *f*
- Logic-mathematical representation: Object relations are presented in formal notations (e.g. synonym(a, b):=synonym(b, a);).

Analogously to a database, wherein structure and data form the whole, an ontology consists of rules and concepts. Languages for the description of ontologies are RDF-S, DAML+OIL, F-Logic, OWL, WSML or XTM. Using rule-based representations in deductive databases ensures further facts can be deduced from stored relations.

Current ontology languages, like OWL, provide efficient tools to perform complex operations on the ontology database, like consistency verification, rule induction or reasoning process. The main application area of ontology frameworks is the area of knowledge engineering [11], where the ontology engine is integrated into internal module of expert systems. The efficiency of the ontology engines is based primarily on the correctness, completeness and integrity of the ontology database. From this point of view, the key element in ontology management is application of efficient and proper ontology database construction methods.

There are many difficulties in ontology construction, for instance, huge amounts of information to be collected, huge diversity of information sources and inconsistency within the different information sources. The ontology database construction usually contains the following steps [23, 42]:

1. Ontology scope
2. Ontology capture
3. Ontology encoding
4. Ontology integration

5. Ontology evaluation
6. Ontology documentation.

The current ontology construction methods are mainly based on automated ontology construction methods. In the automated ontology construction methods, the information is usually extracted from documents in natural language. This step requires a complex knowledge extraction engine including among others a natural language processing (NLP) module and a concept identification module.

The main goal of the paper is to introduce a novel conceptualization algorithm optimized for a distributed, Big Data environment. The next section provides a survey on the development of the text to ontology methods. The main goal of text to ontology module is to assign the best matching concept cluster to the new words found in the source text. The third section introduces some approaches for concept assignment. The most sophisticated and most complex one is based on clique detection mechanism in the context graph. The paper presents a novel incremental version of Bron-Kerbosch maximal clique detection method. The efficiency of the method is evaluated with random context test. The fourth section contains the analysis of the distributed Big Data architecture. This architecture provides an efficient implementation framework to process a large amount of heterogeneous text document sources. The proposed architecture and map-reduce processing models are implemented in a test environment where the efficiency of the prototype system could be evaluated.

2 Process of Word to Concept Mapping

The Word Sense Disambiguation (WSD) [18] is a method to select the appropriate meaning for a given context. The Word Category Map method [13] clusters the words based on their semantic similarity. The words having similar contexts belong to the same cluster or they appear close to each other on the map. The context of a word can be constructed in many different ways. In the simplest case, the context is equal to the set of neighboring words within the sentences [21]. In this model, the context is converted into a vector representation calculating the average neighborhood vector. The main benefit of this method is the cost efficiency and the simplicity. On the other hand, this model cannot manage the ambiguity of the words, i.e. a word can carry many different meanings. In this model, the semantic similarity is measured using the standard vector distance methods. Beside the simple vector similarity methods, there are approaches to use more sophisticated clustering methods, like the SOM method [21]. In the other group of approaches, the context is given with a graph instead of a simple vector. In ontology management, the thesaurus graphs are used to denote the higher level semantics.

The WSD ontologization process can work in one the following modes [4]: dictionary based, supervised, semi-supervised and unsupervised. In the first mode, a dictionary containing the definitions of the different concepts is used as background knowledge. The similarity of two words is measured with the overlap of their dictionary definitions [16]. In the case of supervised mode, a background ontology is referenced to get information on existing semantic clusters. In the ontology database, a word can be assigned to several meanings. In order to distinguish the different concepts related to a word, a specific component is introduced which represents the meaning. The most widely used implementation of this component is the synset component defined in a wordnet ontology database.

Wordnets are usually based on architecture presented first in [8] and they organize semantic-lexical knowledge into a graph knowledge base. Nowadays, Wordnet knowledge bases are available for many different languages.

The synset can be considered as the set of words carrying a common meaning. The elements of a synset are synonyms and a word can be an element of different synsets. Using the wordnet knowledge base, the ontologization process performs a pattern matching task, where the matching method locates the synset most similar to the given word. Within this method, the key parameters refer to the similarity measure applied to compare a word and a synset. In most approaches, the similarity measure between the word and the synset is aggregated from the similarity values between the target word and the words in the synset.

In the literature there are different approaches to measure similarity between a word and a synset. The extension of the related proportion approach [20] yields the neighborhood similarity measured with:

$$d(w,C) = \sum_r \alpha_r \sum_{v \in N_{wr}} \frac{n_{Crw}}{1+\log(|C|)}.$$

Where

w: the target word

C: the synset

v: word in the knowledge base

r: relation in the semantic graph

n_{Crv} : the number of edges from elements of C to v along with an edge type r

N_{wr} : the set of words connected to w along with an edge of type r

α_r: the weight factor of the relationship r.

The average cosine method (see for example [3]) calculates distance as the average cosine value between the description vectors:

$$d(w,C) = \frac{\sum_{c \in C} \cos(\overline{N}_c, \overline{N}_w)}{|C|}.$$

Where

$\overline{N}_v$: the adjacency vector for word v (describing the words connected to v).

Having an ontology graph, an important step in text to semantic conversion is to select the proper meaning of the word.

The supervised approach requires the development of a background knowledge base in the form of a dictionary or of an annotated text corpus. The quality of the WSD process depends on the quality of the background knowledge base. The main problem of this approach is the high cost in the generation of a comprehensive and valid knowledge base. The unsupervised methods provide automatic approaches for construction of the knowledge base.

The first important result on the field of unsupervised WSD was the semi-supervised proposal in [27]. The proposed method is based on two main properties of human languages:

- Nearby words provide strong and consistent clues as to the sense of a target word
- The sense of a target word is highly consistent within any given document.

The algorithm first generates a small set of seed representative of the different senses of a word. This step is a supervised phase to assign the semantic label to some of the word occurrences. In the next step, a supervised classification algorithm is used to learn the differences between the contexts of the different word senses. In the last step, all word occurrences are classified with the generated classifier to one of the sense labels.

The unsupervised method of [4] requires only the WordNet sense inventory and unannotated text to determine the meaning category of a target word. The algorithm includes the following processing steps. First, a pool of application context is collected from the web. In the next step, the sentences containing the target word are parsed with the help of a dependency parser in a parser tree. The third phase is used to merge these trees into a dependency graph. In the last step a graph matching algorithm is applied to find the appropriate meaning. This phase is based on the idea that if a word is semantically coherent with its context, then at least one sense of this word is semantically coherent with its context.

In the case of fully unsupervised methods, no background knowledge base is available, only an unannotated source text can be used. In this case, only the unsupervised clustering method can be used to create synsets for the words in the document pools. Clustering methods are used to partition objects into groups where the objects within the same group are similar to each other while objects from different groups are dissimilar. In general, the mentioned clustering methods do not usually meet this basic constraint of clustering, as they allow building of large clusters containing object pairs with low similarity. One way to provide better clustering is the application of Quality Threshold Clustering [12]. The QTC method ensures that the distance between any two elements within a cluster should be below a given threshold.

All of the mentioned clustering algorithms generate non-overlapping clusters. There are some application areas where the constraint that every element must belong to only one cluster that is not met. In the semantic clustering, for example, a word may belong to different clusters, i.e. to different meanings at the same time. Clustering in social networks or in distributed networking are other application areas where non-overlapping clustering yields in significant loss of information. It is shown in [15] that overlapping improves the approximation algorithms significantly for minimizing graph conductance.

Overlapping clustering can be considered as a generalization of the standard clustering methods. The first important approach for overlapping clustering is given in [14]. In this approach, the input structure is a graph where the edges denote object pairs having a similarity value above a given threshold. A cluster is considered as a maximal complete subgraph, i.e. all of the members are connected to each other. The proposed cluster detection method is based on the k-ultarmetrics.

Another group of approaches is based on the extension of the classical clustering methods. In [5], the k-means method is adjusted for overlapping clustering. Initially, random cluster centers are specified. In the next phase, the elements are assigned to a subset of clusters. For a given element *x*, the set of container clusters is generated in the following way. First, sort the cluster centers based on the increasing distances from the given element. Calculate the set *A* of the first *k container c*lusters for which:

$$\sum_x \| x - \varphi(x) \|^2 \to \min$$

is met, where

$$\varphi(x) = \frac{\sum_{c \in A} m_c}{|A|},$$

m_c : the cluster prototype for cluster c.

In the third step, the new positions of the cluster centers are calculated. The performed tests show that this method is a good alternative of the more sophisticated complex techniques.

The third important category of overlapped clustering methods is based on the mixture model. In general, the key input source for a mixture model is the observation matrix *X*. The unknown parameter set is denoted with Θ. The basic assumption is that each data X_i point belongs to the following probability density [1]:

$$p(X_i \mid \Theta) = \sum_{h=1}^{k} \beta_h p_h(X_i \mid \Theta_h)$$

where

k: the number of mixture components

Θ_h: the parameters of the h-th mixture component

β_h: the probability of the h-th mixture component.

The parameter values are usually calculated with the EM method where the goal is to maximize likelihood of the given observation set:

$$\prod_{i=1}^{n} p(X_i \mid \Theta).$$

3 Incremental Bron-Kerbosch Algorithm

Our investigation focuses on the clustering with the clique detection approach originated from the work of [14]. The observation graph can be considered as a similarity graph. The nodes are objects of the problem domain and there is an undirected edge between two vertices (v_i,v_j) if and only if the $d(v_i,v_j) < \varepsilon$ for a given threshold ε. A clique, i.e. maximal complete subgraph corresponds to a cluster. A cluster symbolizes a concept in the target ontology knowledge base. The goal of the investigated algorithm is to detect all maximal cliques in the observation graph.

The Bron-Kerbosch algorithm is one of the most widely known and most efficient algorithms for maximal clique detection. The algorithm was presented first in [2]. The Bron–Kerbosch algorithm uses a recursive backtracking method to search for all maximal cliques in a given graph *G(V,E)*. The pseudocode of the algorithm:

```
BronKerbosch(R, P, X):
if $|P| = |X| = 0$
    report R as a maximal clique
$\forall v \in P$:
    BronKerbosch($R \cup \{v\}, P \cap N(v), X \cap N(v)$)
    $P = P \setminus \{v\}$
    $X = X \cup \{v\}$
```

In the pseudocode, the following notations are used:

P: the subset of V that can have some common elements with the investigated clique

R: the subset of V that share all of its elements with the investigated clique

X: the subset of V that is disjoint with the investigated clique

N(v): the set of vertices connected to v.

Initially, the set P contains all of vertices of G and both R and X are empty sets. In every iteration call, an element of P is processed and the sets P, R, X are refined based on the current neighborhood set. For the recursive call, the set of possible clique members are restricted to the elements in the neighborhood set. Similarly, the set of possible excluded elements are reduced to a subset of the neighborhood set. In the main loop, the elements of P are tested in a predefined order. After testing an element v from P, it will be removed from the set of candidate vertices.

The basic version of the Bron-Kerbosch algorithm uses a large number of recursive calls, resulting in an execution complexity of worst-case running time O($3^{n/3}$) [7] [24]. The updated method uses a specific pivoting strategy to cut computational branches. The pivot element is selected as the element with highest number of neighbors.

```
TomitaBronKerbosch(R, P, X):
if $|P| = |X| = 0$
    report R as a maximal clique
let $u \in P \cup X$: $|N(u) \cap P| \rightarrow max$
$\forall v \in P \setminus N(u)$
    TomitaBronKerbosch($R \cup \{v\}$, $P \cap N(v)$, $X \cap N(v)$)
    $P = P \setminus \{v\}$
    $X = X \cup \{v\}$
```

The presented methods generate the output structure for a fixed given input context. This approach is used for static investigation when there is no change in the input context. The incremental construction method is used for such problems where the initial context is extended incrementally. The incremental methods are used for applications where context changes from time to time. The modeling of cognitive learning processes is a good example of such dynamic problem domains.

Our investigation focuses on the incremental clique generation method. For the analysis of the proposed method, the following initial notations are used:

G(V,E) : context graph

V : set of vertices in *G*

E : set of edges in *G*

N : number of vertices in *V*.

It is assumed that there exists a total ordering of the vertices, .i.e.

$$V = \{v_i \mid i \in [1..N]\}\ .$$

Taking only the first *n* elements of *V*, we get a reduced context graph $G_n(V_n,E_n)$, where

$$V_n = \{v_i \mid v_i \in V, i \le n\}$$

$$E_n = \{(v_i, v_j) \mid (v_i, v_j) \in E, v_i, v_j \in V_n\}\ .$$

Let $C(G_n)$ denote the set of maximal cliques related to G_n. The cliques in $C(G_n)$ can be separated into two disjoint groups, depending on the property whether they are new cliques or old cliques related to $C(G_{n-1})$.

$$C(G_n) = C^u(G_n) \cup C^o(G_n)$$

$$C^o(G_n) = \{c \mid c \in C(G_n) \cap C(G_{n-1})\}$$

$$C^u(G_n) = C(G_n) \setminus \mathrm{C}^{\mathrm{o}}(G_n)$$

It can be easily verified that for every $c \in C^u(G_n)$,

$$v_n \in c\ .$$

Regarding $C^u(G_n)$, the following proposition can be used in the incremental clique generation method.

Proposition 1:

The set of new cliques of G_n is equal to the clique set generated for the inclusive neighborhood of the new vertex.

$$C^u(G_n) = C(S_n)$$

where

$$S_n(x) = (V'_n, E'_n)$$

$$V'_n = \{v_i \mid v_i \in V, i < n, (v_i, v_n) \in E\} \cup \{v_n\}$$

$$E'_n = \{(x,y) \mid (x,y) \in E, x \in V'_n, y \in V'_n\}.$$

The inclusive neighborhood set S_n is defined as the neighborhood of v_n extended with the element v_n.

Proof.

Let us take a new clique in G_n:

$$c \in C^u(G_n)\ .$$

It can be seen that

$$v_n \in c\,,$$

i.e. v_n is connected to all other elements of c. Thus, all elements of c are in the inclusive neighborhood of v_n and all the edges in G_n are also edges in S_n. This means that c is a complete subgraph of S_n, too. As the assumption that c is maximal in G_n and not maximal in S_n, imply a contradiction, the clique c is a maximal complete subgraph in S_n. Thus

$$C^u(G_n) \subseteq C(S_n)\,.$$

On the other hand, taking a c clique from $C(S_n)$, c will be complete in G_n, because all the edges in S_n are also edges in G_n. If c is not maximal in G_n then there exists a vertex v_i such that

$$v_i \notin c, v_i \in V_n, (v_i, v_n) \in E_n\,.$$

Thus, $i < n$ and

$$v_n \in V^n\,.$$

This means that c is not maximal in S^n, and this is a contradiction, i.e.

$$C^u(G_n) \supseteq C(S_n)\,.$$

Based on the considerations shown in the proof, we get:

$$C^u(G_n) = C(S_n)\,. \qquad \square$$

Proposition 1 can be used to generate the new cliques in an effective way, but not only the insertion of the new cliques is the required update operation on $C(G_{n-1})$. Namely, some of the cliques in $C(G_{n-1})$ may become invalid as they are not maximal anymore. For example, in Figure 1, the clique set of G_4 is {(1,3),(2,3),(3,4)}. After adding v_5, S_5 is equal to ({2,3,4,5}, {(2,3), (2,5), (3,4), (3,5), (4,5)}). The clique set for S_5 is {(2,3,5),(3,4,5)}. The resulting clique set for G_5 is {(1,3), (2,3,5), (3,4,5)}. Thus, the clique (2,3) is covered by (2,3,5) and (3,4) is covered by (3,4,5).

Thus, the update of $C(G_{n-1})$ after generation of $C(S_n)$ includes the removal of some existing cliques and the insertion of some new cliques. As the covered clique is the same as the new clique except the v_n vertex, it is worth to reduce S^n to the elements of the neighborhood and to exclude v_n. This new graph is denoted with S_{n-}.The incremental clique generation algorithm can be summarized in the following listing:

```
IncrementalBronKerbosch($G_{n-1}$, $C(G_{n-1})$, $S_n$):
$C(S_{n-})$ = BronKerbosch($S_{n-}$)
$C(G_n)$ = $C(G_{n-1})$
```

```
$\forall c \in C(S_{n-})$:
    $C(G_n) = C(G_n) \setminus \{c\}$
    $c' = c \cup \{v_n\}$
    $C(G_n) = C(G_n) \cup \{c'\}$
return $C(G_n)$
```

The lookup operation in the set $C(G_n)$ has a central role in optimization of the algorithm. Instead of a naive sequential lookup operation having a cost $O(D)$ where D denotes the number of elements in the set, a prefix tree structure is used. In the prefix tree structure, the clique sets are represented with an ordered list of vertex identifiers. The tree stores these ordered lists where the lists having the same prefix part share the same prefix segments in the tree.

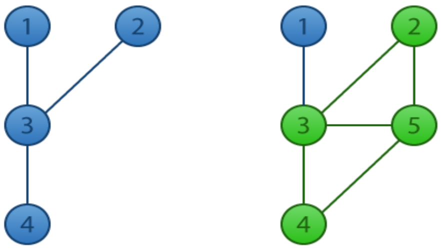


Figure 1
Extension of the similarity graph

The results of the performed direct tests show that the proposed method provides an efficient solution for incremental clique detection tasks. In Figure 2, the comparison of the naive incremental method and the proposed incremental method is presented. The time shows the execution time in logarithmic scale. As the result shows, the proposed algorithm is about 100 times faster than the naive method. In the naive method, for every new incoming object, the whole clique set is recalculated from scratch. In Figure 2, the data set NI denotes the naive method and symbol I is for the proposed incremental method.

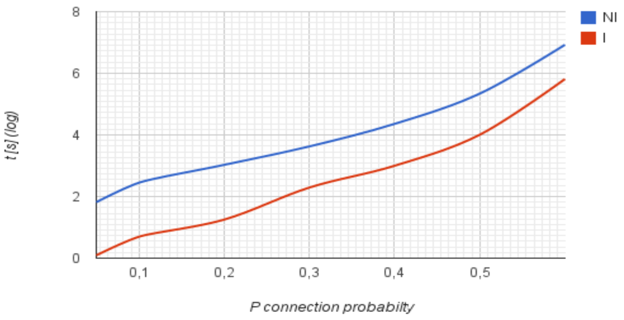


Figure 2
The comparison of the naive and the proposed methods

The method outperforms not only the other incremental approaches but it has better cost characteristics than the standard batch method. For larger input graphs, the proposed method is significantly faster than the basic Bron-Kerbosch method (see Table 1, Table 2). The runtime is displayed in seconds.

Table 1
Comparison of the basic Bron-Kerbosch and the proposed methods for edge probability 0.5

Vertices	Cliques	BK Runtime	Incremental Runtime
100	5407	0.41	0.32
150	23440	2.47	1.91
200	76838	10.42	8.36
250	214457	38.1	28.6
300	496754	103.5	82.4

Table 2
Comparison of the basic Bron-Kerbosch and the proposed methods for edge probability 0.3

Vertices	Cliques	BK Runtime	Incremental Runtime
100	749	0.04	0.04
150	2277	0.12	0.08
200	5126	0.18	0.12
250	9566	0.41	0.24
300	16858	0.80	0.51

The efficiency of the proposed incremental algorithm is based on the divide and conquer paradigm. The algorithm generates the clique set only for the neighborhood graph during each iteration. The size of the neighborhood graph depends on the edge probability in the input graph. If the cost function of the basic Bron-Kerbosch algorithm is denoted with

$$O(C_b(n))$$

then the complexity of the proposed incremental method belongs to

$$O(n(C_b(n') + D_b(n')))$$

where

n' : the average size of the neighborhood graph,

$D_b(n)$: the number of cliques in a graph containing n vertices.

The first term in the formula corresponds to the generation of the cliques for the neighborhood graph while the second term relates to the elimination of the covered cliques from the result set. The presented formula provides an acceptable approximation of the measured cost values but further investigation is needed to work out a more accurate cost function involving the additional parameters, too.

4 Implementation in the MapReduce Architecture

Although the speed of computers is increasing rapidly, the fact that using several machines in an interconnected system can be more effective in certain cases was recognized many years ago. There are many solutions in the literature that can be used in Big Data analysis. MPP (Massively Parallel Processing) systems [29] store data after splitting it according to its features, for example we could store regional data split by the country or state. In-memory database systems [30] are very similar, except that they store data in memory, thus speeding up the retrieval of the records. This area is the topic of active research by Big Database vendors such as Oracle. BSP (Bulk Synchronous Parallel) systems [31] use multiple transformation processes that run in parallel on different nodes. Each process gets data from a master node and then sends the result back. After this barrier-like synchronization the next iteration can be executed.

The big breakthrough came when Google published their scientific article involving a new architecture for processing huge amounts of data. This architecture is called MapReduce. The article called *MapReduce: Simplified Data Processing on Large Clusters* [28] created a whole new concept that became the basis for the most popular framework of Big Data analysis to date. The concept itself is very simple. We all know that parallel tasks are most effective when we can run in parallel for a long time without any synchronization barriers. This means that if we can divide our algorithms in such a manner, the result can outperform the single-threaded version. Conversely, if we need to communicate between the threads, these synchronization points can slow down the algorithm.

MapReduce got its name from its two most important phases: map and reduce. The map phase produces key-value pairs which become the input of the reducer phase that yields the final results. Although Google's implementation is not open source, there are multiple open source implementations among which the most popular one is Apache Hadoop. This framework is actively developed, constantly improving and has many technologies built on top of it like Apache Pig (SQL-like interface), Apache Hive (data warehouse infrastructure), Apache HBase (distributed NoSQL database), etc. that specializes the Hadoop platform to more specific problems.

The base concept of MapReduce and Apache Hadoop [32] is to split the incoming data to multiple nodes and process them locally. This way – after the initial data copies – the mapper tasks can work on their local data instead of retrieving them during processing. To make sure that the data is not lost on node failures, every data chunk is stored on multiple nodes, but in case of normal work without failures every node works on its local hard disk. These low-level interactions are abstracted by the Hadoop Distributed File System (HDFS), inspired by the Google File System (GFS) [9]. This is the most important component of Hadoop as it provides a distributed file system for the MapReduce tasks. The input data must be copied to this file system – which makes sure that every data chunk is persisted on multiple nodes –, then the mappers get data from this file system and the reducers write the results back to HDFS. Figure 3 displays an overall view of a very simplified MapReduce task. On the left side of the image we can see the input records that come from HDFS. Next to them there are some mapper tasks that receive one input record at a time and produce key-value pairs from them. For simplicity the figure only has two types of keys (green and blue). The reducers receive objects with common keys and create the final output of the application that is persisted back to HDFS.

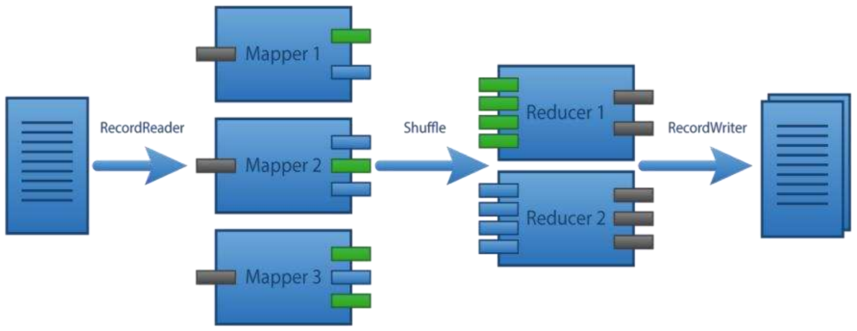


Figure 3
The MapReduce architecture

There are many scientific research areas that are directly or indirectly related to MapReduce and Apache Hadoop. During literature research we can find scientific articles from the area of agriculture [33] through telecommunication [34] to AI-based recommendation systems [35] that use Hadoop as the application platform. It is not surprising that graph algorithms can be ported to this parallel architecture and thus speeds up different search methods.

The main research areas of Hadoop based graph processing are the semantic web related problems and processing of social network data. This work [36] tries to solve the problem of maintaining and querying huge RDF graphs by using Hadoop, and provides an interface on top of it to answer SPARQL queries. (SPARQL [37] is the W3C standard language for querying ontologies based on

RDF triplets.) This is a classic ontology related problem that can be integrated with MapReduce to make the processing distributed.

There are also proposals [38] on a Hadoop based solution to the problem of processing huge graphs in parallel. Unlike most solutions that try to separate the nodes in the form of classic MapReduce jobs, this article presents a system that has a highly flexible self-defined message passing interface that makes it easier to port graph algorithms on top of the MapReduce platform. GPS (Graph Processing System) [39] is a complete open-source system similar to Google's Pregel [40], extending it to provide an interface for processing huge graphs in parallel with global communication among the nodes. As we can see, these two research results combine Big Data analysis with graph processing, thus connecting to our research area and results presented in this article.

The Spider system [41], tries to solve the problem of processing data on the semantic web. The system has two modules: one that loads the graph and one that can query the previously loaded graph leveraging the Hadoop framework.

Considering the implementation of the clique detection algorithm, two different approaches were tested. In the first approach, the task unit is the processing of the neighborhood graph of a node. Here, for every node a separate task is generated. The mapper calculates the clique set in the neighborhood. The drawback of this approach is that it generates the same clique several times. The main benefit of the method is that every node can work separately with a smaller amount of local data.

In the second approach, all of the nodes work with the global data set and, share the global graph. The work is separated in such way that every clique is generated only once. Here, the merging process can be executed with low cost. Alternately, every node requires the whole dataset.

Regarding the first approach (A), the full graph is divided by its nodes. After the GraphRecordReader reads the input file from HDFS and reconstructs the graph object, it takes each node from the graph and all of its neighbors and yields a new subgraph from these nodes. This way if we find a maximal clique in the subgraph, it will be a maximal clique of the full graph as well. The Hadoop framework distributes these subgraphs based on the system configuration and it makes sure that every mapper node gets approximately the same number of subgraphs. Each mapper process gets one subgraph at a time with a null key, and executes one of the Bron-Kerbosch variations on it. After getting the maximal cliques, it calculates the hash code of the resulting cliques and yields key-value pairs consisting of the hash code as the key and the clique as the value. Since the mapper node might produce the same clique multiple times, a combiner is used on each node that drops every repeated result to optimize speed so that these don’t have to be sent over the network to the reducer nodes.

The reducer then gets one hash code at a time and the list of cliques that have that hash code. Ideally this list only contains one clique, but if multiple mappers produced the same clique, the size of the list can be more than one. Therefore the reducer yields only the first element of the list. The Hadoop framework then writes the resulting cliques from the reducer's output to HDFS.

In the second approach (B), the set of cliques is divided by the smallest node index value within the clique. This method yields in a disjoint partitioning, thus the merge phase can be implemented with a minimal cost. In the implementation of the method a shared HDFS storage is used to store the input similarity graph. The map component performs a Bron-Kerbosch clique detection algorithm where the main loop is restricted to the nodes assigned to this mapper process. The key value in the MapReduce framework is equal to the set of minimum index values assigned to a node for processing. Every mapper node works with the common shared input graph to generate the corresponding clique set.

We tested the two approaches on randomly generated graphs with fixed node counts and edge probabilities to verify their correctness in practice. Although we didn't have a full Hadoop cluster up and running, we simulated such environments with the help of Docker, a lightweight virtualization software. We set up two docker images and initiated one master and related slave containers, both executing map and reduce tasks. In the future we would like to set up a physical Hadoop cluster and verify our assumptions in a real distributed environment.

The cost models of the proposed methods can be approximated with the following formulas:

$$\cos t_A = \frac{N}{L} C(N_n) + N \cdot C(N) + N_n \cdot N$$

$$\cos t_B = \frac{C(N)}{L} + L \cdot N$$

where

N: the number of nodes in the input graph

N_n : the number of nodes in a neighborhood graph

L: the number of mapper nodes

C(n) : the number of cliques for a graph having *n* vertices.

The formula is based on the fact that the cost of clique detection algorithm is a linear function of the number of generated cliques. The experimental function of *C(N)* is shown in Figure 4, while Figure 5 presents the corresponding execution costs. These formulas and the test results show that method A is suitable for those architectures where the neighborhood graph is relatively small (sparse input graph) and the L value is large. The method B is optimal for the cases where L is small and the graph is relatively dense.

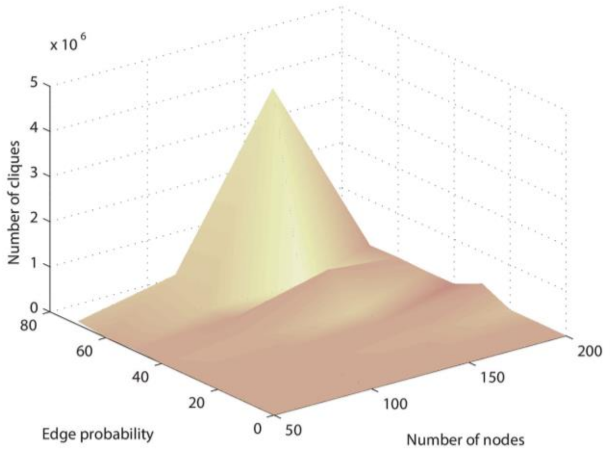


Figure 4
The number of generated cliques

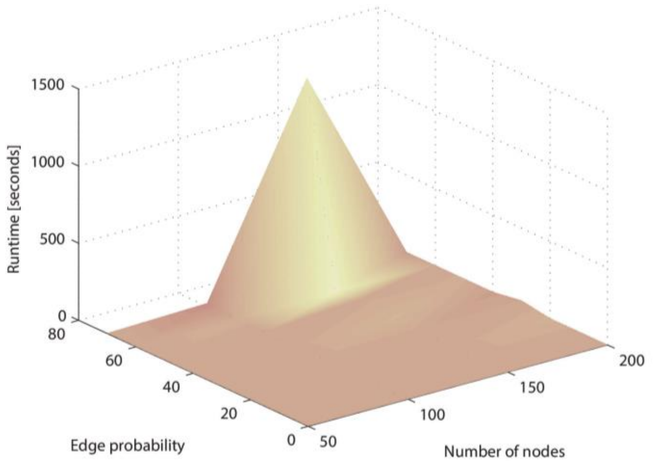


Figure 5
The cost function of the proposed iterative method

Conclusions

One option to generate concepts in an ontology framework, is to build clusters of similar objects. The cliques in a similarity graph can be considered as overlapping quality threshold clusters. The proposed incremental clique detection algorithm provides an efficient clustering method for dynamic contexts changing from time to time. The presented incremental model is comparable even with the usual batch methods. The cost analysis for large contexts shows that the optimal implementation in the MapReduce architecture depends on the basic parameters of the input similarity graph.

References

[1] Banerjee A., Krumpelman C., Basu S., Mooney R.: Model-based Overlapping Clustering, Proc. of Eleventh ACM SIGKDD, pp. 532-537

[2] Bron C., Kerbosch J., "Algorithm 457: Finding All Cliques of an Undirected Graph", Communications of ACM 16 (9): 1973, pp. 575-577

[3] Caraballo S.: Automatic Construction of Hypernym-Label Noun Hierarchy from Text, Proc. of Annual Meeting ACL, 1999, pp. 120-126

[4] Chen P, Ding W, Bowes C, Brown D: A Fully Unsupervised Word Sense Disambiguation Method Using Dependency Knowledge, The 2009 Annual Conference of the North American Chapter of the ACL, pp. 28-36

[5] Cleuziou G.: An Extended Version of the k-means Method for Overlapping Clustering, Pattern Recognition, ICPR 2008, pp. 1-4

[6] Dean J., Ghemawat S.: MapReduce: Simplified Data Processing on Large Clusters, Proc. of OSDI 2004, pp. 10-20

[7] Eppstein D., Löffler M., Strash D.: Listing all Maximal Cliques in Sparse Graphs in Near-Optimal Time, Proc. of ISAAC 2010, pp. 403-414

[8] Fellbaum C.: WordNet: An Electronic Lexical Database (Language, Speech and Communication), MIT Press, 1998

[9] Ghemawat S., Gobioff H., Leung S.: The Google File System. Proc. of the 19th SOSP 2003, pp. 29-43

[10] Gruber T.: A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition, 5(2): 1993, pp. 199-220

[11] Happel H., Seedorf S.: Applications of Ontologies in Software Engineering. Proc. of SWESE 2006, Athens, GA, USA

[12] Heyer L., Ramakrishanan R., Livny M.: BIRCH: An Efficient Data Clustering Method for Very Large Databases, Genome Research, Vol. 9, 1999, pp. 1106-1115

[13] Honkela T: Comparisons of Self-Organized Word Category Maps, In Proceedings of WSOM'97, pp. 298-303

[14] Jardine N., Sibson R.: Mathematical Taxonomy, John Wiley and Sons Publ.

[15] Khandekar R, Kortsarz G., Mirrokni V.: On the Advantage of Overlapping Clusters for Minimizing Conductance. Algorithmica, 69(4), 2014, pp. 844-863

[16] Lesk M.: Automatic Sense Disambiguation using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In Proc. of SIGDOC ’86

[17] Moon, J. W., Moser, L., "On Cliques in Graphs", Israel J. Math. 3:, 1965, pp. 23-28

[18] Navigli R.: Word Sense Disambiguation: A Survey, ACM Computing Surveys, 41(2), pp. 1-69

[19] Oberle D.: Semantic Management of Middleware, Volume I of The Semantic Web and Beyond Springer, New York (2006)

[20] Pennacchiotti M., Pantel P.: Ontologizing Semantic Relations, Proc. of 21st COLING ACL, 2006, pp. 793-800

[21] Ritter H., Kohonen T.: Learning 'Semantotopic Maps' from Context. In Proc. IJCNN- 90-WASH-DC, Int. Conf. on Neural Networks, Vol. I, pp. 23-26

[22] Sieber T., Kovács L.: Mapping XML-Documents into Object-Model, Proc. of CINTI 2005, pp. 343-354

[23] Subhashini R., Akilandeswari J.: A Survey on Ontology Construction Methodologies, International Journal of Enterprise Computing and Business Systems, Vol. 1, Issue 1, 2011

[24] Tomita E., Tanaka A., Takahashi H., The Worst-Case Time Complexity for Generating all Maximal Cliques and Computational Experiments, Theoretical Computer Science 363 (1): 2006, pp. 28-42

[25]. Uschold M., Gruninger M.: Ontologies: Principles, Methods, and Applications. Knowledge Engineering Review 11 (1996) pp. 93-155

[26] Wood D.: On the Maximum Number of Cliques in a Graph, Graphs Combin. 23 (2007) pp. 337-352

[27] Yarowsky D.: Unsupervised Word Sense Disambiguation Rivaling Supervised Methods, Proc. of ACL'95, 1995, pp. 189-196

[28] Dean J., Ghemawat S.: Mapreduce: Simplified Data Processing on Large Clusters. Commun. ACM 2008, 51(1), pp. 107-113

[29] Simon, H. D.: Partitioning of Unstructured Problems for Parallel Processing. Computing Systems in Engineering, 2(2), 1991, pp. 135-148

[30] Berkowitz, B. T., Simhadri, S., Christofferson, P. A., and Mein, G. In-Memory Database System. US Patent 6, 2002, 457,021

[31] Gerbessiotis, A. V. and Valiant, L. G.: Direct Bulk-Synchronous Parallel Algorithms. Journal of Parallel and Distributed Computing, 22(2):1994, pp. 251-267

[32] Wadkar, S., Siddalingaiah, M., and Venner, J.: Pro Apache Hadoop. Apress, 2014

[33] Yang, F., Wu, H.-R., Zhu, H.-J., Zhang, H.-H., and Sun, X.: Massive Agricultural Data Resource Management Platform Based on Hadoop [J]. Computer Engineering, 2011, 12:083

[34] Yang, G., Shu, M., Wang, X., and Xiong, A.: Application Practice of Hadoop Platform for Telecommunication Enterprise. Digital Communication, 4:019, 2014

[35] Wang, C., Zheng, Z., and Yang, Z.: The Research of Recommendation System Based on Hadoop Cloud Platform. In Computer Science & Education (ICCSE), 2014 9th Int. Conference on, pages 193-196. IEEE

[36] Husain, M. F., Doshi, P., Khan, L., and Thuraisingham, B.: Storage and Retrieval of Large RDF Graph Using Hadoop and MapReduce. 2009, pp. 680-686

[37] Quilitz, B. and Leser, U.: Querying Distributed RDF Data Sources with SPARQL. In The Semantic Web: Research and Applications, Vol. 5021 of Lecture Notes in Computer Science, 2008, pp. 524-538

[38] Pan, W.; Li, Z.-H.; Wu, S.; Chen, Q.:Evaluating Large Graph Processing in MapReduce Based on Message Passing. Chinese Journal of Computers., 2011, pp. 1768-1784

[39] Salihoglu, S. and Widom, J.: GPS: A Graph Processing System. In Proceedings of the 25th International Conference on Scientific and Statistical Database Management, SSDBM, 2013, pp. 22:1-22:12

[40] Malewicz, G., Austern, M. H., Bik, A. J., Dehnert, J. C., Horn, I., Leiser, N., and Czajkowski, G. (2010) Pregel: A System for Large-Scale Graph Processing. In Proc. of ACM SIGMOD 2010, pp. 135-146

[41] Choi, H., Son, J., Cho, Y., Sung, M. K., and Chung, Y. D.: Spider: A System for Scalable, Parallel/Distributed Evaluation of Large-Scale RDF Data. In Proceedings of ACM Conference on Information and Knowledge Management, 2009, pp. 2087-2088

[42] Furdik K., Tomasek M., Hreno J.: A WSMO-based Framework Enabling Semantic Interoperability in e-Government Solutions, Acta Polytechnica Hungarica, Vol. 8, No. 2, 2011, pp. 61-79

Visualising Software Developers' Activity Logs to Facilitate Explorative Analysis

Alena Kovarova, Martin Konopka, Lukas Sekerak, Pavol Navrat

Slovak University of Technology, Ilkovicova 2, 84216 Bratislava, Slovakia
alena.kovarova@stuba.sk, martin_konopka@stuba.sk, xsekerakl1@stuba.sk, pavol.navrat@stuba.sk

Abstract: In this paper, we discuss whether data collected from monitoring software developers' logs can be considered big. We hypothesize that it falls within the category of Big Data. The main topic of our paper however, is how to facilitate analysis of such data. Due to the specificity of the monitored activity, the analysis is at least partially explorative in its nature. We hypothesize that visualisation can be a productive approach in such a case. We present several visualisation schemes (diagram types) and show those applied to explorative analysis of data gathered within one four year project that we have been participating in.

Keywords: Activity log; log stream; Programmer; Software development; Visualisation; Big data

1 Introduction

Let us consider a serious creative human activity, which is supposed to result in developing a very complex technical product. The human activity is inherently individual by definition and at the same time, due to the nature of the task, it can rarely be accomplished by a single person because of task complexity, delivery time requirements, etc. Thus, we envisage a complex process of multi-human activity that requires coordinated cooperation over a long period of time, with a collective very structured outcome.

There are many occasions during the development of a software product, when it is desirable to know what a particular developer was doing and how they performed, what they were doing with a particular part of the product being developed, or any other similar question. But perhaps, the nature of the technical product is such that it evolves over time or users of the product change their expectations. This translates to modifications of product requirements and specifications, and thus, onward to redeveloping the product, which may be a very complicated endeavour. Certain questions arise, such as "Who of the original developers should be assigned the task?" or "In which place should they attempt to make an amendment?"

To find answers to these and similar questions, we need to know more about the developers' activity. But do we know exactly what the required information is? On the side of data, assume that the development process takes place for several months and there are several, or even a few dozen developers involved. Let us consider that we are able to monitor developer's activity at the granularity of elementary steps taken once a few seconds if not more frequently. No matter how small the single record is, recording it every few seconds over a period of time, for example, several months for several developers will definitely surpass any reasonable volume of data storage available for the usual data processing tasks. More importantly, that would not be a reasonable modus of operation. The data are required to be processed in real-time so that we have the information when needed. Recently, it has become fashionable to describe such data as Big Data [5].

The outcome of software development is a software system, which is by itself a very complex product. The development process as a rule involves activities of several programmers, or more generally software developers, over a period of several months, sometimes years. Their work is essentially writing or modifying texts in a source programming language, and also writing a documentation, solving problems that occur during development, or communicating with each other. However, their actions can be logged at a very low level, way below the level of the programming language, not to speak of the level of developers' actions.

Currently, it is technically feasible to monitor not only when and which text (source code) they write, but also how they write it and which information and communication technology tool they employ. Monitoring generates a lot of data. In this paper, we discuss if data collected from monitoring software developer's logs can be considered big. We hypothesize that it falls within the category of Big Data. The main topic of our paper is, however, how to facilitate analysis of such data. Due to specificity of the monitored activity, the analysis is at least partially explorative in its nature. We hypothesize that visualisation can be a productive approach in such a case. We present several visualisation schemes (diagram types) and show those applied to an explorative analysis of data gathered within one four year project that we have been participating in.

The rest of the paper is structured as follows. In Section 2, we discuss whether data collected on software development can be considered Big Data. We identify so-called interaction data to be such data. In Section 3, we discuss what information could be extracted by visualising data and by what approaches this can be accomplished. We present our approach in Section 4. First, we describe what data is potentially the most interesting in the logs, and then we show our visualisation schemes. We evaluate the approach in Section 5. The paper is concluded in Section 6 where we also suggest some possibilities for future work.

2 Big Data in Software Engineering

In order to evaluate actual status and progress of a software project, it is required to monitor it on a suitable level of detail, from the lowest hardware interactions up to the highest level with performed development tasks [10]. Current research in empirical measurements and evaluation of software development focuses on monitoring developers on the level of interactions, which brings us into dealing with Big Data in software engineering. More precisely, interaction data fulfils the *4 Vs rule* of Big Data [24] and also occurs in real-time, thus the sensible approach is to represent this as a data stream that may be computationally analysed to reveal patterns [12], trends, and associations, especially related to human behaviour and interactions [21]. The motivation for gathering, processing and evaluating such streams of interaction data in software engineering is to get a detailed overview of the developers' work [14], to understand how they behave individually or in groups, and to avoid problems in development.

2.1 Interaction Data in Software Development

Traditional methods of evaluating progress on software projects are based on monitoring completion of development tasks by developers in task management systems. Developers are given tasks to complete, or they identify the tasks themselves, and then update the status of a task in the course of the work [14]. Team leaders are able to observe progress on a project, communicate the current status with developers, or reason upon it. However, this approach fails to identify causes of developers' mistakes, faults, or delays in completion of tasks. A more detailed approach is to monitor and evaluate a software project on the level of source code [14]. However, this still does not resolve the aforementioned problems.

Practical research in software engineering requires the monitoring of software developers in a greater detail [1, 10]. Evaluating software projects and developers through tasks and source code is limited to observing the results of the work only, not the processes that led to those results. Software developers interact with tools to fulfil a task assignment and are affected and surrounded by several different contexts. We summarize this information under the name *interaction data* [14]. The main sources of interaction data are tools and systems that developers work with during software development and maintenance, such as IDE – integrated development environments (Microsoft Visual Studio, Eclipse), revision control systems (Git, Microsoft Team Foundation Server, SVN), task management systems (Redmine, Atlassian Jira), software CASE tools (Sparx Enterprise Architect, IBM Rational Software Architect), operating systems and many other tools, e.g., a web browser, instant messaging, or e-mail client, note-taking software, etc.

As pointed out by the authors in [14], interaction data encompasses mainly the following four types of data:

- Interactions – observable actions of a developer within tools, e.g., navigation in source code, code editing, mouse movements, committing changes to a source code repository, etc.
- Artefacts – entities that a developer interacts with, e.g., source code documents, documentation web pages, physical artefacts or even people.
- Tools – software applications and systems that a developer works with.
- Contexts – decisions, reasons and other circumstances of developer's work, e.g., what, when, how, and why affected them during their work.

As can be seen, interaction data cover almost everything that developers may come into interaction with, and also how they interact in that situation. Although developers' *interactions* are monitored at the lowest possible level of detail, they can still be used to track development tasks (as artefacts) in task management systems/tools, as well as source code (artefacts) and changes (interactions) within revision control systems/tools. Recording this amount of data about software developers allows us to attempt to identify their expertise or familiarity with code [13, 14], to annotate source code with important information [19], or even search for unknown or hidden connections between source code documents [11]. Several other approaches based on utilizing interaction data nowadays arise using the proposed systems [14], although always with predefined assumptions.

2.2 Monitoring Software Developer's Activity

Many approaches and systems have been proposed for monitoring interaction data in software development [1, 6, 10, 18]. However, many operate at different levels of detail. We can observe developers' activity at the following levels [21] (starting with the lowest level):

- Hardware interactions, e.g., mouse moves, key presses, touch gestures.
- Widget interactions – a developer interacts with areas (widgets) of a tool.
  - Single widget interactions, e.g., scrolling in code editing window in an IDE, copying code fragments, selecting a text.
  - Multi widget interactions, e.g., searching for references of a source code entity and using them in a code editing window.
- Activities – enclosed parts of developer's work on a task, e.g., studying a code, adding a new code, debugging, documenting completed work.
- Tasks – described with a goal which a developer is about to accomplish, e.g., fix a bug no. 324, added service endpoint ABC.

Based on the employed level of interactions, we encounter problems with the collecting, processing and storing of interaction data [14]. For example, Mylyn [10] aggregates interaction events in an IDE into five pre-selected types of activities, the IDE++ project [6] records even the key presses and mouse events. However, although authors of IDE++ record almost every event in Eclipse IDE, they do not attempt to store them due to a high volume of data. PerConIK [1] records various interaction events in an IDE. At first, authors in PerConIK attempted to persistently store compressed keyboard button presses and mouse events, but later left them out for the same reasons, despite it being the finest grained data. Such data could help us determine user activity duration or some individual characteristics of either the user or the domain. IDE++ alternatively records all available data and redirects it to other connected tools that examine these events.

Monitoring a developer on the hardware and widget levels is possible thanks to available application programming interfaces in operating systems and development tools. However, identification of activities is not possible with tools because of a vague notion of what an activity in fact is. The authors in [19] attempted to automatically identify activities using Hidden Markov Models, because this technique matched attributes of difference between activity and interaction. Activity is composed of interactions, but only those of a certain intention for a developer, which they had to undertake during their work on a task. From the automation viewpoint, we are not able to unambiguously distinguish between types of activities, e.g., adding a new functionality or refactoring. Because a developer’s interactions occur in real-time, we may attempt to incrementally identify developer’s activities in real-time as well [12]. The motivation behind identifying activities is to better describe development tasks or untangle them when they appear to be overlapping [12, 20].

Whether we monitor developers’ interactions, activities or tasks, the motivation remains the same: to understand how developers approach work on a software project individually and/or cooperatively, how they progress, and how they identify issues with respect to finishing their work in the desired quality and on time. Interaction data occurs in real-time and may be used for identifying patterns, trends or associations in developers’ interactions and activities.

2.3 Software Developer Interaction Data Can Become Big

Interaction events occur very fast during a developer’s work, from the finest level of granularity with recording of every keystroke, up to recording changes in source code contents after every widget interaction, e.g., navigation or scrolling in a document. Using the 4 Vs characteristics of Big Data, we may look upon interaction data as Big Data as well [24]:

- Volume – recording interactions in tools, e.g., every change in a source code document after navigation, mouse button presses and moves.

- Velocity – multiple interactions may occur within a second, or changes in mouse coordinates may be recorded with high frequency (1000 positions per second), multiplied by large development teams.
- Variety – monitoring various keystrokes, mouse, or events in an IDE, e.g., from opening a source code document, through to adding a new line of code, to identifying changes in abstract syntax trees of source code.
- Veracity – monitoring of IDE events cannot be predicted – a developer may study code while not interacting with tools or a computer in any way, thus unexpectedly not generating any data.

Our work is part of the research project PerConIK [1] where we monitor software developers at a medium size software company. Besides, we monitor students of Masters study programmes in Software Engineering and Information Systems who develop their semester projects. For the monitored interaction events within the infrastructure of PerConIK, see section 3.1. Also, note that we do not monitor mouse movements and key strokes that IDE++ does (but without storing them) [6], only interactions in tools because of the high volume which has to be stored if doing so. As an example of a possible data flow, consider the following statistics even for 10 developers monitored within the PerConIK project during 38 workdays in February and March 2015:

- Average velocity if any interaction event occurred in a time frame:
  - 3.682 per second and 18.893 per minute,
- Average velocity if over 10 interactions occurred in that time frame:
  - 71.351 per second and 42.849 per minute.

Although data of 10 developers may not seem really big, we use them as a representative sample from our dataset because of different work habits and experiences that students have, and because the setup of the PerConIK infrastructure has evolved over time. During this period, there were exactly 27,994 interactive minutes (i.e. minutes with at least 1 recorded interaction; if counting only interactive seconds, there were exactly 29,628 of such seconds). Pointedly, monitoring more developers for a longer time would increase these numbers in orders of magnitude.

Because of complying with the 4 Vs characteristics, interaction data and developers' activity are often subject to visualisation. Omoronyia et al. provide a good overview of 12 tools [18] visualising developers' activity. One example is the visualisation tool Team Tracks that displays the current activity of a developer, or which files are currently edited and who altered them. Their plugin to an IDE monitors which files were frequently visited and assumes that these files can be problematic. However, none of the mentioned tools [18] dealt with the visualisation of software development from a perspective similar to ours. This paper indicates the importance of capturing data for tracking developers' activity and, of course, the visualisation of this data.

3 Visualisation of Software Developers' Activity

In order to record huge amounts of data, it is usually required to choose a form of representation that facilitates eliciting important information. The information may be in the form of patterns, trends or associations found in raw data. There exists a variety of methods for acquiring such information based on statistics, artificial intelligence, machine learning, or formal concept analysis. In most of the cases, we need two things to choose the right method – 1. to have the data, and 2. to know what we are looking for in the data (either exact or abstract). For example, we can track certain quantities of events, or find certain patterns in them, and so on. But there are moments when we do not know the data and/or we do not know what we are looking for – we just want to find something new. Existing methods often fail if they do not have a goal set beforehand. In such a case, an explorative analysis (visual data mining) can be very useful. In general, it is the first step on the following path (see Figure 1):

- *Step 1*: In the case that data contains unknown information, it is visualised using different graphs or visualisation approaches. One can find, by inspecting them, something interesting and set it as an assumption.
- *Step 2*: In the case that there has been formulated an assumption (either resulting from the first step or simply from knowing a domain), this can be verified either by an experiment or by broader systematic data analysis. This verification can either support or refute the assumption and thus can be reformulated or accepted for the next step (e.g. as a new part of user or domain model).
- *Step 3*: In the case that it is already known what information is contained in the old data (either resulting from the first two steps or simply from knowing the data domain), there is a chance that it will be contained in the new data. To find it there, one can reason upon the live (streaming) data, and log only the found (derived) information – metadata.

In our work, we deal only with the first step of this (meta) information retrieval. Therefore, to maximize the success of a search, the visualisation has to follow the *type by task* taxonomy (TTT) of information visualisations proposed by Shneiderman [23]. It has its roots in the Visual Information Seeking Mantra: "Overview first, zoom and filter, then details-on-demand," and contains the following seven tasks:

- Overview: Gain an overview of the entire collection.
- Zoom: Zoom in on items of interest.
- Filter: Filter out uninteresting items.
- Details-on-demand: Select an item or group and get details when needed.

- Relate: View relationships among items.
- History: Keep a history of actions to support undo, replay, and progressive refinement.
- Extract: Allow extraction of sub-collections and of the query parameters.

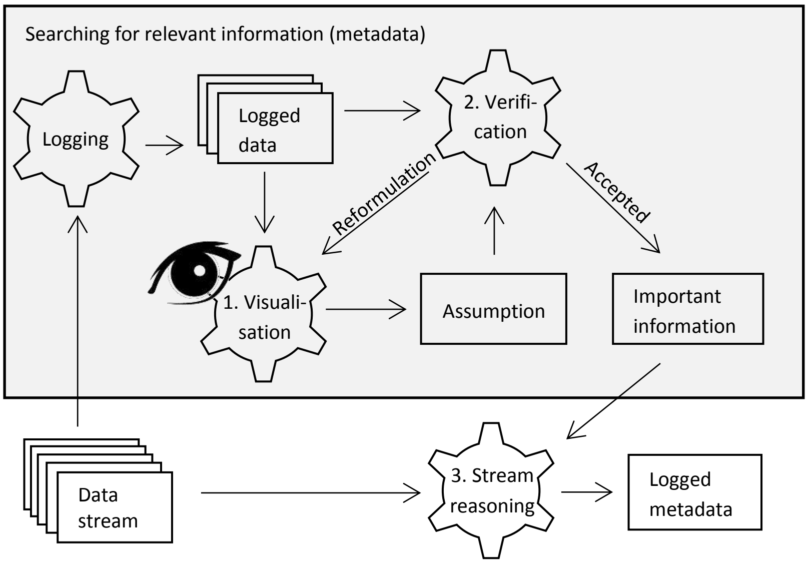


Figure 1
Exploratory analysis of logged data involving assumption formation based on visualisation

On the other side, the goal of a visualisation must be clear. Maletic, et al. [15] proposed that each software visualisation system supporting large-scale software development and maintenance has to have the following five dimensions. These dimensions reflect the why, who, what, where and, how questions to be addressed for the developed software visualisation:

- Tasks – *why* is the visualisation needed?
- Audience – *who* will use the visualisation?
- Target – *what* is the data source to represent?
- Representation – *how* to represent it?
- Medium – *where* to represent the visualisation?

In our case we answer these questions: Tasks – find new information in the data that could possibly improve software development, Audience – software development analysts, researchers, possibly project managers, Target – developers' interactions, Representation – graphs, Medium – computer monitor.

There are various approaches to visualisation in the domain of software development. Most of them visualise code structure [9] and/or source code metrics, and/or tasks (Gantt chart), e.g., Source Miner Evolution [17], YARN [8], Forest Metaphor [4], ClonEvol [7], GEVOL [3], EPOSee [2]. Some visualisations are even animated in time. DFlow [16] visualises developer's interactions during the development process, it is oriented only to navigating, writing, and understanding the source code and thus miss the wider context of developers' work. More complex tools, which are more similar to our solution, can be found in the time management domain, e.g., ManicTime[1] application.

3.1 Data Available on Software Developers' Activity

We monitor interaction events in tools that a software developer interacts with during their work. Although we may record even elementary events such as key presses or mouse movements that make our data big, we empirically selected only some of the events provided by tools (Microsoft Visual Studio, Eclipse, Git, Bash shell, and Mozilla Firefox). An interaction event is reported by a tool noting that a developer performed a meaningful operation. For each developer we also log which processes and applications were running in the operating system.

With a web browser we record *navigation* to URL addresses (through a link/URL bar/bookmark/other), *actions with tabs* (switch to/open/close), saving a *document* (and its name), or even creating a *bookmark* (with its name).

In an IDE we record interactions with source code *documents* (add, open, close, switch to, remove, save, rename), *projects* and *solutions* (add, open, close, switch to, remove, rename, refresh). We also record source code content interactions, specifically *code fragments manipulation* (copy, paste, cut, paste from a webpage) and *searching in source code* (searched expression, used search options, number of searched documents and results). We are also interested in a work to Git (commits) or Microsoft Team Foundation Server (check-ins). Because developers also use bash shell during their work, thus we record executed *bash commands*.

Recording of interaction events in tools is realized by custom plugins that communicate with local client application, called UACA, running on a developer's workstation. This application sends recorded data in chunks to a centralized repository using REST web services. With this approach we are able to monitor high quantities of interesting and very detailed data about software development, but still use it for evaluation, visualisation, or further processing. Further details on the PerConIK infrastructure can be found in [1].

Countless numbers of charts can be devised from the listed data and source code of monitored software projects. In this paper, we cannot discuss all the devised

[1] http://www.manictime.com/

data or graphs, and in any case, we did not employ all of them. However, to explain our visualisation approach in a simple and comprehensible manner, while still showing the potential of such visualisations, we include only charts that contain two activities – those of a developer interacting with a web browser and with an IDE. These are not further fragmented. We also did not include activities of bash commands because of their insufficient number. Answers to these questions can be found in such graphs: When and which developer was the most active? Is it always at the same time? Do all of the developers use a web browser and IDE by the same share? Is there a pattern or dependency at work in a web browser or an IDE? Which web sites were visited by developers the most? Is there a relation between the amount of time spent in a web browser during a work in an IDE, to the number of source code changes by a developer?

The graphs may also contain various metrics, e.g., for source codes: time spent by a developer typing them, studying them (reading without changing them), number of functions edited, how many times and how long a certain file was opened, the number of time the file was changed, etc. For web browsing: the ratio between the number of web pages related to work and private purposes, the ratio between the time spent on work vs. private web pages, the absolute number of web pages or absolute time spent on work/private web pages, correlation of these numbers with other quantities, the number of copied elements, etc.

Other potential graphs using mentioned metrics may answer the following questions: Can we categorize developers according to their read/edit/copy/paste behaviour? Can we evaluate their productivity? Is there any correlation between different types of activities that help us to indicate developers' experience, their strengths and weaknesses? Is there any harmful behaviour occurring in specific situations, e.g., introducing a bug by the end of a project? Is there any visible trend or pattern between the number of source code changes and other situations?

Answers to some of these questions can be found only if data is from a long enough period of time, which could take many years.

3.2 Graphs Devised from Software Developer's Activity

In order to identify any information from interaction data, it is necessary to combine events into higher level activities [12, 20] and then to visualise them. It is not easy to determine which actions form an activity. One way to do so is to apply a time threshold, as we describe in more detail in [22]. The visualisation graphs themselves should be transparent, readable and adaptable. We propose three kinds of graphs:

1) Timeline graphs, which depict selected type of activities – e.g., when and how long a developer worked in a certain application (see ManicTime graph in Figure 2); here it cannot easily be seen how many of them there are or how much time is spent in total. This type of graph can be found also in [16]

2) Scatter graphs – the dependency of selected activities (see Figure 6)
3) Column graphs – the cumulative numbers of the selected types of activities:
    a. One column is for one time step – e.g., it depicts number of web and IDE activities done within an hour; columns are ordered by time and have the same width (see Figure 3 and 4).
    b. One column is for one type of activity – e.g., it depicts the number of visits on stackoverflow.com domain; columns are ordered by height and have the same width.
    c. One column is for one block of activity – e.g., it depicts the number of changed lines in source code; columns are ordered by time and each block has a different width equal to the duration of a depicted block (see Figure 5)

Our implementation of the visualisation allows the user an interactive modifying time axis scale for each graph with a time axis. It is possible to choose time units to be used for data accumulation in the graph. By clicking on a column in a graph, a new graph with a finer time scale will open.

4 Evaluation

In order to be able to evaluate our proposed approach, we devised a prototype tool IVDA (Interactive Visualisation of Developers' Actions). It is implemented as a service that provides visualisations of logged data [22]. The visualisations are shown directly on the web browser, while computation takes place on the server.

There are tools that are able to monitor a software developer, but not to visualise their activity. There are also tools, on the other hand, that visualise software development but do not monitor the software developer so closely. Since our tool IVDA is no exception to this, some experiments are by design not fully supported.

For comparison with existing systems, we compared our tool with several other similar tools that were compared in [18]. Moreover, we also include the tool ManicTime in the comparison.

The tool is not from the domain of software development, but from the more general domain of time management. It monitors computer usage in any work. It is oriented toward applications or environments used, and documents (context) that the user is working within the given application (see Figure 2). The tool monitors how long activities lasted. Recorded data can be seen on the time axis. They serve the user as an overview of their work with the computer. The tool also offers cumulative results, albeit only in numerical form.

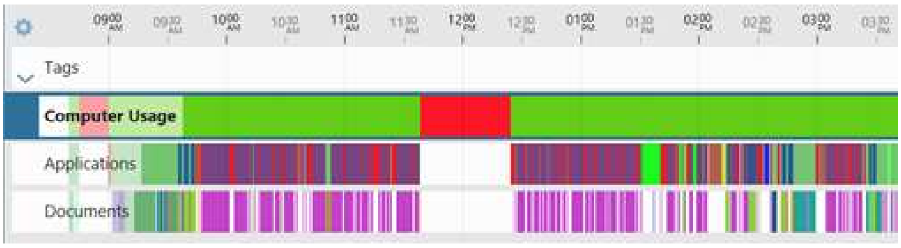


Figure 2
Timeline in ManicTime interface

See Table 1 for results of the comparison with our IVDA tool with existing tools mentioned in [18] that monitor developers' work, along with ManicTime. The comparison is done by classification based on workspace awareness elements and does not take into account visualisation ability. The number of individual elements across all 14 tools represents the need of developers to know that information. As we can see, IVDA offers the five most wanted elements. ManicTime offers three of them, although this tool is not specific to the domain of software development.

Table 1
Omoronyia, et al. [18] classification of visualisation tools with added IVDA and ManicTime

Tool	Identity	Location	Activity level	Actions	Intentions	Changes	Objects	Extents	Abilities	Influence	Expectations
TagSEA	x				x	x	x				
Jazz	x				x	x	x	x		x	
Expertise browser	x	x				x			x	x	
Sysiphus	x	x			x	x					
Hipikat	x										
Palantír	x		x			x	x	x			
FASTDash	x	x		x		x	x				
Team tracks			x							x	x
CASS	x				x	x	x				
Augur	x				x	x	x				
Ariadne	x							x			
Mylyn		x	x	x		x	x				x
ManicTime	**x**	**x**	**x**								
IVDA	**x**	**x**	**x**			**x**	**x**				
14 tools	12	6	6	2	5	10	8	3	1	3	2

To proceed with the evaluation of our approach, we performed a user test aimed at its visualisation quality. The tools from Table 1 do not provide visualisation as IVDA or ManicTime. Therefore we needed to choose other tools. As we already mentioned, there are tools, which visualise code structure and/or source code metrics, and/or tasks [2, 3, 4, 7, 8, 9, 16, 17], but they do not visualise programmer's activities with their context. This led us to find a tool for visualisation of activities, but from another domain, e.g., ManicTime, and to compare it with IVDA. The test included 10 test cases (TC) of differing difficulty (see Table 2). We conducted this test with 7 participants (4 males and 3 females) ages 19-35, with a university degree in an engineering field. None of them had previous experience with visualisations or tools that monitor time spent with a computer, and they use a web browser for 2 to 8 hours a day.

Table 2
10 test cases (TC1-TC10) conducted for evaluation of the IVDA tool

TC1	Inspect activity of any developer over the previous week.
TC2	Determine by inspection, for any developer, what activities prevailed during the previous day.
TC3	Find out which processes were run on the user's computer on a given day.
TC4	During which part of the last 3 days was the developer most productive?
TC5	Which files and environments did the developer work with yesterday?
TC6	Which particular file has been the most frequently modified one by a developer during the whole previous year?
TC7	Did the developer write source code over the last week more frequently in the mornings or in the evenings?
TC8	For how long did the developer use the browser after 11 o'clock?
TC9	Try to identify some habit in the developer's behaviour within the last month.
TC10	There are a developer's activities that do not relate to the actual development. If they indeed occur, find out when they started, how long they lasted and what is their nature. Choose a single August day in 2014.

Table 3 shows the success rates of tasks completion (within the allotted time limit) – the test participants completed most of the required tasks on time for both of the compared tools. Moreover, we noticed that the test participants frequently had problems with initial tasks, which took them longer than expected. This may be caused by a lack of intuitiveness of the tool in the initial phase. Keeping the main goal of this research in mind, i.e., to facilitate explorative analysis, the TC9 task was the most critical one. Our IVDA tool has higher potential to analyse monitored and visualised data, and to seek new information from it. Moreover, IVDA offers information, which other visualising tools do not. One of the interesting feedbacks from one participant was that she became curious after experiments and started to explore the data herself to answer her own questions.

In Figure 3, the graphs reveal information about the logged developer "Puma". It appears "Puma" mostly writes code during working days. There are a few days when "Puma" works really hard (maybe refactoring some code since a lot of lines of the code were changed). But there are also days (and even weeks), when "Puma" uses only the web browser. The gap in August suggests that Puma was on vacation.

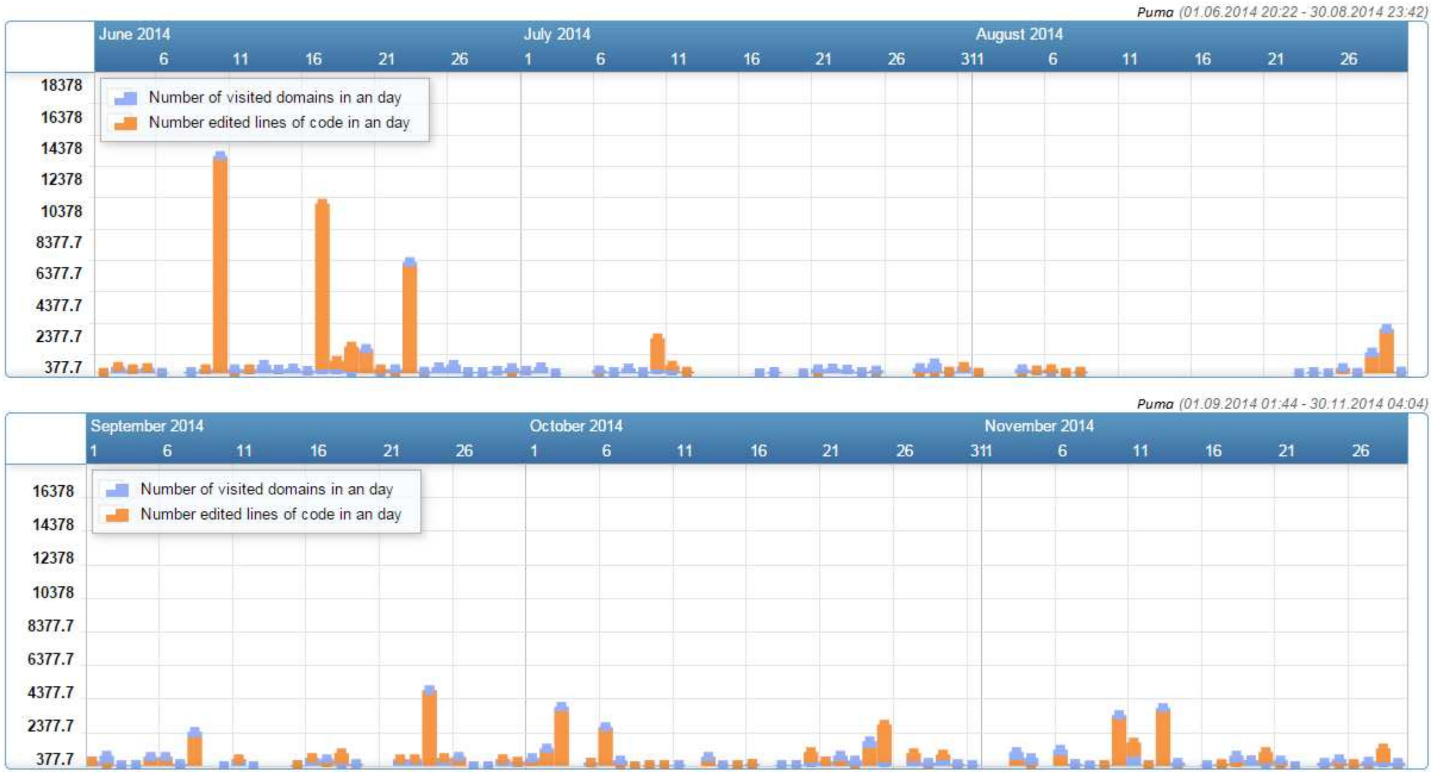


Figure 3

Overview of work by a developer "Puma" within a timeframe of 6 months –
columns in graphs depict cumulated number of visited domains and edited lines of code per day

Table 3

Success rate of tasks TC1-TC10 completion in IVDA and ManicTime (%)

	TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10
IVDA	85	100	85	100	57	100	100	85	100	100
ManicTime	100	100	100	57	100	85	85	85	57	71

Graphs in Figure 4 show that different developers work differently during the same time period. “Jaguar” edited hundreds lines of code during selected days. “Puma” worked a lot with a browser – probably searching for something. In both graphs it is clearly visible that both of them are more active during the day and resting during the night. No regularities like lunch breaks, meetings, etc. can be found in this part of a year, but in our dataset there are other time periods, where such events are visible.

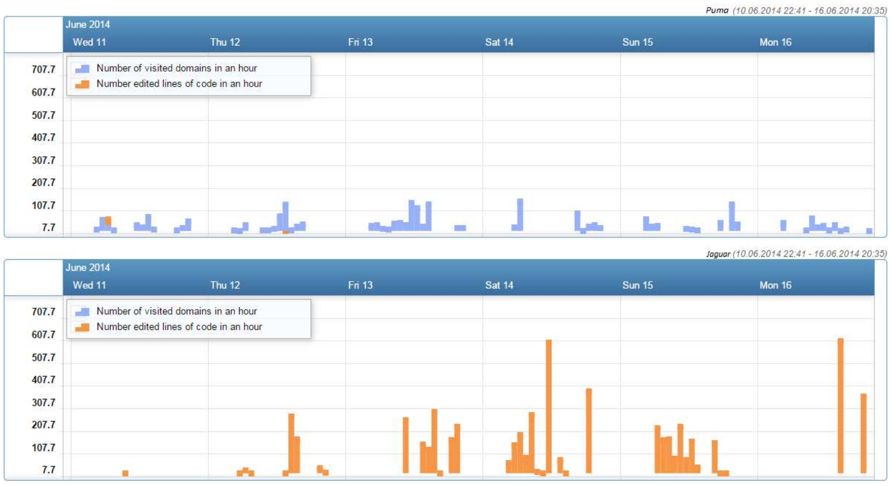


Figure 4

Comparison of two developers “Puma” and “Jaguar” over one week –the number of visited domains together with the number of edited lines, both cumulated per hour

Graphs in Figure 5 depict columns of different widths since the activities are cumulated per continuous activity (either continuous in a web browser or continuous in IDE). This type of visual representation helps the explorer quickly find the most active periods in a developer’s working day – the highest and the widest columns. As the example of the developer “Puma” shows, developers' days can vary widely – there is a different time of day for the highest number of edited lines of code, different amounts of visited domains and different ratios of these two activities. Although these three days look so different, they have also something in common – the Pareto principle also plays a role here: it looks like the most active periods cover 20% of the developer’s working day and during these peaks, 80% of their activities are completed.

In another type of graph one can try to find an answer to the following question: "Does the number of edited lines of the code depend on the time the developer spent using a web browser when writing it?" As we can see in Figure 6, there is no dependency. However, we can postulate that when this developer spends more than 20 minutes using a browser, they are probably not writing a code.

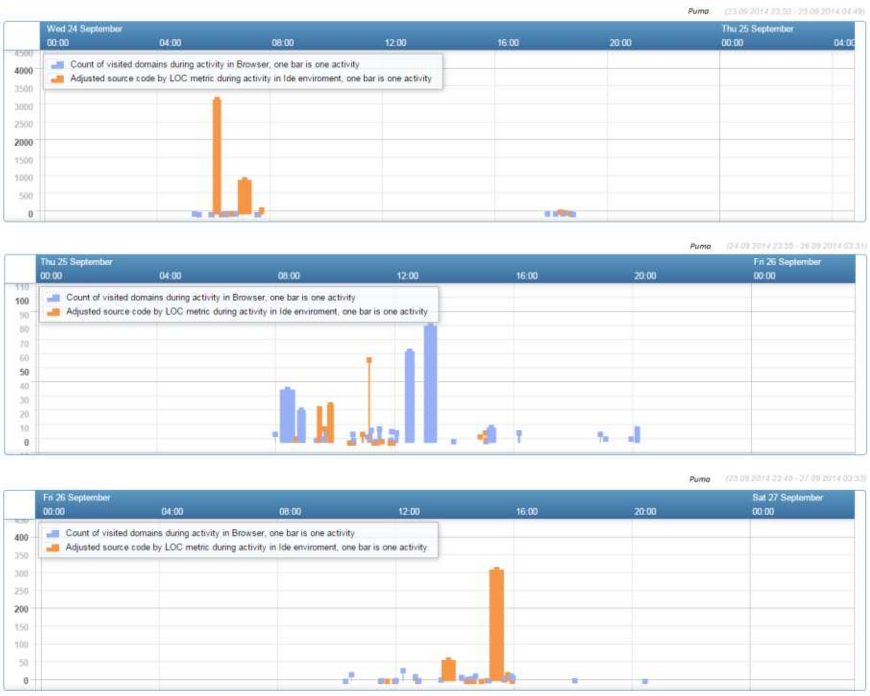


Figure 5

Number of visited domains together with number of edited lines by developer "Puma", both cumulated per continuous activity, within three subsequent days

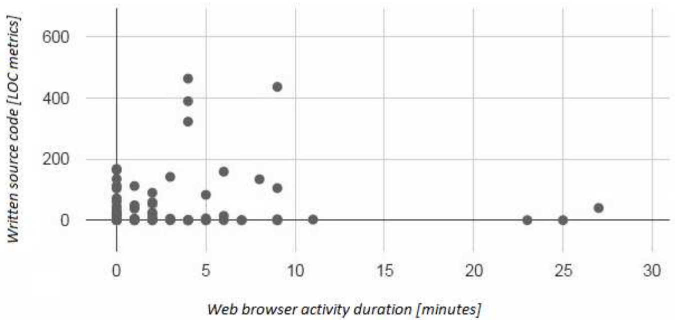


Figure 6

Scatter graph, where every point represents the duration of the web browser activity which was immediately followed by a depicted amount of source code changes in IDE

Conclusions

The work presented in this paper contributes to the methods of better understanding a software developer's activities. Understanding them is vitally important during software evolution. In development, a project manager can better reassign tasks and allocate resources. In maintenance, any revealed code dependencies suggest places for a possible remedy. Activities emerge as high level outcomes of exploratory analysis of low level data, logging a developer's rather elementary actions. These elementary actions are such that many of them occur within small periods of time. As a result, logging them also requires some kind of pre-processing, since without reduction, storing them would be too prohibitive. This is essentially one of the characteristics of Big Data.

Software development is a creative process, which together with maintenance are the two most extensive elements of a software project. A software developer, as an executor committed to bringing specifications and ideas into usable product, is often difficult to monitor, evaluate and reason upon identified information, in order to support the particular software project meeting the deadlines on time and in acceptable quality. As we pointed out, this motivates current research in software engineering to employ new approaches for monitoring software development directly on the level of interaction events performed by developers in tools that they use during their work.

Our contribution is to transform low level logged data into higher level information on activities and then to attempt various schemes of visualisation in order to facilitate better understanding of the data. We employ interactive visualisation in a customizable manner to find answers to various interested questions of team leaders or software developers. We understand visualisation as a tool to open the way to explorative analysis of a software developer's activity. We described explorative analysis as a three step process, depicting the importance of a proper visualisation tool. We devised a number of simple graph schemes. They visualise the partially processed data and allow a human analyst to make assumptions by generalising what they see in visualising graphs. We applied the devised graph schemes to data gathered within the project PerConIK that we have been participating in over the last four years. We devised and implemented the interactive visualisation tool IVDA. User-defined graphs generated by the tool show information about activities, developers' actions and their metrics, which a team leader may use to identify possible causes of problems, delays in delivering results, anomalies and trends in activities of developers in a team. Furthermore, this tool may help any scientist to endorse or refute assumptions about a software developer's activity. By inspecting the visualised data, the analyst is able to gain much useful information on the software development of a particular project.

Since data that can be received by logging is way below the level any analysis of software development should be performed, one challenge for future work is to further automate identification of developers' activities from interactive events.

Such identification, especially when accomplished in real-time during a developer's work, may support not only a team leader or scientist in answering hypotheses, but also developers themselves before committing their results to revision control systems or when describing completed tasks in task management systems. Developers often work on several development tasks, more or less arbitrarily, that may tangle them into composite change (commit) in the end. Such tangled changes in an RCS are difficult to review, describe, or merge with other changes. Existing approaches attempt to identify and untangle such changes by analysing a static snapshot of commit history. However, we argue that the approach of doing so in real-time is required, to prevent a software developer from tangling their changes even before committing them to an RCS.

Acknowledgement

This work was supported by the Research and Development Operational Programme for the project Research of methods for acquisition, analysis and personalized conveying of information and knowledge, ITMS 26240220039, co-funded by the ERDF. This work was also partially supported by the Scientific Grant Agency of The Slovak Republic, Grant No. VG 1/0752/14.

References

[1] Bieliková, M., Polášek, I., Barla, M., Kuric, E., Rástočný, K., Tvarožek, J., Lacko, P.: Platform Independent Software Development Monitoring: Design of an Architecture. In: Proc. of 40th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2014) Springer-Verlag, 2014, pp. 126-137

[2] Burch, M., Diehl, S., Weissgerber, P.: EPOSee – A Tool For Visualizing Software Evolution. In: Proc. of the 3rd IEEE International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2005) IEEE, 2005, pp. 1-2

[3] Collberg, C., Kobourov, S., Nagra, J., Pitts, J., Wampler, K.: A System for Graph-based Visualization of the Evolution of Software. In: Proc. of the 2003 ACM Symposium on Software Visualization (SoftVis '03) ACM, 2003, pp. 77-86

[4] Erra, U., Scanniello, G., Capece, N.: Visualizing the Evolution of Software Systems Using the Forest Metaphor. In: Proc. of the 16th International Conference on Information Visualisation (IV 2012) IEEE, 2012, pp. 87-92

[5] Garzo, A., Benczur, A. A., Sidlo, C. I., et al.: Real-time Streaming Mobility Analytics. In: Proc. of the 2013 IEEE International Conference on Big Data, IEEE, 2013, pp. 697-702

[6] Gu, Z., Schleck, D., Barr, E. T., Su, Z.: Capturing and Exploring IDE Interactions. In: Proc. of the 2014 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software (Onward! 2014) ACM, 2014, pp. 83-94

[7] Hanjalic, A.: ClonEvol: Visualizing Software Evolution with Code Clones. In: Proc. of the 1st IEEE Conference on Software Visualization (VISSOFT 2013) IEEE, 2013, pp. 1-4

[8] Hindle, A., Ming Jiang, Z., Koleilat, W., Godfrey, M. W., Holt, R. C.: YARN: Animating Software Evolution. In: Proc. of the 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2007) IEEE, 2007, pp. 129-136

[9] Kapec, P.: Knowledge-based Software Representation, Querying and Visualization. In: Information Sciences and Technologies. Bulletin of the ACM Slovakia, Vol. 3, No. 2, Slovak University of Technology Press, Bratislava, Slovakia, 2011, pp. 1-11

[10] Kersten, M., Murphy, G.C.: Using Task Context to Improve Programmer Productivity. In: Proc. of the 14th ACM SIGSOFT International Symposium on Foundations of Software Engineering (SIGSOFT '06/FSE-14) ACM, 2006, pp. 1-11

[11] Konôpka, M., Bieliková, M.: Software Developer Activity as a Source for Identifying Hidden Source Code Dependencies. In: Proc. of the 41st International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2015) Springer-Verlag, LNCS 8939, 2015, pp. 449-462

[12] Konôpka, M., Návrat, P.: Untangling Development Tasks with Software Developer's Activity. To appear in: Proc. of the 2nd International Workshop on Context for Software Development (CSD 2015) in companion with the 37th International Conf. on Software Engineering (ICSE 2015) IEEE Press, Florence, Italy, 2015, 2 p.

[13] Kuric, E., Bieliková, M.: Estimation of Student's Programming Expertise. In: Proc. of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '14) ACM, 2014, Article No. 35

[14] Maalej, W., Fritz, T., Robbes, R.: Collecting and Processing Interaction Data for Recommendation Systems, in Recommendation Systems in Software Engineering, Robillard, M. P., Maalej, W., Walker, R.J., Zimmermann, T. (Eds.) Springer Berlin Heidelberg, 2014, pp. 173-197

[15] Maletic, J. I., Marcus, A., Collard, M. L.: A Task Oriented View of Software Visualization. In: Proc. of the 1st Int. Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2002) IEEE, 2012, p. 32

[16] Minelli, R., Mocci, A., Lanza, M.: Visualizing Developer Interactions. In: Second IEEE Working Conference on Software Visualization (VISSOFT 2014) IEEE, 2014, pp. 147-156

[17] Novais, R., Lima, C., de F. Carneiro, G., Paulo, R. M. S., Medonca, M.: An

Interactive Differential and Temporal Approach to Visually Analyze Software Evolution. In: Proc. of 6th IEEE International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2011) IEEE, 2011, pp. 1-4

[18] Omoronyia, I., Ferguson, J., Roper, M., Wood, M.: Using Developer Activity Data to Enhance Awareness during Collaborative Software Development. In: Computer Supported Cooperative Work (CSCW) Vol. 18, Issue 5-6, Springer Netherlands, 2009, pp. 509-558

[19] Rástočný, K., Bieliková, M.: Enriching Source Code by Empirical Metadata. In: Proc. of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '14) ACM, 2014, Article No. 67

[20] Roehm, T., Maalej, W.: Automatically Detecting Developer Activities and Problems in Software Development Work. In: Proc. of 33rd International Conference on Software Engineering (ICSE 2012) IEEE, 2012, pp. 1261-1264

[21] Roehm, T., Gurbanova, N., Bruegge, B., Joubert, C.: Monitoring User Interactions for Supporting Failure Reproduction. In: Proc. of the 21st IEEE International Conference on Program Comprehension (ICPC 2013) IEEE, 2013, pp. 73-82

[22] Sekerák, L.: Interactive Visualization of Developer's Actions. In: Proc. of the 11th Student Research Conf. on Informatics and Information Technologies (IIT.SRC 2015) Slovak University of Technology Press, Bratislava, Slovakia, 2015, pp. 281-286

[23] Shneiderman, B.: The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In: Proc. of the 1996 IEEE Symposium on Visual Languages, IEEE, 1996, pp. 336-343

[24] Zhang, J., Huang, M. L.: 5Ws Model for Big Data Analysis and Visualization. In: Proc. of 16th IEEE International Conference on Computation Science and Engineering (CSE 2013) IEEE, 2013, pp. 1021-1028

Adaptive Collaborative Filtering Based on Scalable Clustering for Big Recommender Systems[*]

O-Joun Lee[†], Min-Sung Hong[†], and Jason J. Jung[‡]

School of Computer Engineering, Chung-Ang University
Heukseok-Dong, Dongjak-Gu, 156-756, KS013, Seoul, Korea
E-mail: {concerto34, minsung87, j3ung}@cau.ac.kr

Juhyun Shin

Department of Control and Measuring Robot Engineering, Chosun University
Seoseok-dong, Dong-gu, 501-759, KS008, Gwangju, Korea
E-mail: jhshinkr@chosun.ac.kr

Pankoo Kim

Department of Computer Engineering, Chosun University
Seoseok-dong, Dong-gu, 501-759, KS008, Gwangju, Korea
E-mail: pkkim@chosun.ac.kr

Abstract: The large amount of information that is currently being collected (the so-called "big data"), have resulted in model-based Collaborative Filtering (CF) methods to encountering limitations, e.g., the sparsity problem and the scalability problem. It is difficult for model-based CF methods to address the scalability-performance trade-off. Therefore, we propose a scalable clustering-based CF method in this paper that can help provide a balance by re-locating elements in the cluster model. The proposed method is evaluated by performing a comparison against existing methods in terms of measurements for the Mean Absolute Error (MAE) and response time to assess the performance and scalability. The experimental results show that the proposed method improves the MAE and the response time by 50.79% and 48.25%, respectively.

[*] This paper is significantly extended from an earlier version presented at the 3rd International Conference on Smart Media Applications in December 2014.

[†] These authors contributed equally to this work as the first author.

[‡] Corresponding author.

Keywords: Big data; Recommender System; Adaptive System; Clustering-based Collaborative Filtering; Scalable System

1 Introduction

Collaborative filtering (CF) can be used to find a set of the relevant items that are assumed to be the most appropriate for a target user. Most CF approaches have analyzed user ratings to discover the various relationships between users, between items, and between users and items [1-4]. Since these methods mainly focus on collecting more user ratings (without integrating much external knowledge of a specific domain), we have realized that they have some fundamental limitations. The first limitation is the sparsity problem (which is also called the cold-start problem and first-users/items problem). If the density of the user rating matrix is too low and cannot represent users' preferences, the performance of the CF method will decrease [2].

The second limitation lies in the scalability problem (particularly, the big data issue). As the number of users and the number of items in a CF-based recommender system increase, the computation time to build user/item subsets exponentially increases [5].

A number of methods have been proposed to solve these problems, and these methods can be categorized into two main groups: model-based CF methods and hybrid CF methods [6-7]. Table 1 shows the advantages and disadvantages of these [8].

Table 1
Advantages and disadvantages of model-based CF and hybrid CF

	Advantages	Disadvantages
Model-based CF	· better addresses the sparsity, scalability and other problems · improves performance	· expensive model-building process · trade-off between performance and scalability
Hybrid CF	· improves performance · overcomes CF problems, such as sparsity and gray sheep	· increased complexity · needs external information that is usually not available

As can be seen, these cannot solve all the fundamental problems of the CF method, i.e., data sparsity problem and scalability problem. For the Model-based CF, these methods exhibit a trade-off between the predictive performance and the scalability since a reduced coverage results in a rating table that is relatively sparse. Furthermore, the cold-start problem still exist. The cold-start problem and the first-user/item problem are simply caused by the absence of 1data, so the model-based CF cannot be a fundamental solution.

Hence, the various hybrid CF methods that combine model-based CF and Content-Based Filtering (CBF) have been proposed in order to address these problems. However, most of these methods are too complicated to implement and external knowledge or data are required. If an excess of external data is necessary, then the applicable domain for the system will be restricted.

This paper proposes Adaptive Collaborative Filtering Based on Scalable Clustering (ACFSC) which is focusing on solving scalability problem by reducing time complexity to compose neighborhood. Also, it improves the data sparsity problem by making users' and items' feature vectors incrementally learning.

This method adopts three major policies: the use of minimal external data, combining, and re-locating. First, to maximize the adaptable domain of the system, we use the minimal external data, such as the user profile and the item metadata. Second, rating data and external data are combined to solve the cold-start problem and the first user/item problem. Third, newly-arrived rating data is used to re-locate users and items to form more appropriate clusters in order to solve the reduced coverage issues.

Based on the fore-mentioned policies, implementation of this method is composed of four parts, as shown in Fig. 1. The parts interact with each other following a cyclic architecture. It makes user ratings gradually improve the cluster models.

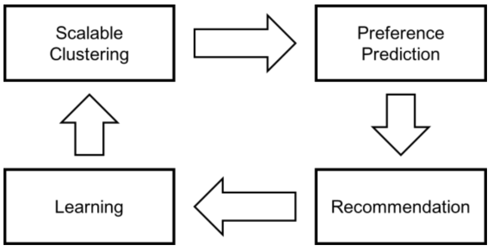


Figure 1
Conceptual Composition of the ACFSC

1) *Scalable clustering*: this step creates a cluster model for users/items that is based on feature vectors of them. It reduces the time complexity to produce user/item subsets. Feature vectors of these are composed by combining user profile data, item metadata, and user ratings. This step is intended to solve the cold-start and first-user/item problems.

2) *Recommendation*: this step recommends the top-N items that are selected according to the preference of the users who are included in the same user subsets. In order to minimize the system load, most tasks to create subsets proceed as in the first step.

3) *Preference prediction*: this step predicts users' missing preference information by using clusters of users and items to resolve the problems caused by the sparsity of the user rating matrix.

4) *Learning*: this step resolves the difficulty in quantifying the qualitative characteristic of the users and items through the use of learning feature vectors of users and items.

The remainder of this paper is organized as follows. In Section 2, we present an outline of related work. In Section 3, we introduce the Scalable clustering method to create the recommendation model as described in the first step above. In Section 4, the other steps are described to implement the Scalable Collaborative Filtering techniques. In Section 5, we evaluate the performance by comparing the results for the proposed method against those of others, and we discuss the results of the experiments. Finally, the conclusion provides a summary of the proposed algorithm and outlines future work.

2 Related Work

2.1 Model-based CF

In order to solve the scalability problem, various model-based CF methods based on machine learning or data mining models have been proposed, e.g., Bayesian belief nets, clustering models, latent semantic indexing, sparse factor analysis, and dimensional reduction. These use rating data to estimate or learn a model to predict users' preferences and can partially mitigate the scalability problem and the sparsity problem and improve the performance of the recommender systems. However, these have not yet been able to overcome the trade-off between performance and scalability and the cold-start problem still exists. The cold-start problem and the first-user/item problem are caused by the absence of data. Thus, model-based CF is not enough to solve them. For the moment, we introduce two representative approaches for these methods: clustering-based methods and dimensional reduction-based methods.

Clustering-based methods that use a cluster model to reduce the time complexity have been proposed. These methods build a cluster model by using correlations and similarities and use clusters that are built as subsets of users or items. Li and Murata [22] presented a CF method that is based on multi-dimensional clustering which involves clustering user/item profiles that are generated as background data, followed by clustering pruning and preference prediction through the weighted average of neighbors. This method has an advantage in that it maintains the balance in the performance of the recommendations, even when a growing diversity of items is handled. However, it still has the same limitations as the model-based CF method. Bellogin and Parapar [23] implemented in a normalized cut (N-Cut), a graph cut-based clustering solution, to facilitate the formation of similar user groups. Despite the improvement in performance over existing CF methods, the solution was unable to address the reduced coverage issue. Although these kinds of methods are effective in solving the scalability problem of CF, they have two additional limitations. First, the costs of building cluster models are higher than for CF, so the models are hard to update dynamically. Second, since the target areas to compose the user/item subsets are restricted by the clusters, a reduced coverage problem occurs and the performance of the system deteriorates.

Dimensional reduction-based methods have also been proposed. These methods aim to cancel the noise in the rating matrices. Buried correlation information is generated between the users and the items underline. Zhong and Li [10] suggested a unified method that combines the latent and external features of users/items to ensure the accuracy in the predictive preference. A probabilistic latent semantic analysis is used to extract the latent features of the historical rating data. Luo et al. [11] implemented an incremental CF recommender system based on Regularized Matrix Factorization. This method supports incremental updates for the trained parameters as new ratings arrive. These methods partially solve the sparsity problem, including the cold-start problem and the first-user/item problem. However, the scalability problem becomes aggravated.

2.2 Combining Model-based CF and Content-based Filtering

The data sparsity problem in the model-based CF can be addressed through hybrid CF methods, which have been widely studied by combining model-based CF and Content-Based Filtering (CBF). These methods use external data to address the cold-start problem and the first-user/item problem of the model-based CF.

Parallel methods that use model-based CF and CBF at the same time have been suggested. These methods print out top-N items by integrating the results of model-based CF and CBF. Park et al. [13] suggested single-scaled hybrid filtering that uses a weight decided through an experiment, and this method showed a slight improvement in performance. Shen et al. [14] presented a hybrid filtering method that applied an optimized weight. This method used a simple learning algorithm based on user feedback to find more optimized weights. These methods are effective in solving the remaining problems for model-based CF. However, these methods do not provide improvements in terms of the trade-off between performance and the scalability, and also introduce two additional problems. First, they depend on the CBF at the initial time and for the first-user/item, so the overall performance can be comparatively decreased. Second, they need to dynamically update the weights as the data grows, but it is difficult to dynamically allocate time to update or optimize the system.

At the same time, other methods have been proposed to use external data to build corresponding models. These methods initially use external data when rating data does not yet exist, and the system functions by using rating data after the model has been built. Cho et al. [9] proposed a map-based personalized recommender system that uses Bayesian Networks built by an expert with a parameter that was learned from the dataset. Contextual information was collected (e.g., location, time, weather), and the user request was provided a mobile device. Campos et al. [17] employed Bayesian networks to combine the characteristics of both CBF and CF and to generate more accurate recommendations by using probabilistic reasoning to compute the probability distribution over the expected rating. These methods are remarkable in solving the cold-start problem. However, it is hard to find external data that can be adapted to the appliance domain. If the external data is not adaptable, the performance of the system will severely decline.

On the other hand, other methods that integrate external data with user rating data have been presented. These methods use external data as the rating data or transform the external data into rating data. Bogers and Bosch [16] combined CF and CBF by using tags in social bookmarking websites. They examined how to incorporate tags and other metadata into a hybrid CBF/CF algorithm by overlapping the traditional user-based and item-based similarity measures with the tags. Hu and Pu [18] proposed a method that combines the personality characteristics of the users into traditional rating-based similarity computations for user-based collaborative filtering systems. Kim et al. [19] proposed a new approach to model users in a collaborative manner by using user-generated tags. This can be exploited in a recommender system by leveraging user-generated tags as preference indicators. These methods therefore effectively improve the sparsity problem. However, logistic evidence to transform external data as rating data are insufficient, and in addition, the scalability problem is not solved.

3 Scalable clustering with Data Streams

The scalable clustering method is conducted in three steps. In the first step, each cluster model for users and items is created according to their feature vectors. This step creates their clusters using the Expectation Maximization (EM) algorithm. The maximum likelihood is estimated based on the Gaussian-Bayesian Probabilistic Model and the cosine similarity, and it is formulated as

$$L_{C_i,x_j} = f\left(x_j, \mu_i, \sigma_i\right) \times \frac{1}{N_{C_i}} \sum_{l=1}^{N_{C_i}} \frac{R_j \cdot R_l}{\| R_j \| \, \| R_l \|}, \tag{1}$$

where C_i indicates the i-th cluster, x_j indicates the j-th element, μ_i and σ_i represent the average and the standard deviation of the elements in the i-th cluster C_i, and R_j is a rating vector of element j. N_{C_i} represents the number of elements included in the i-th cluster. $f\left(x_j, \mu_i, \sigma_i\right)$ indicates the probability that the j-th element is included in the i-th cluster. L_{C_i,x_j} indicates the maximum likelihood of the j-th element for the i-th cluster.

In the second step, the inter-cluster preferences between each of the user-clusters and item-clusters are estimated. This indicates the representative value of the preference of a particular user cluster for particular item-cluster, and it is formulated as

$$CP_{i,j} = \sum_{l}^{N_{uc_i}} \sum_{n}^{N_{cc_j}} W_{UC_i,u_l} \times W_{CC_j,c_n} \times R_{u_l,c_n}, \tag{2}$$

where UC_i and CC_j indicate the i-th user-cluster and the j-th item-cluster. u_l and c_n indicate the l-th user and the n-th item. R_{u_l,c_n} indicates the rating for user u_l and item c_n, W_{UC_i,u_l} represents the likelihood of the user u_l for the user-cluster UC_i, and W_{CC_j,c_n} is the likelihood of the item c_n for the item-cluster CC_j. $CP_{i,j}$ is the inter-cluster preference for the user-cluster UC_i for item-cluster CC_j.

In the third step, a user-item preference matrix is created. This matrix marks the preferences that are predicted using the proposed method, and it is different from the rating matrix that simply marks the rating score entered by the users. The prediction of the user-item preference is estimated according to the inter-cluster preferences and the user ratings, and it is formulated as

$$P(i_j|u_i)(predicted) = P(UC_l|u_i)P(IC_m|i_j)CP_{l,m}R_{i,j}, \tag{3}$$

where u_i and i_j indicate the i-th user and the j-th item, the user-cluster UC_l includes the u_i, and the item-cluster IC_m includes the i_j. In addition, $P(UC_l|u_i)$ is the probability that u_i belongs to the UC_l, and $P(IC_m|i_j)$ is the probability that i_j belongs to the IC_m. $CP_{l,m}$ is the inter- cluster preference of the UC_l for the IC_m. $P(c_j|u_i)(predicted)$ represents the predicted preference for the u_i corresponding to i_j.

4 Scalable Collaborative Filtering

The scalable collaborative filtering is composed of three modules. 1) The first module recommends the top-N items based on the user-item preference matrix, 2) the second module predicts the missed rating scores in the rating matrix, and 3) the third module enables the feature vectors of the users and the items that are learned by using feedback from the users.

4.1 Recommendation Module

This module chooses the top-N items in order to recommend a particular user. To improve scalability, a ranking of the item-clusters is referenced for a user-cluster that includes the user. This process consists of two steps.

1) In the first step, the item-clusters are ordered according to the inter-cluster preferences for the user-cluster that includes the target user.
2) In the second step, the items are added to the top-N list sequential search of the ordered item-clusters. The items that previously received a high preference are selected, and if no items received a preference in the search range, the search range is shifted to the next item-cluster.

4.2 Preference Prediction Module

This module predicts the missed rating scores through a hybrid preference prediction. The prediction is performed by using the weighted average of two widely used prediction methods: user-oriented prediction and item oriented prediction. This process is composed of three steps.

In the first step, similarity matrices are created for both the users and the items. Each component of the matrices has a similarity between the users or items, and the similarity is measured by using a cosine coefficient of the rating vectors.

In the second step, the two previously mentioned prediction methods make each of the prediction results by using the weighted average of the similarities. This step is formulated as

$$p_{a,m}(u) = \overline{R_a} + \frac{\sum_{n \in UC_a} (R_{n,m} - \overline{R_a}) \times uw_{a,n}}{\sum_{n \in UC_a} uw_{a,n}} \tag{4}$$

$$p_{a,m}(i) = \frac{\sum_{l \in CC_m} R_{a,l}\, cw_{m,l}}{\sum_{l \in CC_m} cw_{m,l}}, \tag{5}$$

where UC_a indicates the user-cluster that includes user a, and CC_m indicates the item-cluster that includes item m. $uw_{a,n}$ and $cw_{m,l}$ represent the similarities between users a and n and between items m and l. $\overline{R_a}$ represents the average of the ratings inserted by user a. $R_{n,m}$ and $R_{a,l}$ indicate ratings of user n for the item m and of user a for the item l, respectively. $p_{a,m}(u)$ and $p_{a,m}(i)$ indicate the predicted rating scores for user a and item l estimated by the user-oriented and item oriented method, respectively.

In the third step, the hybrid preference prediction method deducts the predicted rating score by combining the results of the preceding step according to the weighted average. The weighting is dynamically decided based on the consistency of the source datasets. This step is formulated as

$$p_{a,m}(hybrid) = \alpha \times p_{a,m}(u) + (1 - \alpha) \times p_{a,m}(i) \tag{6}$$

$$\alpha = \sigma(CC_m)/\sigma(UC_a) + \sigma(CC_m), \tag{7}$$

where $\sigma(UC_a)$ and $\sigma(CC_m)$ are standard deviations of the clusters that include user a and the item m, respectively. $p_{a,m}(hybrid)$ is the predicted rating score for user a and item m, and α is the dynamic weighting.

4.3 Learning Module

This module provides the feature vectors of users and items that are learned according to the rating data. This process consists of three steps: normalizing, learning, and re-locating.

In the first step, the newly-arrived ratings are normalized by the Gaussian Probability Model that is composed of the historical user ratings. Since standard points and measures of the rating scores vary across individuals, we need to unify these. This step is formulated as

$$preR_{a,m} = (F_{a,m} - \overrightarrow{F_a})/\sigma(F_a) \tag{8}$$

$$R_{a,m} = \begin{cases} 1, \; if \; preR_{a,m} \geq 10 \\ preR_{a,m}/20 + 0.5, \; if \; |preR_{a,m}| < 10 \\ 0, \; if \; preR_{a,m} \leq -10 \end{cases} \tag{9}$$

where $F_{a,m}$ represents a newly arrived rating for user a and item m, $\overrightarrow{F_a}$ and $\sigma(F_a)$ are the average and standard deviation of the historical ratings for user a, respectively, $preR_{a,m}$ indicates the preprocessed and non-normalized rating, and $R_{a,m}$ indicates the normalized rating.

In the second step, mutual complementary learning between the feature vector of the user who inserted the rating and the item which was rated by user is performed. This step uses the normalized rating as a weighting and is formulated as

$$\overrightarrow{UV_a} = R_{a,m} \times CV_m + (1 - R_{a,m}) \times pre\overrightarrow{UV_a} \tag{10}$$

$$\overrightarrow{CV_m} = R_{a,m} \times UV_a + (1 - R_{a,m}) \times pre\overrightarrow{CV_m}\,, \tag{11}$$

where $\overrightarrow{UV_a}$ and $\overrightarrow{CV_m}$ are feature vectors for user a and item m.

In the third step, the user who inserted the rating and the item which is rated by user are re-located to more suitable clusters. This step does not rebuild the cluster model, but rather just searches the clusters with greater likelihood following the change in the feature vectors.

5 Evaluation and Discussion

In this section, we present the results for two experiments that aimed to investigate two different issues: the cold-start problem and the scalability problem. In the first experiment, ACFSC is compared against existing methods to assess whether the proposed method provides better performance with a sparse rating table. Our conjecture is that the ACFSC should show better performance during the initial stage, and the gap between the ACFSC and the other methods should subsequently become smaller.

In the second experiment, ACFSC is compared against existing methods to verify whether these have an improved robustness for a high-load environment. Our hypothesis is that when a larger scale is obtained, this system should be more robust than the others. In summary, the experiments address the following research questions.

- Q1: Can ACFSC improve the performance of an initial system that has an extremely sparse rating matrix?
- Q2: Is ACFSC robust for use in the high-load environment of a large-scale system?

5.1 Experimental Environment

The experiments were conducted on representative service over 6 months based on the Ameba recommendation engine that was developed by the authors for a service providing wellness content. As a comparison group, we selected the following hybrid CF methods: Single-Scaled Hybrid Filtering (SSF) [13], Hybrid Recommendation System Based on Usage frequency (HFUF) [21], and Reinforcement Learning Algorithm Based Hybrid Filtering (RLHF) [14]. We then implemented a simulator based on Ameba for the subject methods.

The environment for the experiments consisted of an Android application and Windows server. A server was implemented with Apache Tomcat 7.0 and Windows 7. The DBMS of the server was MySQL 5.5. The integrated development environment (IDE) of the server was Visual Studio 2010, and the language used was Visual C++. A client was implemented on Android, and Eclipse Indigo was the IDE for the client with JAVA as the language for the Android SDK.

For the experimental settings, the subject user group was composed so that the age of the subjects was evenly distributed. This group consisted of the 150 people between 20 and 60 years of age who were randomly selected from students and faculty members of Dankook University. The subject item set was composed in such a way for the characteristics of the subjects to be spread over various areas. The set consisted of the 347 wellness content items, including cultural, tourism, and leisure content that were spread over the metropolitan areas of South Korea.

5.2 Improvement of Cold-start Problem

In order to assess the improvement in the cold-start problem, we measured the Means Absolute Error (MAE) as the number of users increased. The MAE is widely used to measure the performance of recommender systems [20, 21]. It is an average of the absolute deviations between a predicted ranking and an actual ranking for the recommended items. This measure is formulated as

$$MAE = \sum_{i=1}^{N} |r_i - p_i| / N, \tag{12}$$

where N is the number of items, and p_i and r_i represent the predicted ranking and the real ranking of i-th item. To obtain the experimental data, we make the subject user group, excluding the ordinary users, insert rating scores for all items that were recommended.

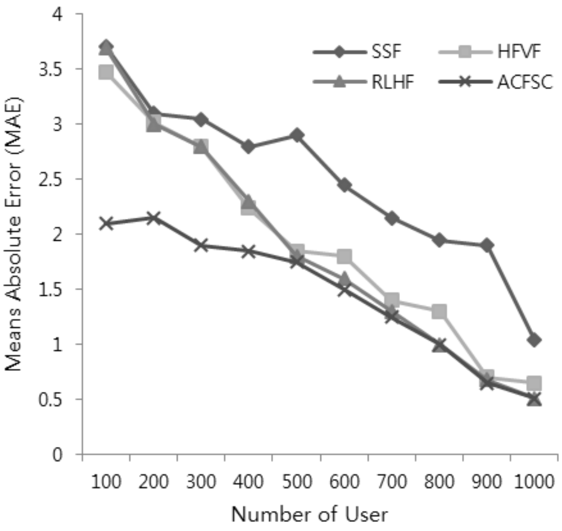


Figure 2
MAE of each Method according to users' number

Fig. 2 shows the MAE for each method with respect to the number of users. In addition, Table 2 represents the average, standard deviation, and range of the MAE for the selected methods.

As shown in Fig. 2, ACFSC exhibits its greatest performance during the initial time and also shows a comparatively steady performance after that. However, the gap between ACFSC and the other methods gradually decreases as the number of users increases. When the number of users is greater than 500, the methods show a performance similar to that of RLHF, which shows a comparatively higher performance than the other two methods. Also, when the number of users is higher than 1000, the proposed method exhibits a lower performance than RLHF.

As shown in Table 2, ACFSC presents a more stable performance than the other methods and also shows a slight improvement on average. ACFSC shows an improvement of 50.79% relative to RLHF over the given range. Also, the average MAE improved by 22.48%. In addition, we can make sure that the performance is stable for the proposed method in terms of the standard deviation.

Table 2
Average, Standard Deviation and Range of MAE for selected methods

	SSF	HFVF	RLHF	ACFSC
Average	2.516	1.956	1.899	1.472
Standard Deviation	0.778	0.907	1.038	0.562
Range	2.678	2.650	3.132	1.540

With respect to Q1, we can claim that the proposed method improves the cold-start problem. Also, since the problem of the first-user/item is caused by similar reasons as the cold-start problem, we can say that the proposed method probably can resolve the first-user/item problem as well. Nevertheless, we find that the proposed method is not able to improve the performance of the CF in an environment without the influence of cold-start problem. If we proceeded with the experiment for over 1000 users, the proposed method would not show an improvement in performance in this manner.

5.3 Robustness for Large Scale Service

In order to compare the robustness in a high-load environment, we measured the average response time as the number of users increased. The response time is a critical requirement for web-based services, such as a search engine and a recommender system. This method is defined as the amount of time that is required to provide a service. The gradient in the response time for the number of users can reveal how scalable a recommender system actually is. The average of the response time is then calculated based on the historical log of the server.

Fig. 3 represents the average for the response time of each method according to the number of users. Table 3 shows the average, standard deviation, and the range of the response time for the selected methods.

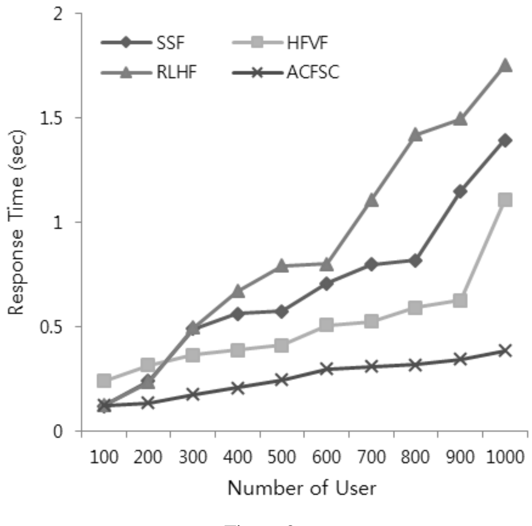


Figure 3
Average Response Time of each method according to the number of users

As shown in Fig. 3, we find that ACFSC shows a much shorter response time than the other methods by excluding the initial time. In addition, it exhibits a remarkably low and steady gradient. When the number of users is greater than 100, it has much shorter response time than HFVF, which shows a shorter response time than the other two methods. Furthermore, when the number of users is greater than 900, ACFSC shows a stable gradient while the response time of HFVF starts to exponentially increase. In particular, as mentioned above, when the number of users is greater than 500, RLHF shows similar values as MAE with the proposed method while we can see that the response time for RLHF exponentially increases. On the other hand, ACFSC shows a linear increase at that time.

In addition, as shown in Table 3, ACFSC presents a more stable response time than the other methods, and also shows a remarkable improvement on average. Our method improves by 48.25% on average, as compared with HFVF. Also, the range improved by 74.18%, and furthermore, we can see clear improvement in the standard deviation.

Table 3
Average, Standard Deviation and Range of Response Time for selected methods

	SSF	HFVF	RLHF	ACFSC
Average	0.672	0.485	0.894	0.251
Standard Deviation	0.359	0.267	0.531	0.093
Range	1.131	0.980	1.587	0.253

With respect to Q2, we can make sure that the proposed method improves the robustness of the system for use in a large-scale service. At the initial time, ACFSC shows a similar response time as the others, but this is caused by the fact that all of the other methods have extremely sparse rating matrices at that time.

5.4 Discussion

As comparing with SSF and RLHF, the proposed method (ACFSC) outperforms the existing contents-based CF methods with respect to the data sparsity problem and the scalability problem. SSF and RLHF tried to improve the data sparsity problem by building CBF model based on metadata of items (i.e., genre, running time, and so on). However, as shown in Fig. 2, the metadata cannot improve the data sparsity problem enough since it can only reflect superficial features of items. The proposed method solved this issue based on mutual learning between the feature vectors of items and users. It makes an accuracy of the feature vectors gradually better.

Also, in terms of the scalability problem, ACFSC overcomes a limitation of the existing contents-based CF methods. As shown in Fig. 3, SSF and RLHF cannot improve the scalability problem since they estimate user preference by combining two independent filtering methods which are CF and CBF. On the other hand, the proposed method improved the scalability problem based on cluster models of items and users. It reduces time complexity to compose neighborhoods of items or users. In case of original CF methods, the time complexity to build model is directly proportional to the number of items and users. However, in case of the proposed method, the time complexity to build model is directly proportional to the number of clusters of items and users.

Furthermore, as comparing with HFVF, ACFSC outperforms the existing rule-based CF methods with respect to the fore-mentioned two problems. HFVF used usage frequencies of items to improve the scalability problem. It extracted association rules from the usage frequency and applied them to reduce the number of target items to recommend (called as a coverage). As shown in Figs. 2 and 3, it can improve the scalability problem partially, but it cannot improve the data sparsity problem. Also, the proposed method reduces the coverages for both items and users. However, it improves both the data sparsity problem and scalability problem, since it reduces the coverages based on the cluster models built by preferences of users and improves them gradually.

Conclusion

The exponential increase in information (the so-called "Big Data paradigm") has caused difficulties in searching for desirable information and in addressing the increase in content. Therefore, the necessity of developing scalable recommender systems is on the rise. In this paper, we have proposed a highly scalable method (Adaptive Collaborative Filtering Based on Scalable Clustering) to guarantee stable performance and robustness of a recommender system for use in a large-scale system.

With respect to the two research questions that were previously mentioned, the proposed method performed outstanding improvements. With respect to Q1, we can claim that the proposed method improves system performance when there is a cold-start problem. Also, since the first-user/item problem is caused due to reasons similar to those of the cold-start problem, we can say that the proposed method can probably resolve the first-user/item problem, too. With respect to Q2, the proposed method was found to improve robustness for use in large-scale services.

Nevertheless, we also find that the proposed method cannot improve performance in a CF environment that does not have the influence of the cold-start problem. Our future work will therefore improve the clustering algorithm to improve system performance not only during the initial time.

Acknowledgement

This Research was supported by the Chung-Ang University Research Scholarship Grants in 2015. Also, this work was supported by the Human Resource Training Program for Regional Innovation and Creativity through the Ministry of Education and National Research Foundation of Korea (NRF-2014H1C1A1066494).

References

[1] Lee, O. J., Hong, M. S., Lee, W. J., and Lee, J. D.: "Scalable Collaborative Filtering Technique based on Scalable Clustering," Journal of Intelligence and Information Systems, Vol. 20, No. 2, 2014, pp. 73-92 (in Korean)

[2] Adomavicius, G., and Tuzhilin, A.: "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," IEEE Transactions on Knowledge and Data Engineering, Vol. 17, No. 6, 2005, pp. 734-749

[3] Pirasteh, P., Hwang, D., and Jung, J. J.: "Exploiting Matrix Factorization to Asymmetric User Similarities in Recommendation Systems," Knowledge-Based Systems, Vol. 83, 2015, pp. 51-57

[4] Pham, X. H., Nguyen, T. T., Jung, J. J., and Nguyen, N. T.: "<A, V>-Spear: A New Method for Expert Based Recommendation Systems," Cybernetics and Systems, Vol. 45, No. 2, 2014, pp. 165-179

[5] Bhosale, N. S., and Pande, S. S.: "A Survey on Recommendation System for Big Data Applications." Data Mining and Knowledge Engineering, Vol. 7, No. 1, 2015, pp. 42-44

[6] Lee, N., Jung, J. J., Selamat, A., and Hwang, D.: "Black-Box Testing of Practical Movie Recommendation Systems: A Comparative Study," Computer Science and Information Systems, Vol. 11, No. 1, 2014, pp. 241-249

[7] Ren, X., Lin, J., Yu, X., Khandelwal, U., Gu, Q., Wang, L., and Han, J.: "ClusCite: Effective Citation Recommendation by Information Network-based Clustering," Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 821-830

[8] Su, X., and Khoshgoftaar, T. M.: ″A Survey of Collaborative Filtering Techniques," Advances in Artificial Intelligence, Vol. 2009, 2009, Article 421425

[9] Cho, S.B., Hong, J. H., and Park, M. H.: "Location-based Recommendation System using Bayesian User's Preference Model in Mobile Devices," Proceeding of the 4th International Conference on Ubiquitous Intelligence and Computing (UIC 2007), Lecture Notes in Computer Science, Vol. 4611, Hong Kong, China, July 11-13, 2007, pp. 1130-1139

[10] Zhong, J., and Li, X.: "Unified Collaborative Filtering Model based on Combination of Latent Features," Expert Systems with Applications, Vol. 37, No. 8, 2010, pp. 5666-5672

[11] Luo, X., Xia, Y., and Zhu, Q.: "Incremental Collaaborative Filtering Recommender based on Regularizad Matrix Factorization," Knowledge-Based Systems, Vol. 27, 2012, pp. 271-280

[12] Cacheda, F., Carneiro, V., Fernandez, D., and Formoso, V.: "Comparison of Collaborative Filtering Algorithms: Limitations of Current Techniques and Proposals for Scalable, High-Performance Recommender Systems," Journal ACM Transactions on the Web, Vol. 5, No. 1, 2011, Articl 2

[13] Park, K. S., Choi, J. M., and Lee, D. H.: “A Single-Scaled Hybrid Filtering Method for IPTV Program Recommendation,” International Journal of Circuits, Systems and Signal Processing, No. 1, Vol. 4, 2010, pp. 161-168

[14] Shen, Y., Shin, H. C., Kim, D. G., Hong, Y. H., and Rhee, P. K.: “Reinforcement Learning Algorithm Based Hybrid Filtering Image Recommender System,” Journal of the Institute of Webcasting, Internet and Telecommunication, Vol. 12, No. 3, 2012, pp. 75-81

[15] Tewari, A. S., Kumar, A., and Barman, A. G.: "Book Recommendation System Based on Combine Features of Content Based Filtering, Collaborative Filtering and Association Rule Mining," Proceedings of the 2014 IEEE International Advance Computing Conference (IACC) 2014, pp. 500-503

[16] Bogers, T., and Van Den Bosch, A.: “Collaborative and Content-based Filtering for Item Recommendation on Social Bookmarking Websites,” Proceedings of the ACM RecSys'09 Workshop on Recommender Systems & the Social Web, 2009, pp. 9-16

[17] Campos, L. M., Fernandez-Luna, J. M., Huete, J. F., and Rueda-morales, M. A.: “Combining Content-based and Collaborative Recommendations: a Hybrid Approach based on Bayesian Networks,” International Journal of Approximate Reasoning, Vol. 51, No. 7, 2010, pp. 785-799

[18] Hu, R., and Pu, P.: “Using Personality Information in Collaborative Filtering for New Users,” Proceedings of the 2nd ACM RecSys’10 Workshop on Recommender Systems & the Social Web, 2010, pp. 17-24

[19] Kim, H. N., Alkhaldi, A., El Saddik, A., and Jo, G. S.: “Collaborative User Modeling with User-generated Tags for Social Recommender Systems,” Expert Systems with Applications, Vol. 38, No. 7, 2011, pp. 8488-8496

[20] Goldberg, K., Roeder, T., Gupta, D., and Perkins, C.: “Eigentaste: a Constant Time Collaborative Filtering Algorithm,” Information Retrieval, Vol. 4, No. 2, 2001, pp. 133-151

[21] Kim, Y., and Moon, S. B.: “A Study on Hybrid Recommendation System Based on Usage Frequency for Multimedia Contents,” Journal of the Korean society for information management, Vol. 23, No. 3, 2006, pp. 91-125 (in Korean)

[22] Li, X., and Murata, T.: “Using Multidimensional Clustering-based Collaborative Filtering Approach Improving Recommendation Diversity,” Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intell. Intell. Agent Technol., (2012) 169-174

[23] Bellogin, A., and Parapar, J.: “Using Graph Partitioning Techniques for Neighbor Selection in User-based Collaborative Filtering”. Proceedings of the 6th ACM conference on Recommender systems, ACM, (2012) 213-216

Ontology-based Multilingual Search in Recommendation Systems

Xuan Hau Pham[1], **Jason J. Jung**[2*], **Ngoc Thanh Nguyen**[3], **Pankoo Kim**[4]

[1]QuangBinh University, Vietnam
[2]Chung-Ang University, Korea
[3]Wroclaw University of Technology, Poland, ngoc-thanh.nguyen@pwr.edu.pl
[4]Chosun University, Korea, pkkim@chosun.ac.kr

Abstract: The information on the web is not only published by an original language, but also expressed in many different languages. Almost recommendation systems also lack mechanisms to support users overcoming the language problem. In these systems, it is difficult to search a specific value (e.g., movie artist, movie title in movie domain) by using native language. In this paper, we present our approach to deal with this problem. We develop an ontology-based multilingual recommendation system using integrated data from Linked Open Data to support user with in different languages on movie domain. Multilingual Movie Recommendation System (MMRS) for searching as a case of study is developed. In this system, we illustrate a more comfortable and flexible implementation.

Keywords: multilingual entities; Linked Open Data; interlink; movie; recommendation system

1 Introduction

Nowadays, user acquire information, including attributes within various media (e.g., television, radio, news paper, blog, and social networks) by a native language. The traditional recommendation systems cannot usually be applied, for efficiently searching the various media. Traditional systems have some (a few) languages to switch amoungst, however, the languages have been obtained from translation machines. This leads to connection data between certain languages and other languages that is not easy. With the developed open data system, it allows connection multiple data in different languages.

In recommendation systems, most users face language problems. They want to search some content in either their native language or some other learned

* Corresponding author

languages. The information is often published on the web by original language and expressed in different languages. After publishing, it will be translated to different languages by themselves or a community. In fact, it is always difficult to search a specific entity when users do not remember the original name (e.g., English name), they only know your country name. For example, *Cameron, J.* is a famous director, in Korea some people want to search about him, but they have a problem, they do not remember his English name. How to search in this case? In this paper, we present our approach to deal with this problem.

Multilingualism becomes an important task in natural language processing. Multilingual systems have been designed either by the system (i.e., user has to switch among languages and number of languages is limited) or the community, group users (i.e., there are a lot of languages that are published). For example, Wikipedia[1] is a huge open data source that is edited and developed by a community of volunteers in the world. Its contents are described with in many different languages, including movie.

Multilingualism is an interesting topic on the web [5, 17]. Data will be integrated from multi-resources, associated with multilingual content. Each content is expressed in different languages. However, in this paper, we only take into account multilingual searches, based on integrated data based on LOD [2, 3, 8]. Linked Open Data (LOD), provides an effective mechanism to connect data from multiple data resources by using the Resource Description Framework (RDF) and the Hypertext Transfer Protocol (HTTP). We will extract data on a movie domain. There are several LOD of movies on the web and many other movie-related data (e.g., IMDB[2], LinkedMDB[3], DBpedia[4],...).

Our proposal focuses on the integrated information from multi-resources to put them into MMRS. It will help the user overcome the language problem. In this paper, we propose and develop a onotology-based multilingual search, to support the user when searching the entities in different languages in the system. Our approach is illustrated using "movie domain". In this system, users can search a certain entity, within 145 different languages.

The outline of paper is organized as follows. In Sect. 2, we present related work about multilingual entities and ontology-based multilingual systems. In Sect. 3, we present integrated data based on LOD and ontology-based multilingual entity in recommendation systems. In Sect. 4, we develop our system and how it works. Finally, we talk about the major conclusions and future of our work herin.

[1] http://www.wikipedia.org/
[2] http://www.imdb.com/
[3] http://data.linkedmdb.org/
[4] http://dbpedia.org/

2 Related Work

Multilingual search is a challenge for social network (Facebook, Twitter, Flickr) and also open web (Wikipedia) [12, 7, 6] and recommendation systems [13, 19, 21]. Almost systems only take into account switching different languages by translating [4, 20]. The system quality depends on the translation quality. In the fact, as we know the translation machine makes a lot of mistakes. Thus, using multilingual entities based on ontology and integrated data from Linked Open Data (LOD) will improve the representation on the systems.

Ontology-based multilingual models are also proposed by [9, 15, 18]. In [9], authors proposed a new multilingual retrieval model, named CL-ESA. This model exploits the multilingual alignment of Wikipedia to represent a document as a concept vector and the similarity between two vectors can computed with the cosine similarity. In [15], the system, LabelTranlator, was proposed as an ontology to identify different languages labels. In [18], authors presented the translation-tree technique, that is based on an ontological representation for multilingual information retrieval. Each language is built as a multilingual onlology to map corresponding terms.

A multilingual search model for Flickr was proposed by Peinado et al. [7] and they called it *FlickLing*. This system allows to search monolingual and multilingual images and return a set of images with annotation in different languages. However, it can support six languages and applies a term-by-term translation for the multilingual search.

A fuzzy-based method for multilingual patent search, Fuzzy Logic Decision Support, is proposed by Segev et al. [11]. The patents are represented by a set of concepts related to a multilingual knowledge ontology. The model analyzed a several patents from Korean, US and Chinese as support for a multilingualism process.

The most important task in the multilingualism problem, is to extract name entity matching. In Wikipedia, the multilingual entity is represented as a set of InterLanguages-Links (ILL). A multilingual named entity recognition from Wikipedia is proposed by [6]. In this paper, the authors classify each article into "name entity", in nine languages and project the links onto name entities. In [10], they introduce a fuzzy-based method to extract metadata automatically and cognitive metadata generation. They also apply different document parsing algorithms to extract rich metadata from multilingual content. This framework is evaluated on three languages, English, German and French. In order to measure the similarity between multilingual sentences, Adafre et al. investigated multilingual analysis to generate similar sentences in different languages [1].

The multilingual search does not only find related results that a user needs, but also returns new information for the user. It is also a challenge for novelty mining [14]. In our approach for multilingual search within recommendation systems; the results can contain multilingual entities, where users can understand their content. For example, a user enters a query to find the director Cameron J., in Korean, the system will return a lot of information about this name in different languages (e.i., not only Korean but also other languages such as Japanese, German, Vietnamese and so on) and user can get several information.

In [8], we have applied integrated data from LOD to recommend not only a movie domain, but also books and music. The representation data on recommendation systems in different languages that are based on integrated data from LOD, will be better and more flexible, than traditionl systems.

3 Ontology-based Multilingual Recommendation Systems for Searching

3.1 Multilingual Concepts and Integrated Data on Movie Domain

The main contribution of this paper is to propose a multilingual search process to match an search entity to its corresponding concept in different languages. In our approach, we try to implement the system in a movie domain. User do not need to remember the original names of any artist or any title (e.g., English name). User can enter their language and search a certain value.

Bilingualism and multilingualism are being discussed and developed within social networks and economic systems. For movie domains, they also try to fully support users. Most systems are monolingual, such as IMDB, LinkedMDB, MusicBrainz, etc. and a few arevmultilingual systems, such as, BDpedia, Wikipedia.

IMDB is a huge moviedata repository. It describes a vast number of movies, with full information (e.g., title, genre, actor, director, music, URI, company, rating, runtime, and so on). Each entity is accessed by using URI, for example, the link *http://www.imdb.com/name/nm0000116* describes "James Cameron" director and http://www.imdb.com/title/tt0499549/ is an URI that describes about "Avatar" movie.

LinkedMDB describes movie information based on movie entities, namely interlink. Movie entities have been extracted from IMDB, DBpedia and other sources.

DBpedia is open dataset on the Internet. It is organized by categories (e.g., movie, book, music and so on). Its information is extracted from Wikipedia. Users can easy access data by using the interlinks. In DBpedia, the entities matching are based on its properties. Table 1 shows properties on movie domain. For example, in order to find the matching between two entities, on actor or director, we have to take into account *dbpedia-owl:starring* or *dbpedia-owl:director*. The name of *director* or *starring* will be represened as a list of entities in different languages.

Table 1
The properties on DBpedia and LinkedMDB for movie domain

DBpedia		**LinkedMDB**	
Property	Attribute	Property	Attribute
dbpedia-owl:work/runtime	Runtime	movie:actor	Starring
dbpedia-owl:abstract	Abstract	movie:cinematographer	Cinematographer
dbpedia-owl:cinematography	Cinematographer	dc:date	Event
dbpedia-owl:director	Director	movie:director	Director
dbpedia-owl:distributor	Distributor	movie:editor	Editor
dbpedia-owl:editing	Editing	movie:genre	Genre
dbpedia-owl:musicComposer	Composer	movie:initial_release_date	Release date
dbpedia-owl:producer	Producer	movie:language	Movie language
dbpedia-owl:writer	Writer	foaf:page	IMDB link
dbpedia-owl:starring	Starring	movie:producer	Producer
rdfs:comment	User comment	movie:runtime	Runtime
rdfs:label	Title	dc:title	Title
owl:sameAs	Multilingism	rdf:type	Type
dbpprop:language	Original language	movie:writer	Writer
dbpprop:country	Country	movie:actor_name	Actor name
dbpedia-owl:abstract	Abstract	movie:director_name	Director name
dbpedia-owl:alias	Alias	movie:film_genre_name	Genre name
dbpedia-owl:birthDate	Birthday	movie:editor_name	Editor name
dbpedia-owl:birthName	Name		
dbpedia-owl:birthPlace	Birthplace		
dbpedia-owl:birthYear	Birthyear		
dbpedia-owl:education	Education		
rdfs:comment	Comment		
foaf:givenName	Given name		
dbpprop:occupation	Occupation		

In our scenario, each entity (label, concept) will be detected by URI on the resource systems. It will identify the languages, contents, description and interlinks. In multilingual recommendation systems, each value of an item, as a concept, takes the information from resource data and has a list of corresponding different languages for the concept. This list is automatically detected and obtained from multilingual systems.

The following algorithm is used for integrating data:

```
Input: a list of movies, I
Output: a set of multilingual movie concepts
Algorithm:
Foreachi ∈ I
        V = a set of movie concepts from IMDB
        Foreachv ∈ V
                    Mapping into LinkedMDB
                    Extracting data from DBpedia
    Return E = a set of multilingual movie concepts
```

Figure 1 shows the relationships between Korean information and English information of "The Spy" movie based on DBpedia properties (e.g., *dbpedia-owl:director*, *owl:sameAs*, *dbpedia-ko* and so on). We can see that each value of movie will be expressed by its corresponding properties. Since DBpedia data is extracted from Wikipedia, some contents are inconsistent. Thus, we use movie data from IMDB as standard information for integrating data.

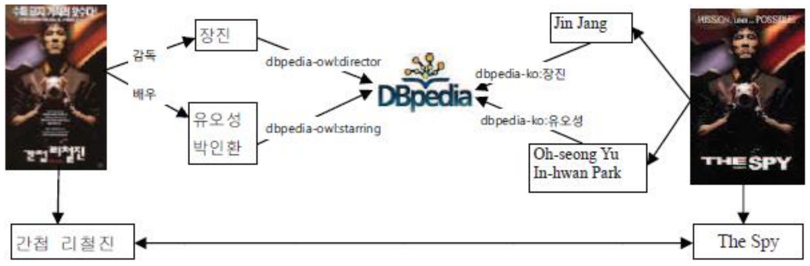


Figure 1
Multilingual entities matching

3.2 Multilingual Recommendation System

The aims of the recommendation system is not only suggest a set of new items based on user preference, but also explore the relationship among users and items. Therefore, multilingual recommendation systems will produce more interesting items in different languages and support users in overcoming the language problem. In order to understand our proposal we present some definitions for the multilingual concept and our model as follows:

Definition 1 (Multilingual Recommendation System Framework) *A multilingual recommendation system is defined as a 6-tuple:*

$$S = \langle U, I, A, V, L, R \rangle \tag{1}$$

where U is a set of users, I is a set of items, A is a set of attributes, V is a set of concepts, L is a set of languages and $R \subseteq A \times V \times L$ is a set of links. Rcan be represented as a set of interlink-languages (ILL).

Each $i \in I$, we can extract:

$$R_i = \{(a, v, l) | a \in A, v \in V : l \in L\} \tag{2}$$

For example, considering the movie *Titanic*, we obtain:

- (Title, “Titanic”, “en”, “http://dbpedia.org/page/Titanic_(1997_film)”),
- (Direktor, “Cameron, J.”, “de”, “http://dbpedia.org/page/James_Cameron”),
- (Acteur, “LeonardoDiCaprio”, “fr”,

  “http://dbpedia.org/page/Leonardo_DiCapri”),
- (Genre, “Adventure”, “en”, “http://data.linkedmdb.org/page/film_genre/31”).

LOD provides a mechanism to connect data from multiple resources. The connections can be established by links In LOD, each concept is described by its content and interlink. We will use interlinks to find out the entities matching.

Definition 2 (Concept Matching) *Let* $v_1, v_2 \in V$*and* $i \in I$*, the matching between two concepts is computed as follows:*

$$M(v_1, v_2) = \begin{cases} -1 & if v_1, v_2 \in R_i \\ 0 & otherwise \end{cases} \tag{3}$$

Searching on multilingual systems takes into account the entities matching. Each concept will have a set of different multilingual concepts which have the same description. For example, we consider the “Titanic” title, in other languages, in Table 2.

Table 2

The “Titanic” title in different languages

Title	Language	Interlink
Titanic	French	http://fr.dbpedia.org/resource/Titanic_(film,_1997)
Titanic	German	http://de.dbpedia.org/resource/Titanic_(1997)
타이타닉	Korean	http://ko.dbpedia.org/resource/타이타닉_(1997년_영화)
タイタニック	Japanese	http://ja.dbpedia.org/resource/タイタニック_(1997年の映画)
Titanic	Italian	http://it.dbpedia.org/resource/Titanic_(film_1997)

Each concept will have a set of corresponding entities. A set of concepts will have a set of corresponding interlinks. It will help improve the matching.

Definition 3 (Multilingual Search)

Given V is a search entity, the result of this search, is represented as follows:

$$M(v) = \{(v', l, r) | v' \in V, l \in L, r \in R: Lex(v) \subset V\} \quad (4)$$

where *Lex(v)* is a function to return a set of result concepts.

In recommendation systems, the most important task is to build the user profile (user preferences). Each user will record all of their interactions and all the information in a session. The systems will discover these data to understand what user needs and predict recommendation for next time.

Definition 4 (User Preference) Give certain $u \in U$, *the ontological user profile in the recommendation system is expressed as follows:*

$$f(u) = \{(v, l) | \forall v \in V, \exists l \in L: (v, l) \subset R\} \quad (5)$$

Definition 5 (User Similarity) *Given two users* $u_1, u_2 \in U$ *.T the similarity between two users based on user preference is defined as follows:*

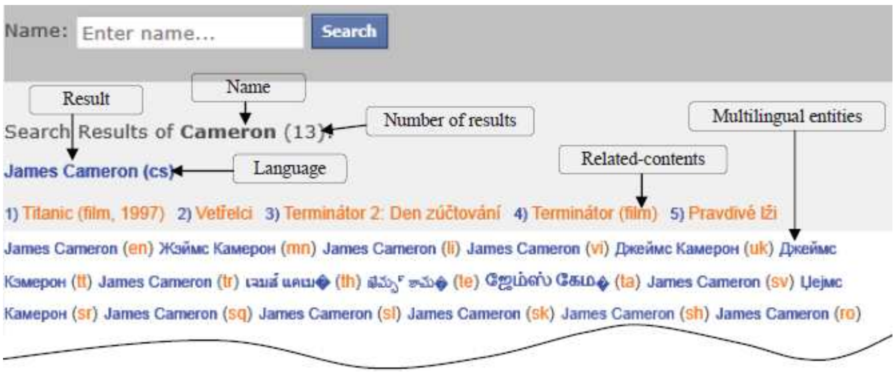
$$Sim(u_1, u_2) = \frac{sim(v_{u_1}, v_{u_2})_{f(u_1) \cap f(u_2)}}{sim(v_{u_1}, v_{u_2})_{f(u_1) \cup f(u_2)}} \quad (6)$$

4 Multilingual Search Movie Recommendation System: a case of study

In this paper, we propose multilingual searches in a movie recommendation system, as a case study. Figure 2 shows the main the interface of our system. It is easy to input a name (e.g., movie, artist) and search. The result will be represented related-movie blocks.

- Number of results
- List of results
- <name, language>
- Related-contents

  (e.g. a list of movies for an artist, list of artists for a movie)
- Multilingual entities, languages and interlinks



Search Results of **Cameron** (13)

James Cameron (cs)

1) Titanic (film, 1997) 2) Vetřelci 3) Terminátor 2: Den zúčtování 4) Terminátor (film) 5) Pravdivé lži

James Cameron (en) Жэймс Камерон (mn) James Cameron (li) James Cameron (vi) Джеймс Камерон (uk) Джеймс Кэмерон (tt) James Cameron (tr) เจมส์ แคเม� (th) జేమ్స్ కామె� (te) ஜேம்ஸ் கேம� (ta) James Cameron (sv) Џејмс Камерон (sr) James Cameron (sq) James Cameron (sl) James Cameron (sk) James Cameron (sh) James Cameron (ro)

Figure 2
Main interface of multilingual movie search

In order to implement our system, we have extracted the dataset as Section 3.1. Table 3 expresses the multilingual entities in the system.

Table 3
The multilingual entities statistic in our system

Languages	#Artist	#Title
English	10845	1888
German	6614	1165
French	6877	1170
Russian	4546	998
Italian	5933	1100
Korean	3703	162
Japanese	4411	815
Vietnamese	795	61
Chinese	2237	281

Figure 3 shows the results of a search for the movie, "Titanic" in English, Korean and Russian languages.

We can extract the relationships among different languages for one movie. For example, we can extract 15 languages for the *Titanic* movie and 59 languages, for *Morgan Freeman,* the actor. When a user searches a certain name of movie, the system will show the movie information (e.g., director(s), actors, actresses) and a list of the same names in different languages (English (en), French (fr), German (de), Chinese (zh), Korean (ko), Japanese (ja), Vietnamese (vi), and so on). Also, when a user searches a certain name of an artist, it will show a list of movies that they were in and a list of the same their names. Figure 4 shows the connection between *Titanic* in English and *Cameron, J.* director, in Korean on multilingual movie search.

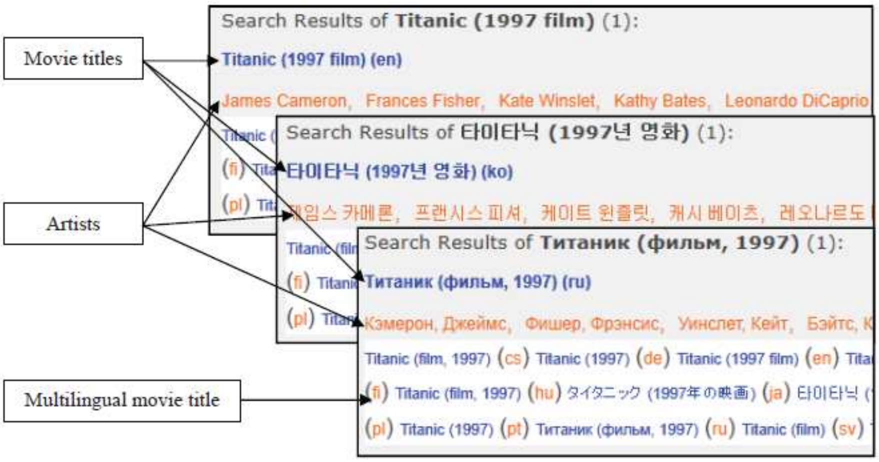

Movie titles
Artists
Multilingual movie title
Search Results of Titanic (1997 film) (1):
Titanic (1997 film) (en)
James Cameron, Frances Fisher, Kate Winslet, Kathy Bates, Leonardo DiCaprio
Search Results of 타이타닉 (1997년 영화) (1):
타이타닉 (1997년 영화) (ko)
제임스 카메론, 프랜시스 피셔, 케이트 윈즐릿, 캐시 베이츠, 레오나르도
Search Results of Титаник (фильм, 1997) (1):
Титаник (фильм, 1997) (ru)
Кэмерон, Джеймс, Фишер, Фрэнсис, Уинслет, Кейт, Бэйтс
Titanic (film, 1997) (cs) Titanic (1997) (de) Titanic (1997 film) (en)
(fi) Titanic (film, 1997) (hu) タイタニック (1997年の映画) (ja) 타이타닉
(pl) Titanic (1997) (pt) Титаник (фильм, 1997) (ru) Titanic (film) (sv)

Figure 3
Representing "Titanic" movie on multilingual movie

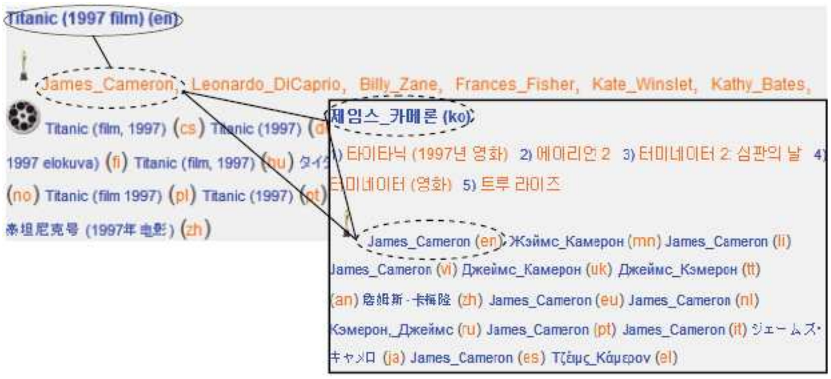
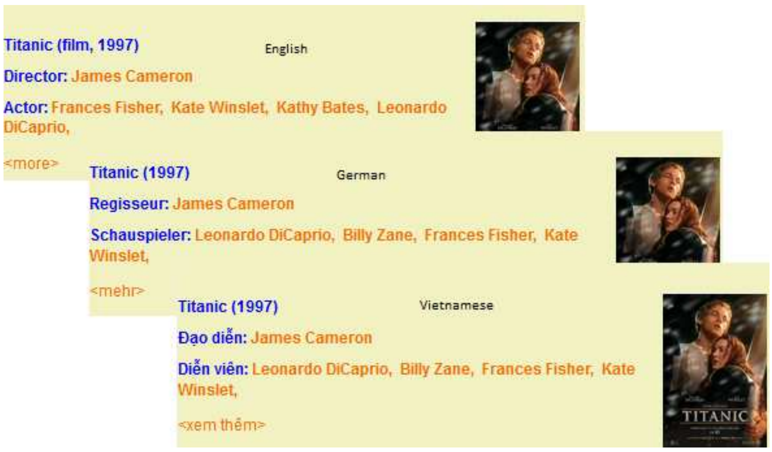

Titanic (1997 film) (en)
James_Cameron, Leonardo_DiCaprio, Billy_Zane, Frances_Fisher, Kate_Winslet, Kathy_Bates,
Titanic (film, 1997) (cs) Titanic (1997)
1997 elokuva) (fi) Titanic (film, 1997) (hu)
(no) Titanic (film 1997) (pl) Titanic (1997) (pt)
泰坦尼克号 (1997年电影) (zh)
제임스_카메론 (ko)
타이타닉 (1997년 영화) 2) 에이리언 2 3) 터미네이터 2: 심판의 날 4)
터미네이터 (영화) 5) 트루 라이즈
James_Cameron (en) Жэймс_Камерон (mn) James_Cameron (li)
James_Cameron (vi) Джеймс_Камерон (uk) Джеймс_Кэмерон (tt)
(an) 詹姆斯·卡梅隆 (zh) James_Cameron (eu) James_Cameron (nl)
Кэмерон,_Джеймс (ru) James_Cameron (pt) James_Cameron (it) ジェームズ・
キャメロン (ja) James_Cameron (es) Τζέιμς_Κάμερον (el)

Figure 4
The connections between multilingual entities on multilingual search

When a user searches a movie title, the system will return a set of movies including artists and multilingual titles. It will return a set of artist names in multiple languages and a set of movies that this artist was involved in when the user searches movie artist.

In order to develop multilingual search systems, that we have implemented on a movie domain. Movie information is extracted from IMDB and DBpedia. The quality and number of languages for multilingualism depend on data in those sources. It means that if the entities are well-known, then the multilingual entities will be more accurate.

In addition, the system will build user profiles for the various users. The recommendation processing is based on an ontological user preference, to predict which is a better search item for each user. Figure 5 shows the recommendation interface in 3 languages (English, German, Vietnamese).



Titanic (film, 1997)

Director: James Cameron

Actor: Frances Fisher, Kate Winslet, Kathy Bates, Leonardo DiCaprio,

<more>

Titanic (1997)

Regisseur: James Cameron

Schauspieler: Leonardo DiCaprio, Billy Zane, Frances Fisher, Kate Winslet,

<mehr>

Titanic (1997)

Đạo diễn: James Cameron

Diễn viên: Leonardo DiCaprio, Billy Zane, Frances Fisher, Kate Winslet,

<xem thêm>

Figure 5

The recommendation interface

Concluding remarks

Multilingual searches in a movie recommendation system will bring a more flexible interaction for users. Users can overcome language problems. Each item not only is described on bilingual data, but also expressed in various languages. In this paper, we presented our approach to discover the relationships among multilingual concepts for searching on a movie domain and ontological user preferences are also considered in recommendation processing. We also developed a demo system for our proposal. However, the integrated data, based on LOD, is extracted offline and several languages are not available in the data resources. Thus, returned results are not a full expression.

As for future work, we will increase the number of entities and the number of movies. We would also like to show a comparision with other current approaches.

Acknowledgement

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2013.12. Also, this work was supported by the Human Resource Training Program for Regional Innovation and Creativity through the Ministry of Education and National Research Foundation of Korea (NRF-2014H1C1A1066494). This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (NRF-2014R1A2A2A05007154).

References

[1] Adafre S. F., and De Rijke M.: Finding Similar Sentences across Multiple Languages in Wikipedia, Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, 2006, pp. 62-69

[2] Bizer C., Heath T. and Berners-Lee T.: Linked Data - the Story so Far, International Journal on Semantic Web and Information Systems, Vol. 5 (3), 2009, pp. 1-22

[3] Hassanzadeh O. and Consens MP.: Linked Movie Data Base, Proceedings of the WWW2009 Workshop on Linked Data on the Web (LDOW 2009), Spain, 2009

[4] Hillier M.: The Role of Cultural Context in Multilingual Website Usability, Electronic Commerce Research and Applications, Vol. 2 (1), 2003, pp. 2-14

[5] Jung J. J.: Cross-Lingual Query Expansion in Multilingual Folksonomies: A Case Study on Flickr, Knowledge-based Systems, Vol. 42, 2013, pp. 60-67

[6] Nothman J., Ringland N., Radford W., Murphy T. and Curran J. R.: Learning Multilingual Named Entity Recognition from Wikipedia, Artificial Intelligence Vol. 194, 2013, pp. 151-175

[7] Peinado V., Artiles J., Gonzalo J., Barker E. and López-Ostenero F.: Flickling: a Multilingual Search Interface for Flickr, Proceedings of CLEF 2008 Workshop Notes, Aarhus, Denmark, 2008

[8] Bello-Orgaz G., Jung J. J., Camacho D.: Social Big Data: Recent Achievements and New Challenges, Information Fusion, Vol. 28, 2016, pp. 45-59

[9] Potthast M., Stein B. and Anderka M.: A wikipedia-based Multilingual Retrieval Model, Proceedings of the IR Research, 30th European Conference on Advances in Information Retrieval, ECIR'08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 522-530

[10] Sah M., and Wade V.: Automatic Metadata Mining from Multilingual Enterprise Content, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 11, 2012, pp. 41-62

[11] Nguyen D. T., Jung J. E.: Real-Time Event Detection on Social Data Stream, Mobile Networks and Applications, Vol. 20 (4), 2015, pp. 475-486

[12] Yarowsky D., Ngai G., and Wicentowski R.: Inducing Multilingual Text Analysis Tools via Robust Projection across Aligned Corpora, Proceedings of the First International Conference on Human Language Technology Research, HLT '01, Association for Computational Linguistics, Stroudsburg, PA, USA, 2001, pp. 1-8

[13] Lops P., Musto C., Narducci F., De Gemmis M., Basile P., and Semeraro G.: Mars: a Multilanguage Recommender System, Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems, ACM, pp. 24-31

[14] Zhang Y., Tsai F. S., and Kwee A. T.: Multilingual Sentence Categorization and Novelty Mining, Information Processing and Management, Vol. 47 (5), 2011, pp. 667-675

[15] Espinoza M., Gómez-PérezA., and Mena E.: Enriching an Ontology with Multilingual Information, Springer Berlin Heidelberg, 2008, pp. 333-347

[16] Embley D. W., Liddle S. W., Lonsdale D. W., and Tijerino Y.: Multilingual Ontologies for Cross-Language Information Extraction and Semantic Search. Conceptual Modeling–ER 2011, Springer Berlin Heidelberg, 2011, pp. 147-160

[17] Gracia J., Montiel-Ponsoda E., Cimiano P., Gómez-Pérez A., Buitelaar P., and McCrae J.: Challenges for the Multilingual Web of Data, Web Semantics: Science, Services and Agents on the World Wide Web, 2012, pp. 63-71

[18] Guyot J., Radhouani S., and Falquet G.: Ontology-based Multilingual Information Retrieval, CLEF Workhop, Working Notes Multilingual Track, 2005, pp. 21-25

[19] Luberg A., Järv P., Schoefegger K. and Tammet T.: Context-Aware and Multilingual Information Extraction for a Tourist Recommender System, Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies, ACM, 2011

[20] Zahed F., Van Pelt W. and Song J.: A Conceptual Framework for International Web Design, Professional Communication, IEEE Transactions, Vol. 44 (2), 2001, pp. 83-103

[21] Pham X. H., and Jung J. J.: Recommendation System Based on Multilingual Entity Matching on Linked Open Data, Journal of Intelligent and Fuzzy Systems, Vol. 27(2), 2014, pp. 589-599

Scalable co-Clustering using a Crossing Minimization – Application to Production Flow Analysis

Csaba Pigler, Ágnes Fogarassy-Vathy[*]

Department of Computer Science and Systems Technology,
University of Pannonia, Egyetem u. 10, H-8200 Veszprém, Hungary
piglercs@dcs.uni-pannon.hu, vathy@dcs.uni-pannon.hu

János Abonyi

Department of Process Engineering, University of Pannonia
Egyetem u. 10, H-8200 Veszprém, Hungary
Institute of Advanced Studies Kőszeg, Chernel u. 14, H-9730 Kőszeg, Hungary
janos@abonyilab.com

Abstract: Production flow analysis includes various families of components and groups of machines. Machine-part cell formation means the optimal design of manufacturing cells consisting of similar machines producing similar products from a similar set of components. Most of the algorithms reorders of the machine-part incidence matrix. We generalize this classical concept to handle more than two elements of the production process (e.g. machine - part - product - resource - operator). The application of this extended concept requires an efficient optimization algorithm for the simultaneous grouping these elements. For this purpose, we propose a novel co-clustering technique based on crossing minimization of layered bipartite graphs. The present method has been implemented as a MATLAB toolbox. The efficiency of the proposed approach and developed tools is demonstrated by realistic case studies. The log-linear scalability of the algorithm is proven theoretically and experimentally.

Keywords: cell formation; co-clustering; co-crossing minimization

[*] Corresponding author

1 Introduction

Industry 4.0 is focused on smart production in smart factories where there is direct communication between humans, machine, and resources. Smart products know their manufacturing process and future application [16, 25]. With this knowledge, they actively endorse the production process and the documentation ("when was I made, which parameters am I to be given, where I am supposed to be delivered."). Thanks to these developments that can be represented by 5C keywords (Connection, Cloud, Content, Community, and Customization) [17], complex production systems generate and store huge datasets, which has a high potential for productivity improvement. All these factors raise the question: "How can we increase productivity of manufacturing systems by the analysis of the huge amount of available data?"

There exists several potentials for Big Data, in manufacturing, such as, production management [24], supply chain planning [12], maintenance, and sales [8, 9]. Among these, production management is the most complex problem as it includes scheduling, optimization, (human) resource management, warehouse logistic, and manufacturing cell formation. Once this enormous amount of information is available from the components, the systems will be processed at the same time and the optimization of the manufacturing systems can significantly improve.

Cell formation (CF) is a widely studied topic in production systems optimization circles. The aim of cell formation is to create manufacturing cells from a given number of machines and products, by partitioning similar machines producing similar products. Since the formation of optimal manufacturing cells contributes significantly to the increased production, several different approaches have been proposed for the solution of a CF problems [5, 11, 15, 18, 23, 34]. Since, it can be difficult to find an optimal solution, in an acceptable amount of time, especially for large scale problems, usually heuristic approaches are used [19-22, 32, 33].

Furthermore, all these methods work only with relationships of machines and products (or with relationships of machines and parts). As these relationships can be represented by a two-layered bipartite graph or by an incidence matrix, the classical cell formation process can be considered as a biclustering task [1, 7].

Although, in complex manufacturing processes machines should be characterized by numerous properties, like the type of products, resources, and required skills from operators. To handle these elements of the production line, the traditional cell formation problem should be extended, and instead of the biclustering task, a co-clustering problem should be solved.

While biclustering algorithms can solve classical manufacturing CF process [1], we try to handle the extended multidimensional problem as a co-clustering task [10] based on crossing minimization of multipartite graphs. Since NP-hard problem, we utilize the widely applied heuristic barycentric method.

In a traditional cell formation problem, crossing minimization reorders the machines into cells, based on their similar part usage. As we mentioned before, cell formation problems can be more complex and they can require more properties to describe the whole production process. While dealing with these complex datasets, we developed a new crossing minimization method for multi-partite graphs. The proposed method sequentially reorders the elements of the node sets, thereby it relocates the elements of the connectivity matrix into a block-diagonal way. As result, the cell formation problem is handled in his original complexity.

Crossing minimization heuristics have been a subject of many years [4, 6, 13, 14, 28, 30, 31]. The complexity of these heuristics are linear or log-linear [26, 27]. This clearly indicates that the proposed problem formulation can lead to efficient solution for the multivariate cell formation problem.

Graphical representation of the manufacturing cell formation may also support optimization of production systems. Approaches like hierarchical clustering or Visual Assessment of Cluster Tendency (VAT) [2] are able to visualize the similarities of the elements, but by default they take only one or two variables into account. Nevertheless, complexities of these methods are not appropriate for Big Data [3], as the time complexity of VAT is $\mathcal{O}(N^2)$, and the complexity of agglomerative clustering is $\mathcal{O}(N^3)$, where N is the number of objects to be clustered.

In this article, we aim to present a novel production process optimization method which is able to accomodate more aspects (machines, suppliers, human resources, bill of materials, etc.) simultaneously and still be able to visualize the cell formation problem and it's solution accordingly for human interpretation. The optimization method is based on a newly developed co-crossing minimization method that solves the co-crossing minimization problem between $\mathcal{O}(N)$ and $\mathcal{O}(N \log N)$ time, and therefore, able to process the Big Data rapidly and concurrently, the productivity of manufacturing systems can be significantly increased.

In the following we formulate the extended cell formation problem, then we present the novel co-crossing minimization algorithm that is able to handle this complex problem. The algorithm provides easy implementation and low computing complexity. Finally, we present the capability of our new method on different cell formation examples. Firstly, we compare our approach with the popular hybrid-heuristic cell formation algorithm [33] and show the applicability of the proposed performance measures. This will be followed by the numerical analysis of the scalability. Finally, an illustrative real-life example will be given.

2 Multidimensional Representation of the Cell Formation Problem

2.1 Classical Bipartite Graph-based Representation

Traditional cell formation problems are represented by bipartite graphs *(V,E)*, where *V* represents the set of vertices and *E* the set of edges. *V* is partitioned into two adjacent subsets. $V_0 = \{v_{0,1}, v_{0,2}, \dots, v_{0,N_0}\}$ represents the set of machines, and $V_1 = \{v_{1,1}, v_{1,2}, \dots, v_{1,N_1}\}$ the set of parts (see Figure 1

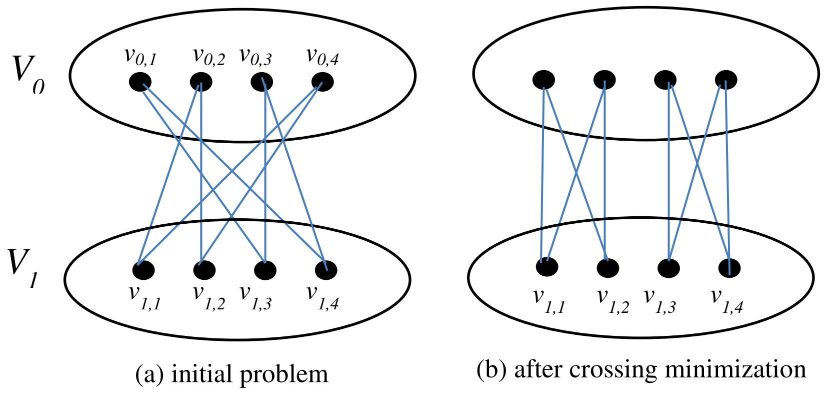


Figure 1
The classical cell formation problem is based on the crossing minimization of a bipartite graph, where V_0 represents the sets of machines and V_1 the set of the parts

The cell formation is based on the rearrangement of the order of the vertices. $\boldsymbol{o}_i$, $i = 0,1$, where $\boldsymbol{o}_i$ represents the sequence of all vertices of V_i . E.g. after the minimization of the crossings in the illustrative problem shown in Figure 1a, the sequence of the vertices becomes $\boldsymbol{o}_0 = [2,4,1,3]$.

The bipartite graph of the machine - part connections can also be represented by an interconnection matrix, $\boldsymbol{A}[\boldsymbol{o}_0,\boldsymbol{o}_1]$. The a_{ij} element of $\boldsymbol{A}[\boldsymbol{o}_0,\boldsymbol{o}_1]$ is 1 when the $\boldsymbol{o}_{0,i}$ -th machine uses the $\boldsymbol{o}_{1,j}$ -th part as input and otherwise 0. Please note, that the k-th element of these $\boldsymbol{o}_i$ vectors is the index which row (*i=0*) or column (*i=1*) is placed at the k-th place in the ordering. Later we are going to also use vector, $\boldsymbol{p}_i$ to show the position of the vertices, $v_{i,j}$. In our illustrative example as the first vertex, $v_{0,1}$, is placed in the third place $p_{0,1}=3$, $\boldsymbol{p}_0 = [3,1,4,2]$.

From this viewpoint, the crossing minimization can be considered as reordering the rows and columns of the interconnection matrix to explore the hidden block-oriented structure of the matrix.

E.g. the initial problem is represented as: $$\boldsymbol{A}[\boldsymbol{o}_0, \boldsymbol{o}_1] = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} \quad (1)$$

After crossing minimisation: $$\boldsymbol{A}[\boldsymbol{o}_0, \boldsymbol{o}_1] = \begin{bmatrix} \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 0 & \mathbf{1} & \mathbf{1} \\ 0 & 0 & \mathbf{1} & \mathbf{1} \end{bmatrix} \quad (2)$$

As this simple illustrative example shows, minimization of the crossings provides not only a better visualization of the bipartite graph, but it also reorders the rows and columns of the incidence matrix in a block-diagonal way. With this order, similar nodes are placed next to each other, so they can form blocks that can be used to define manufacturing cells.

2.2 The Proposed Multidimensional Representation

In complex manufacturing processes, machines are characterized by numerous properties, like the type of products, resources, and the required skills of the operators. To handle all elements of the production line we extended the conventional cell formation task into a multidimensional problem. According to this goal, our key idea is to represent the cell formation problems by multi-layered graphs (see Figure 2).

The proposed n-dimensional representation is based on n sets of vertices

$$V = V_0 \cup V_1 \cup V_2 \cup \dots V_n, \quad V_i \cap V_j = \emptyset, \forall i \neq j \quad (3)$$

where each set represents possible values/categories of an attribute/feature of the machine. E.g. V_0 represents the set of machines, V_1 the parts, V_2 the products, V_3 the resources, V_4 skills of the operators.

Relationships between the machines and the i-th attribute of the production line can also be represented by a sparse matrix, $\boldsymbol{A}^{(i)}[\boldsymbol{o_0}, \boldsymbol{o_i}]$, $i = 1 \dots n$, where the dimensions of these matrices are $N_0 \times N_i$.

The second fundamental idea is that the simultaneous re-ordering of the rows and columns of these matrices clusters the machines and supports the formulation and optimization of the manufacturing cells, as seen in Figure 2

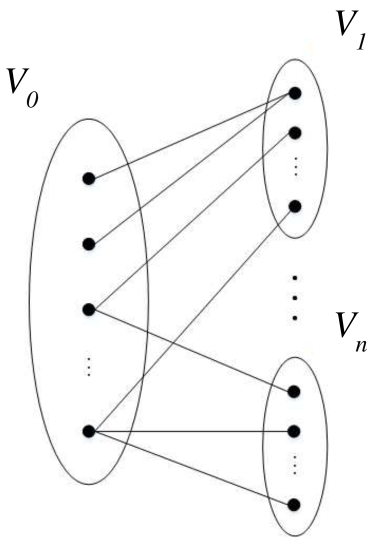


Figure 2

Representation of the multidimensional cell formation problem

The visualized benchmark cell formation problem has $N_0=20$ machines processing $N_1=34$ different parts, utilizing $N_2=31$ different resources, while working $N_3=37$ operators. The structure of this problem can be seen in the first row in Figure 3. The second row of this figure shows the rearranged data after the proposed co-crossing minimization algorithm which will be presented in the following section.

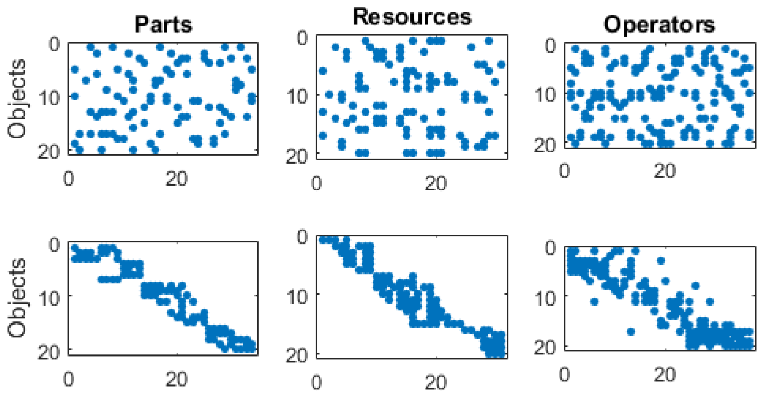


Figure 3

The benchmark cell-formation problem and the result of co-crossing minimization

3 Co-Crossing Minimization Algorithm

3.1 Barycentric Ordering of Crossing Minimization

The classical heuristic barycentric method iteratively reorders the $\boldsymbol{o}_0$ row and the $\boldsymbol{o}_1$ column orders of $\boldsymbol{A}[\boldsymbol{o}_0,\boldsymbol{o}_1]$ to reduce the number of the crossings. The reordering is based on the row and column barycenters of the $\boldsymbol{A}$ interconnection matrix. Barycenter heuristic assigns a new rank to each node based on the mean of ranks of its neighbor nodes.

The third key contribution of our paper is that we formulated the algorithm with the help of matrix operations to support compact, sparse matrix representation based implementation in of data analysis tools, like MATLAB.

The column barycenters are calculated as the mean of the places of the neighboring vertices

$$\boldsymbol{b}_i^C = \boldsymbol{p}_i^T \boldsymbol{A}^{(i)}./\boldsymbol{s}_i^C \tag{4}$$

where $\boldsymbol{p}_i$ represents the vector of the places of the vertices, $\boldsymbol{s}_i^C$ represents a row vector of the sum of the connections calculated as the sum of the columns of $\boldsymbol{A}^{(i)}$

$$\boldsymbol{s}_i^C = \boldsymbol{u}_0^R \boldsymbol{A}^{(i)} \tag{5}$$

with the help of the $\boldsymbol{u}_0^R$, which is an N_0 sized unitary row vector.

With this formulation the j-th element of the $\boldsymbol{b}_i^C$ vector, $b_{i,j}^C$, can be interpreted as the average of the places of the machines that are connected to the j-th element of the i-th feature, $v_{i,j}$.

The row barycenters of $\boldsymbol{A}^{(i)}[\boldsymbol{o}_0,\boldsymbol{o}_i]$ are calculated similarly,

$$\boldsymbol{b}_i^R = \boldsymbol{A}^{(i)}\boldsymbol{p}_0 ./\boldsymbol{s}_i^R \tag{6}$$

where $\boldsymbol{s}_i^R$ represents the sum of rows of $\boldsymbol{A}^{(i)}$, calculated as

$$\boldsymbol{s}_i^C = \boldsymbol{A}^{(i)}\boldsymbol{u}_i^C \tag{7}$$

where $\boldsymbol{u}_i^C$ is an N_i sized unitary column vector.

The standard crossing minimization algorithm iteratively reorders $\boldsymbol{o}_0$ based on shorting $\boldsymbol{b}_i^R$ and generates $\boldsymbol{A}[\boldsymbol{o}'_0,\boldsymbol{o}_1]$, than reorders $\boldsymbol{o}_1$ to generate $\boldsymbol{o}'_1$ based on the shorting the $\boldsymbol{b}_1^C$ barycenters of the columns of $\boldsymbol{A}[\boldsymbol{o}'_0,\boldsymbol{o}_1]$. The number crossings is minimised by repeating these orderings till convergence.

3.2 The Co-Crossing Minimization Algorithm

The key idea of the algorithm is that we simultaneously arrange the rows and the columns of the $\boldsymbol{A}^{(i)}[\boldsymbol{o}_0,\boldsymbol{o}_i]$ matrices. Since the row-orders of these matrices are identical, the row barycenters are calculated as the weighted sum of the row barycenters of the individual matrices:

$$\boldsymbol{b}^R = \sum_{i=1}^{n} \boldsymbol{b}_i^R w_i = \sum_{i=1}^{n} \boldsymbol{A}^{(i)} \boldsymbol{p}_0 \ ./\boldsymbol{s}_i^R w_i \quad (8)$$

The w_i weight can represent the importance of the features to their contribution to the cell-formation problem. When all features have equal importance, w_i should set as $w_i = 1/N_i$ to ensure that features with different number of categorical values have the same weight.

The steps of our new method can be seen in Algorithm 1.

```
initialize the row orders:
$\boldsymbol{o}_i, i = 0 \dots n$ as $1 : N_i$
calculate the sum of the columns and rows:
$\boldsymbol{s}_i^C, \boldsymbol{s}_i^R \ i = 1 \dots n$
while converge do
    calculate $\boldsymbol{b}_i^R, i = 1, \dots, n$, and $\boldsymbol{b}^R$
    calculate the new $\boldsymbol{o}_0$ row order by shorting $\boldsymbol{b}^R$
    calculate the new $\boldsymbol{p}_0$ places of the vertices of $V_0$
    for $i = 1{:}n$ do
        calculate $\boldsymbol{b}_i^C, i = 1, \dots, n$
        calculate the new $\boldsymbol{o}_i$ row order by shorting $\boldsymbol{b}_i^C$
        calculate the new $\boldsymbol{p}_i$ places of the vertices of $V_i$
    end for
end while
```

Algorithm 1

The proposed co-crossing minimization algorithm

Please note that the $\boldsymbol{p}_i$ vectors are generated by sorting $\boldsymbol{o}_i$ the order vectors. The use of these vectors is important since the algorithm does not modify the original $\boldsymbol{A}^{(i)}$ space matrices, which ensures fast and memory effective implementation.

The algorithm stops when the ordering is converged or the maximum iteration number is reached.

3.3 Complexity Analysis

The complexity of the classical barycenter technique is $\mathcal{O}(|E|+|V| \ \log V)$ [1,30]. It should be noted that since we decomposed the problem into n almost independent subproblems as $V = V_0 \cup V_1 \cup \dots V_n$, $V_i \cap V_j = \emptyset, \forall i \neq j$ and $E = E_1 \cup \dots E_n$, $E_i \cap E_j = \emptyset, \forall i \neq j$, the complexity of the algorithm is smaller than if we would handle the standard classical problem with the same size. By decreasing the sparsity of the problem (increasing the number of edges) the complexity linearly increases, while the increase of the number of vertices has log-linear effect as the critical step in the algorithm is, that it requires 2(*n+1)* sorts of the nodes in each iteration.

When this advantageous $\mathcal{O}$ *(N log N)* scaling of the quicksort algorithm is not enough, to achieve speed proportional to the size of the data $\mathcal{O}$*(N),* binsort can be applied. However, this requires auxiliary storage and memory for the bins. Because quicksort manipulates the data in place, it can sort larger arrays, albeit somewhat a bit slower. Another sorting option, is the application of the $\mathcal{O}$ *(N)* counting sort algorithm. It should be noted, that this and another advanced integer sorting algorithm requires the calculation of the median instead of the mean or rounding the ranks into integers. To scale the algorithm, it is possible to execute the loop iterations in parallel.

The algorithm solves the crossing minimization problem rapidly. Similarly to multi-layered application of crossing minimizatuion usually it stops after 5-10 iterations [1, 30]. It calculates the node ranks in each set of nodes linear algebraic way with matrix and vector multiplication. It should be noted, that matrices describing the cell formation problem are sparse, so they can be stored and handled efficiently.

Concluding, the proposed algorithm can be implemented in Big Data environments, as it supports sparse matrices, parallel computing (thanks to the barycenters of the interconnection matrices can be independently calculated), and the application of advanced (integer) sorting algorithms.

3.4 Performance Evaluation

The proposed method can be considered as a visualization tool, similarly to VAT (Visual Assessment of clustering Tendency) [2]. Although the resulted plots are informative, in most of the cases the numerical evaluation of the results is also necessary.

The first and most obvious measurement of the performance of the crossing minimization algorithms is based on the counting of the edge crossings. If we denote the rearranged $\boldsymbol{A}^{(i)}[\boldsymbol{o}_0,\boldsymbol{o}_i]$ matrix as $\boldsymbol{M}^{(i)}$, the number of crossings of the $v_{i,j}$ - th and $v_{i,k}$ - th vertices (represented by the j - th and k - th rows of the matrix) can be calculated as

$$nc^{(i)}(j,k) = \sum_{a=1}^{N_i-1}\sum_{b=a+1}^{N_i} m_{jb}^{(i)}\, m_{ka}^{(i)} \tag{9}$$

The total number of crossings of $\boldsymbol{M}^{(i)}$ can be calculated based on the sum of these $nc^{(i)}(j,k)$ values:

$$nc^{(i)} = \sum_{j=1}^{N_0-1}\sum_{k=j+1}^{N_0} nc^{(i)}(j,k) \tag{10}$$

Since the crossings of the E_i and E_j $\forall i \neq j$ edges are not taken into account, the total number of crossings is calculated as

$$nc = \sum_{i=1}^{n} nc^{(i)}$$

The effectiveness of the ordering from the expected block-diagonality of the resulted $\boldsymbol{M}^{(i)}$ matrices can be measured based on the distance of the nonnegative elements from the diagonal

$$d_{j,k}^{(i)} = m_{jk}^{(i)} \left| \frac{j}{N_0} - \frac{k}{N_i} \right|.$$

The percentage of the non-negative neighbors is also informative to represent the coherence of the resulted "maps", $nn_{j,k}^{(i)}$. Based on the combination of these two measures the we propose the following "goal-oriented" measure of the $\boldsymbol{M}^{(i)}$ ordering:

$$q^{(i)} = \sum_{j=1}^{N_0} \sum_{k=1}^{N_1} \frac{\left(1 - nn_{j,k}^{(i)}\right) d_{j,k}^{(i)}}{m_{j,k}^{(i)}}$$

It is also important to note, that the smaller the q value the better the ordering is in the data matrix.

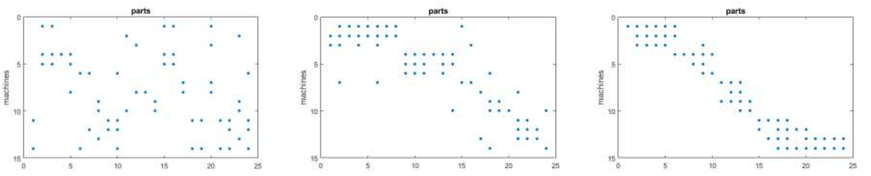
4 Case Studies

Manufacturing cell formation is widely studied and well-documented problem of process flow analysis. In this session, we provide several reproducible comparisons based on the most widely applied benchmark problems. The MATLAB implementation of the algorithm and the related datasets are downloadable from the website of the authors (www.abonyilab.com).

Firstly, we compare our approach with the popular hybrid-heuristic cell formation algorithm [33] and show the applicability of the proposed performance measures. This will be followed by the numerical analysis of the scalability. Finally, an illustrative real-life example will be given.

4.1 Application on Benchmark Problems

Since there are several two-dimensional examples for manufacturing cell formation problems in the literature, we first applied our method on one of those available [33]. The chosen dataset consists of 14 machines and 24 parts. The sparsity pattern of the incidence matrix is depicted in Figure 4(a). The hybrid heuristic algorithm is one of the recently published methods which combines the simulated annealing with genetic methods. This method in many cases outperforms the performance of the classical methods [33]. Solutions provided by the hybrid heuristic algorithm and our method can be seen in Figure 4. As it can be seen, our new approach provides much better block-diagonal ordered solution.



(a) Original dataset

(b) Result of hybrid heuristic method

(c) Result of the proposed method

Figure 4
The original data set and two orderings provided by the hybrid heuristic and our new crossing minimization algorithm

Using the number of edge crossings the hybrid heuristic and the proposed method can be numerically compared. While the original dataset has 674 edge crossings, and the result of the hybrid heuristic method has 256, the proposed crossing minimization algorithm provided an ordering only with 95 edge crossings.

We also compared crossing numbers of another eight benchmark datasets. The results are showed in Figure 5a, where the blue bars show the original, the red bars the hybrid heuristic, and the green ones the number of the crossings of the proposed algorithm. As it can be seen our method outperforms the hybrid heuristic method in every benchmark examples.

A summary of the previously presented *cn* crossing number and *q* block-diagonal ordering results can be seen in Table 1. As this table and Figure 5b show, the crossing minimization also ensures effective block-oriented orderings of these benchmark problems.

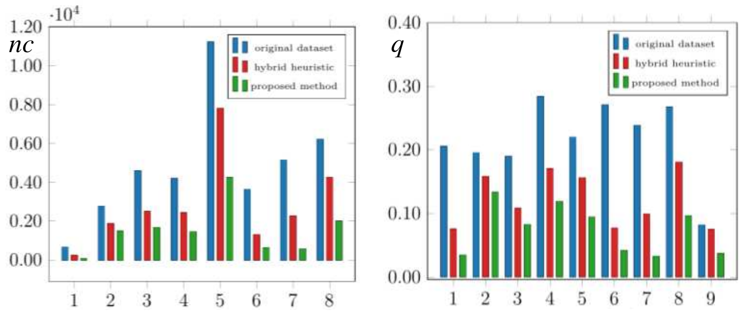


Figure 5
The number of the edge crossings (a) and the measure of the block-diagonal (b) ordering of the benchmark datasets

Table 1
Block-diagonal ordering (q) and crossing number (nc) results on different datasets

Dataset	Original		Hybrid heuristic		Proposed method	
	q	cn	q	cn	q	cn
P1(61)	0.2057	674	0.0756	256	0.0351	95
P2(62)	0.1954	2785	0.1580	1900	0.1336	1506
P3(63)	0.1900	4612	0.1083	2508	0.0826	1688
P4(64)	0.2840	4221	0.1708	2443	0.1193	1467
P5(65)	0.2201	11224	0.1564	7810	0.0948	4258
P6(66)	0.2713	3644	0.0775	1313	0.0422	653
P7(67)	0.2388	5150	0.0991	2283	0.0328	585
P8(68)	0.2675	6221	0.1811	4267	0.0966	2019
P9(69)	0.0786	229570	0.0782	173961	0.0373	131472

4.2 Crossing Minimization of Multidimensional Datasets – Numerical Analysis of the Complexity

While there are only 2D examples in the literature, we have generated several multidimensional benchmark problems to test the presented algorithm. After applying the previously mentioned iterative co-crossing minimization algorithm on the given multidimensional cell formation problem the results are seen in Figure 3, in the second row. As results, each matrix is reordered in a block-diagonal way.

In the following, we validate the log-linear scalability of the algorithm. We defined problems with 10, 100 and 500 properties/categories (N_i) of $i=1\text{-}6$ features. Figure 6 shows the computational costs of these cases as we increased the number of objects (N_0) from 10 to 1.000.000. These graphs on the log-log scale show the log-linear complexity of the algorithm.

Figures 6-8 present similar results where the effects of increasing the number of machines, categories and properties are shown.

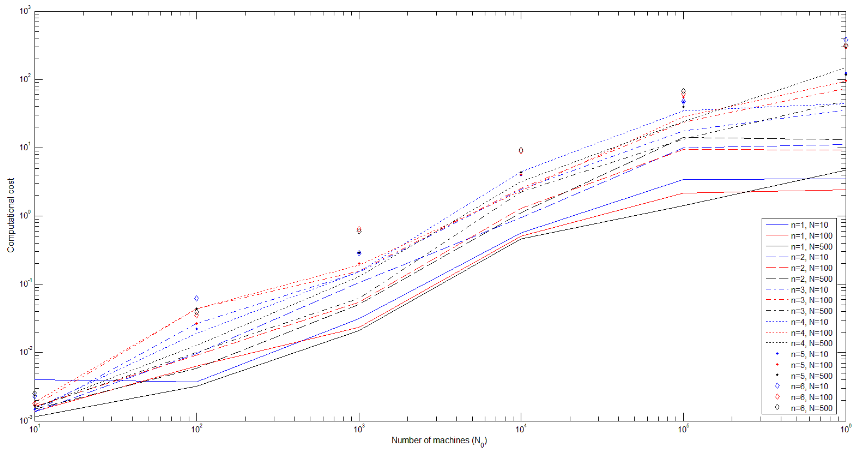


Figure 6

The computational costs of the proposed co-crossing minimization algorithm on different datasets

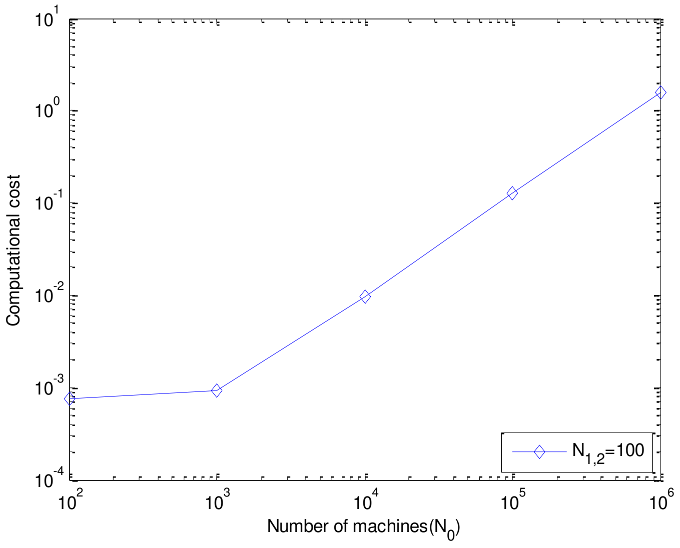


Figure 7

The increase of the number of machines linearly increases the computational complexity

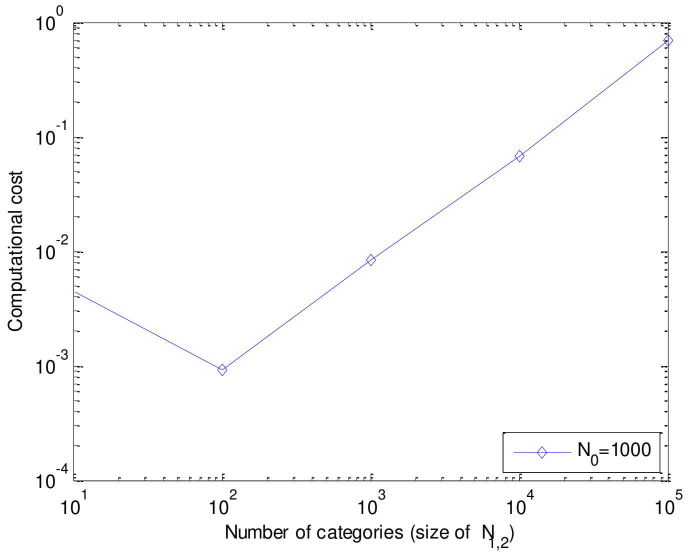


Figure 8

The increase of the number of categories linearly increases the computational complexity

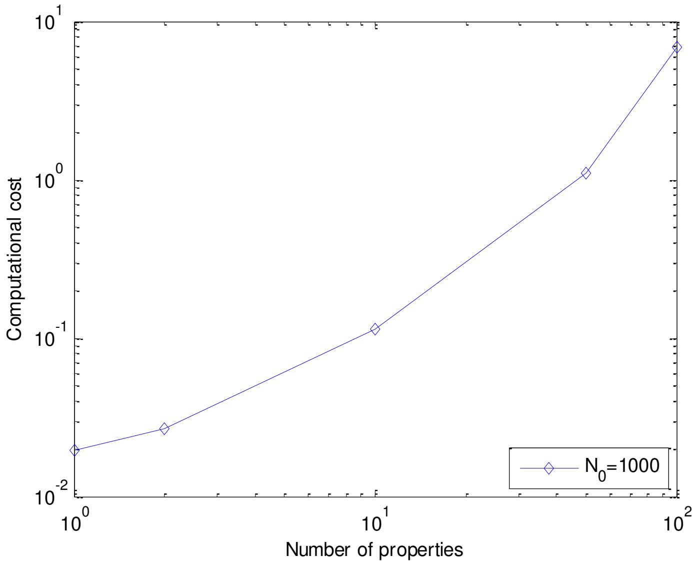


Figure 9

Effect of the increase of the number of properties (*n*)

As we will see, the results confirm the theoretical considerations and the industrial-scale applicability of the method.

4.3 Application on Real Life Problem

The previously presented multivariate co-crossing minimization method was applied on a real life example as well. We used this method on a production line analysis problem, where the primary task is the optimization of the production line of that produces over 5000 types of products assembled from several parts. Assuming that the switching time between similar products is less than the switching time between more different products, our aim was to analyze interconnections between parts and products. Since there are several stages of the production and different types of parts are used in different stages we formulated the multivariate model based on the hierarchy of the bill of materials (BOM) [29]. As Fig. 10 illustrates the methodology worked perfectly, we were able to sort the products according to their similarity.

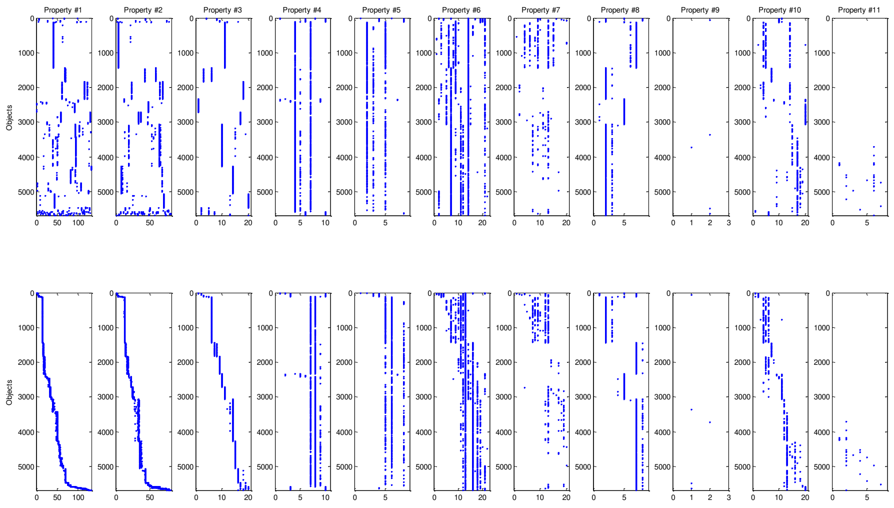


Figure 10

Real-life example for shorting products based on bill of materials

Conclusions

Thanks to the fourth industrial revolution production processes are becoming more and more integrated. This integration allows the simultaneous optimization of the whole supply chain. From this viewpoint production flow analysis is becoming an important tool since the analysis of the integrated marketing, design, production, logistic and sales data can effectively support production scheduling and flexible manufacturing cell formation.

Complex and integrated manufacturing processes require more detailed problem representation than used in classical manufacturing cell formation. This means, production lines should be characterized by several features that should be simultaneously analysed. We proposed a novel multipartite graph based representation of these complex production flow analysis problems and proposed an efficiently scalable clustering and visualization algorithm that sequentially reorders the edge crossings of bipartite graphs. The barycentric heuristic based co-crossing minimization method is simultaneously reorders the rows and columns of the interconnection matrices of the features to highlight their hidden block-diagonal structure, which structure supports data visualization easier. The applicability of the proposed co-crossing minimization is illustrated by several case studies.

We also showed that the proposed algorithm requires low computational capacities as it uses simple linear algebraic operations defined on sparse matrices. Another advantage of the method is its capability of parallelization and handling multi-dimensional sparse datasets. Considering these benefits, the proposed co-crossing minimization method can be scaled and used efficiently when we are dealing with large databases.

Acknowledgement

This publication/research has been supported by the National Research, Development and Innovation Office – NKFIH, through the project OTKA – 116674 (Process mining and deep learning in the natural sciences and process development).

Notations

Representation	
G	graph
V	vertex set of a G graph
V_i	set of vertices, set of objects/properties
$v_{i,j}$	j-th vertex (node) of the i-th set of vertices
N	number of nodes in a network/graph
N_i	number of nodes in the i-th set of vertices
E	edge set of a G graph
e_{ij}	edge between node i and j
$\boldsymbol{A}^{(i)}$	incidence (interconnection) matrix
$\boldsymbol{A}^{(i)}[\boldsymbol{o}_0, \boldsymbol{o}_i]$	ordered interconnection matrix
$\boldsymbol{o}_i$	ordering of the i-th vertex set
$\boldsymbol{p}_i$	positions of the vertices according to the $\boldsymbol{o}_i$ ordering
Crossing minimization	
$\boldsymbol{b}_i^C, \boldsymbol{b}_i^R$	column and row barycenters (vectors)
$\boldsymbol{s}_i^C, \boldsymbol{s}_i^R$	sum of the coumns and rows of $\boldsymbol{A}^{(i}$
Metrics	
$nc^{(i)}(j,k)$	number of crossings of the $v_{i,j}$ - th and $v_{i,k}$ - th vertices
$d_{j,k}^{(i)}$	diagonal distance
$q^{(i)}$	block-oriented quality of the ordering of the i-th intterconneciton matrix

References

[1] Ahmad, W., & Khokhar, A., cHawk : An Efficient Biclustering Algorithm based on Bipartite Graph Crossing Minimization. Computer Engineering, (May 2008) 249-263, http://doi.org/10.1021/es0602492, 2007

[2] Bezdek, J. C., Hathaway, R. J.: Vat: A Tool for Visual Assessment of (Cluster) Tendency, International Joint Conference on Neural Networks IJCNN'02, Vol. 3, pp. 2225-2230, 2002

[3] Chen, C. L. P., Zhang, C.-Y.: Data-Intensive Applications, Challenges, Techniques and Technologies: A Survey on Big Data, Information Sciences Vol. 275, pp. 314-347, 2014

[4] Chimani, M., Mutzel, P., & Bomze, I., A New Approach to Exact Crossing Minimization. Proc. of European Symposium on Algorithms (ESA~2008), 284-296, http://doi.org/10.1007/978-3-540-87744-8_24, 2008

[5] Dimopoulos, C., Mort, N.: A Hierarchical Clustering Methodology based on Genetic Programming for the Solution of Simple Cell-Formation Problems. International Journal of Production Research, Vol. 39, pp. 1-19, 2001

[6] Eiglsperger, M., Siebenhaller, M., & Kaufmann, M., An Efficient Implementation of Sugiyama's Algorithm for Layered Graph Drawing. Journal of Graph Algorithms and Applications, 9(3), 305-325, http://doi.org/10.7155/jgaa.00111, 2005

[7] Erten, C., & Sözdinler, M., A Robust Biclustering Method Based on Crossing Minimization in Bipartite Graphs. 16th International Symposium on Graph Drawing, 5417, 439-440-440, http://doi.org/10.1007/978-3-642-00219-9_45, 2009

[8] Fan, C.-Y., Fan, P.-S., Chan, T.-Y., & Chang, S.-H., Using Hybrid Data Mining and Machine Learning Clustering Analysis to Predict the Turnover Rate for Technology Professionals. Expert Systems with Applications, 39(10), 8844-8851. http://doi.org/10.1016/j.eswa.2012.02.005, 2012

[9] Gansner, E. R., Koutsofios, E., North, S. C., & Vo, K. P. a V. K. P., A Technique for Drawing Directed Graphs- A Technique for Drawing Directed Graphs. Software Engineering, IEEE Transactions on, 19(3), 214-230, http://doi.org/10.1109/32.221135, 1993

[10] Gao, B., Liu, T.-Y., Zheng, X., Cheng, Q.-S., Ma, W.-Y.: Consistent Bipartite Graph Co-Partitioning for Star-structured High-Order Heterogeneous Data Co-Clustering, Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, 2005

[11] Goncalves, J. F., Resende, M. G. C.: An Evolutionary Algorithm for Manufacturing Cell Formation. Computers and Industrial Engineering, Vol. 47, pp. 247-273, 2004

[12] Hazena, B. T., Booneb, C. A., Ezellc, J. D., Jones-Farmerc, L. A.: Data Quality for Data Science, Predictive Analytics, and Big Data in Supply Chain Management: An Introduction to the Problem and Suggestions for Research and Applications, International Journal of Production Economics Vol. 154, pp. 72-80, 2014

[13] Jünger, M., & Mutzel, P. 2-Layer Straightline Crossing Minimization: Performance of Exact and Heuristic Algorithms. Journal of Graph Algorithms and Applications, 1(1), 1-25, http://doi.org/10.7155/jgaa.00001, 1997

[14] Khan, S., Bilal, M., Sharif, M., & Khan, F. A., A Solution to Bipartite Drawing Problem using Genetic Algorithm. Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6728 LNCS(PART 1), 530-538, http://doi.org/10.1007/978-3-642-21515-5_63, 2011

[15] Kusiak, A., Cho, M.: Similarity Coefficient Algorithms for Solving the Group Technology Problem, International Journal of Production Research Vol. 30, pp. 2633-2646, 2007

[16] Lee, J., Kao, H.-A., Yang, S.: Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment, Procedia CIRP Vol. 16, pp. 3-8, 2014

[17] Lee, J., Lapira, E., Bagheri, B., & Kao, H., Recent Advances and Trends in Predictive Manufacturing Systems in Big Data Environment. Manufacturing Letters, 1(1), 38-41, http://doi.org/10.1016/j.mfglet.2013.09.005, 2013

[18] Li, S., & Mehrabadi, H., Generation of Block Diagonal forms Using Hierarchical Clustering for Cell Formation Problems. Procedia CIRP, 17, 44-49, http://doi.org/10.1016/j.procir.2014.01.143, 2014

[19] Mahdavi, I., Teymourian, E., Baher, N. T., & Kayvanfar, V., An Integrated Model for Solving Cell Formation and Cell Layout Problem Simultaneously Considering New Situations. Journal of Manufacturing Systems, 32(4), 655-663, http://doi.org/10.1016/j.jmsy.2013.02.003, 2013

[20] Naadimuthu, G., Gultom, P., & Lee, E. S., Fuzzy Clustering in Cell Formation with Multiple Attributes. Computers and Mathematics with Applications, 59(9), 3137-3147, http://doi.org/10.1016/j.camwa.2010.02.038, 2010

[21] Oliveira, S., Ribeiro, J. F. F., & Seok, S. C., A Comparative Study of Similarity Measures for Manufacturing Cell Formation. Journal of

Manufacturing Systems, 27(1), 19-25, http://doi.org/10.1016/j.jmsy.2008.07.002, 2008

[22] Oliveira, S., Ribeiro, J. F. F., & Seok, S. C., A Spectral Clustering Algorithm for Manufacturing Cell Formation. Computers & Industrial Engineering, 57(3), 1008-1014, http://doi.org/10.1016/j.cie.2009.04.008, 2009

[23] Onwubulo, G. C., Mutingi, M.: A Genetic Algorithm Approach to Cellular Manufacturing Systems. Computers and Industrial Engineering, Vol. 39, pp. 125-144, 2001

[24] Sagiroglu, S., Sinanc, D.: Big Data: A review, International Conference on Collaboration Technologies and Systems, pp. 42-47, 2013

[25] Schuh, G., Potente, T., Wesch-Potente, C., Weber, A. J., Prote, J.-P.: Collaboration Mechanisms to increase Productivity in the Context of Industrie 4.0, ScienceDirect Procedia CIRP Vol. 19, pp. 51-56, 2014

[26] Shahrokhi, F., Sýkora, O., Székely, L. A., & Vrťo, I., On Bipartite Drawings and the Linear Arrangement Problem. SIAM Journal on Computing, 30(6), 1773, http://doi.org/10.1137/S0097539797331671, 2001

[27] Shiyas, C. R., & Madhusudanan Pillai, V., A Mathematical Programming Model for Manufacturing Cell Formation to Develop Multiple Configurations. Journal of Manufacturing Systems, 33(1), 149-158, http://doi.org/10.1016/j.jmsy.2013.10.002, 2014

[28] Stallmann, M., Brglez, F., & Ghosh, D., Heuristics, Experimental Subjects, and Treatment Evaluation in Bigraph Crossing Minimization. Journal of Experimental Algorithmics, 6(212), 8–es, http://doi.org/10.1145/945394.945402, 2001

[29] Stonebraker, P. W.: Restructuring the Bill of Material for Productivity: A Strategic Evaluation of Product Configuration, International Journal of Production Economics Vol. 45, pp. 251-260, 1996

[30] Sugiyama, K., Tagawa, S., & Toda, M., Methods for Visual Understanding of Hierarchical System Structures. IEEE Transactions on Systems, Man, and Cybernetics, 11(2), 109-125, http://doi.org/10.1109/TSMC.1981.4308636, 1981

[31] Warfield, J. N., Crossing Theory and Hierarchy Mapping. IEEE Transactions on Systems, Man, and Cybernetics, 7(7), 505-523, http://doi.org/10.1109/TSMC.1977.4309760, 1977

[32] Wu, T.-H., Chang, C.-C., & Chung, S.-H., A Simulated Annealing Algorithm for Manufacturing Cell Formation Problems. Expert Systems with Applications, 34, 1609-1617, http://doi.org/10.1016/j.eswa.2007.01.012, 2008

[33] Wu, T.-H., Chang, C.-C., & Yeh, J.-Y., A Hybrid Heuristic Algorithm Adopting both Boltzmann Function and Mutation Operator for Manufacturing Cell Formation Problems. International Journal of Production Economics, 120(2), 669-688, http://doi.org/10.1016/j.ijpe.2009.04.015, 2009

[34] Wu, T.-H., Low, C., Wu, W.-T.: A Tabu Search Approach to the Cell Formation Problem. International Journal of Advanced Manufacturing Technology Vol. 23, pp. 916-924, 2004

Conceptual Design of Document NoSQL Database with Formal Concept Analysis

Viorica Varga, Katalin Tünde Jánosi-Rancz, Balázs Kálmán

Babeş-Bolyai University, Kogălniceanu 1, 400084 Cluj-Napoca, Romania
Sapientia Hungarian University of Transilvania, Corunca 1C, 540485 Târgu Mureş, Romania
Babeş-Bolyai University, Kogălniceanu 1, 400084 Cluj-Napoca, Romania
ivarga@cs.ubbcluj.ro, tsuto@ms.sapientia.ro, kbim1225@scs.ubbcluj.ro

Abstract: Early Big Data solutions were not based on database management system principles. As the popularity of these solutions have increased and are applied in more data management scenarios in the recent years, DBMS principles, are being recognized as important factors and are becoming important in newly developed solutions. Big Data collections do not enforce document structure, but just because a data store is schema-less, it does not mean the structure of the stored documents will not play an important role in the overall performance and flexibility of an application. In this paper we will explore a method for the conceptual modeling for document based databases, using Formal Concept Analysis (FCA). We have shown that FCA is a valuable visual analyzer for large-scale data, for example, offering a means of reading the possibility of nested scheme design from the built concept lattice. Results of experiments using our method have proven that decisions affecting the modeling of data can affect application performance and database capacity.

Keywords: conceptual design; NoSQL database; document store; Formal Concept Analysis

1 Introduction

Data is growing exponetially in the digital world, increasing in volume, variety (structured, un-structured or hybrid) and velocity (high speed of growth). This phenomenon is refered to as ‘Big Data’. This growing data collection is so large that it can not be effectively managed using conventional relational data management tools. To handle this problem, traditional RDBMS are complemented with rich set systems: NoSQL [1, 4, 20] data stores, NewSQL and Search-based systems.

NoSQL systems generally have some common features. The first is the ability to horizontally scale simple operations over many servers. They can replicate and partition data over many servers with a simple call level interface. These new systems accept a weaker concurrency model, than the ACID transactions of relational database systems. They are often flexible enough to accommodate semi-structured and sparse data sets [20]. NoSQL data stores vary in their data and query model. The most common categorization of these systems is by data model, distinguishing key-value stores, document stores, column-family stores, and graph databases [4]. Key-value stores data structure is composed of a unique key and an opaque value. Document based NoSQL systems also store key-value pairs, but the values are structured as documents. The document is a set of name-value pairs, usually in JSON (JavaScript Object Notation) [8] format or the binary representation BSON. Name-value pairs represent the properties of data objects. Values can be scalar or appear as lists, but may contain nested documents too. Column-family stores manage records with properties. A schema for a column-family declares property families, and new properties can be added to a property family ad hoc. Graph databases represent data in graph format; objects are stored in nodes and their relationships in the edges.

Most NoSQL data stores do not enforce any structural constraints on the data; they are usually referenced as schema-less data. But programmatically accessing this data, it is important to have some notion about its structure. Without knowing the general structure of the data, it is nearly impossible to perform any application development or data analysis [15]. Many NoSQL data stores provide a declarative query language. Developers need to know which attributes are present (or absent) in persisted objects in order to formulate queries. For OLAP-style data analysis, developers also need to know which structure to expect when parsing JSON documents. So, these tasks require some form of schema description. The structure of the stored documents plays an important role in the overall performance of the application.

In this paper we focus on designing document stores, which are based on a semi-structured data model, implemented as JSON, XML or BSON format. The design of any database follows a well-defined methodology for conceptual, logical, and physical data modeling. A prevalent model in the conceptual database design is the Entity-Relationship (E-R) model. The semi-structured data has a loose schema: a core of attributes is shared by all objects, but many individual variants are possible. A hierarchical structure of the data is a common design opportunity for embedding complex entities.

Formal Concept Analysis (FCA) [11] supports knowledge discovery and knowledge representation [14]. Current FCA methods have the capabilities for taking into account the presence and management of relational attributes or links in the data [18, 19].

In this paper a Formal Concept Analysis (FCA) approach for conceptual modeling of document based databases is proposed. A mapping from Entity-Relationship model to a schema for semi-structured data in the form of concept lattices are presented for relations of type One-to-One, One-to-Many and Many-to-Many. For different relationship types we obtain different concept lattices [22]. The possibility of nested scheme design can be read from the lattices.

We propose a Relational Concept Analysis (RCA) grounded approach to conceptual document based NoSQL database design, one which is a data model, for the systems that can store and manage Big Data.

In Section 2, we assume familiarity with the basic notions of FCA and RCA and after that, in Section 3, we discuss how relational modeling can be emulated in the case of document databases using RCA. Section 4 presents the experiments on DBLP [6] bibliography dataset. Finally, we finish our work with conclusions.

2 Preliminaries and Basic Notions in FCA and RCA

Our research is mainly based on the mathematical foundations of FCA and in this section we introduce the necessary formal background.

FCA is a data analysis method which enables the discovery of knowledge hidden within data, such as association rule mining, ontology engineering, machine learning. FCA actually provides support for processing large dynamic complex data.

In FCA, data are represented by a formal context, which will contain objects and attributes. From the context, formal concepts are generated by grouping objects which have the same set of attributes. Each formal context is transformed into a concept lattice, which forms the basis for further data analysis.

Definition 1. $(Formal\ context)$. A formal context $K = (G, M, I)$ consists of two sets G and M and a binary relation I between G and M. Elements of G are called objects while elements of M are called attributes of the context. The fact $(g, m) \in I$ is interpreted as "the object g has attribute m".

Example 1. Consider the set of objects $G = \{Bicycle, Chariot, Boat,\ Car,\ Airplane\}$. Consider the set of attributes $M = \{Has\ wings,\ Has\ engine,\ Has\ windows,\ Has\ wheels\}$ that are properties that vehicles may have or not. Table 1 gives an example of formal context (G, M, I), the X indicates that a certain object has a certain attribute.

Table 1
An example of formal context $K = (G, M, I)$

	Has wings	Has engine	Has windows	Has wheels
Bicycle				X
Chariot				X
Boat		X	X	
Car		X	X	X
Airplane	X	X	X	X

Definition 2. ($Formal\ concept$). A formal concept of a context (G, M, I) is a pair (A, B) with $A \subseteq G, B \subseteq M, A' = B$ and $B' = A$. A is called the extent of the concept (A, B) while B is called its intent. A' and B' define a Galois connection between the power sets of G and M. The set of all formal concepts of a context (G, M, I) is written $\mathfrak{B}(G, M, I)$. Concepts are partially ordered by $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$ $(\Leftrightarrow\ B_2 \subseteq\ B_1)$. (A_1, B_1) is called sub-concept and (A_2, B_2) a super-concept.

Example 2. From the previous example, it directly follows that the pair $(\{Boat, Car, Air - plane\}, \{Has\ engine, Has\ windows\})$ is a formal concept. A Galois connection implies that if one makes the sets of one type larger, they correspond to smaller sets of the other type, and vice versa. Using this concept, if $Has\ wings$ is added to the list of attributes, the set of vehicles reduces to $\{Airplane\}$.

FCA organizes the information through concept lattices, which fundamentally comprises a partial order, modeling the subconcept-superconcept hierarchy. Concept lattice is the common name for a specialized form of Hasse diagram that is used in conceptual data processing.

Definition 3. ($Concept\ lattice$). The set of all formal concepts from a context $K = (G, M, I)$ ordered with the relation $\leq$ form a complete lattice called concept lattice of (G, M, I) and denoted by $\mathfrak{B}(G, M, I)$.

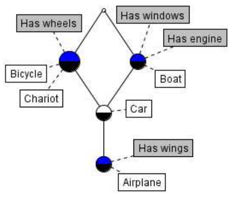


Figure 1
Concept lattice raised from Table 1

Figure 1 shows the concept lattice, associated with Table 1. A line diagram consists of circles, lines and the names of all objects and all attributes of the given context appearing as labels. Each node in the lattice corresponds to a formal concept while a line denotes an order relation between two concepts. An object g has an attribute m if and only if there is an upwards leading path from the circle named by g to the circle named by m.

Real-life data are often more complex than those given by a formal context. There are several extensions of FCA to handle complex data, such as *Conceptual scaling* [23] (where complex data are turned into binary contexts by using scales), *Pattern structures* [24] (a general approach to conceptual scaling by giving a direct method of knowledge discovery in complex data such as logical formulas, graphs, strings, tuples of numerical intervals), *Power Context Families* [25], *Relational Concept Analysis* [18] and *Logical Concept Analysis* (for analyzing arbitrary relations between objects), *Triadic Concept Analysis* [26] (to analyze three-dimensional data), *Fuzzy FCA* [23] and *Rough FCA* [16] (were developed to work with uncertain data and approximations). These papers have proven that FCA is feasible for Big Data. The approach in [16] makes FCA useful for analyzing extremely large data. Also, FCA contributes to the analysis and mining of social networks [27] such as affiliation and interaction networks and possibly more complex structures using this theory and some of its extensions. [17] gives an overview of several extensions of the main FCA model, and shows that FCA algorithms are efficient from both theoretical and practical points of view [25]. Its time complexity and performance proves its supremacy over concurrent methods and allows us to use it for Big Data problems.

In this paper, to cope with complex data, we use the following FCA extensions: conceptual scaling [23] and Relational Concept Analysis (RCA) [18, 19]. The process of deriving a one-valued context from a many-valued context is called *conceptual scaling*.

Definition 4. $(Many - valued\ context)$. A many-valued context (G, M, W, I) consists of sets G, M and W and a ternary relation I between those three sets, i.e. $I \subseteq G \times M \times W$, for which it holds that $(g, m, w) \in I\ and\ (g, m, v) \in I$ always imply $w = v$.

Elements of G are still called objects. Elements of M are called (many-valued) attributes. Elements of W are called attribute values. Accordingly, the fact $(g, m, w) \in I$ means "the attribute m takes value w for object g", simply written as $m(g) = w$.

Standard FCA is restricted to data sets that are either already represented as binary relations or that can be easily transformed into such a representation [14]. Relational Concept Analysis (RCA) [18, 19] extends standard FCA by taking relations between objects into account.

The objective of RCA is to build a set of lattices whose concepts are related by relational attributes, similar to UML associations.

In RCA, data are organized within a structure composed of a set of contexts and a set of binary relations.

Definition 5. $(Relational\ context\ family)$. A relational context family F is a pair (K, R), where K is a set of contexts $K_i = (G_i, M_i, I_i)$ - with objects G_i, properties M_i and a relationship I_i between these objects and properties; and R is a set of Object-Object relations $r_k \subseteq G_i \times G_j$ where G_i and G_j are the object sets of the formal contexts K_i and K_j. The structure (K, R) can be compared to a relational database schema, including both classes of individuals and classes of relations.

For example Table 2 shows two Object – Attribute contexts of some authors and their papers presented in different conferences. Table 3 represents the Object-Object context related to Table 2.

In RCA data tables are iteratively merged into one in the following way: in each step all formal concepts are computed of one data table and these concepts are used as additional attributes for the merged data table. After obtaining a final merged data table, all formal concepts are extracted.

The RCA methodology is the following: given the RCF Object-Attribute contexts and Object-Object contexts - the concept lattice is built for each Object-Attribute context at first, then relational scaling is applied to all Object-Object contexts and relational extension of each Object-Attribute context is built. Finally the concept lattice for each relational extension is constructed, thus a concept lattice family is obtained.

Table 2
RCF: Object-Attributes contexts

Author	Lecturer	PhD Student	Professor
Adam		X	
Tom			X
Jack			X
Lena		X	
Jim	X		
Sophia	X		
Lucia			X
Mike			X

Paper	ICFCA	ADBIS	VLDB	SIGMO	KDD
FESTA	X				
RK			X		
ELKA					X
OpenR					X
XKENA		X			

Table 3
RCF: Object-Object context

	FESTA	RK	ELKA	OpenR	XKENA
Adam	X		X		X
Tom	X				X
Jack		X			
Lena		X			
Jim					
Sophia	X		X		
Lucia				X	
Mike				X	X

3 Document Store Data Modeling using RCA

In this section we investigate the architectural challenges in document based NoSQL database design. The Entity-Relationship diagram is the most common tool for conceptual schema design. It is independent of the physical implementation of the database. It can be transformed to any other data model: relational, object oriented, hierarchical, semi-structured etc. In relational data model design, the tables obtained from the E-R diagram can be analyzed for different normal forms. If an E-R diagram is carefully designed, identifying all entities correctly, the tables generated from the E-R diagram should not need further normalization.

Documents using the semi-structured data model may contain redundant information and may be prone to update anomalies. Such problems are caused by some functional dependencies. XML Functional Dependency (FD) and normal form XNF for XML documents were defined by Arenas and Libkin introducing the so-called tree tuple approach [2]. Yu and Jagadish [13] show, that these XML FD notions are insufficient and propose a Generalized Tree Tuple (GTT) based XML functional dependency and key notion, which includes particular redundancies involving set elements. Based on these concepts, they present the GTT-XNF normal form. We offer some FCA based tools for finding functional dependencies in XML documents and propose a correct XML scheme in [14, 5].

Our FCA based method gives a mapping of E-R diagrams to RCA, using a graphical representation of binary relationships having two cardinalities. We consider that carefully designed relationships using our method will produce a document in XNF normal form. The method is simpler than the normalization process of semi-structured data.

3.1 Conceptual Design of Semi-structured Data

We model the Entity-Relationship (E-R) schema as a Relational Context Family as follows: The E-R schema consists of entity sets, attributes, and relationships between entity sets. Since attributes are properties representative of real world objects, they are many-valued, hence we can represent every entity set of the E-R model as a *many-valued context*. Let $E_1, E_2, \dots, E_n$ be entity sets of an E-R diagram. Every entity set E_i, $i = 1, \dots, n$ will be modelled as a many-valued context. The objects of the many-valued context modelled for entity set E_i will be the entities from E_i. Let us denote by $A_{i_1}, A_{i_2}, \dots, A_{i_k}$ the attributes of entity set E_i, which will be the attributes of the many-valued context. Entity sets are connected by relations. The relationship between entities from E_i and E_j will be represented using their entity keys in a formal context.

Let R_{ij} be a relation between the entity sets E_i and E_j. The Object-Object context of R_{ij} is defined as the context having as object sets different key values of E_i and

E_j. The relationship between objects of E_i and objects of E_j is given in the incidence table.

The simplest form of a relationship is the binary relation. Relationships involving more than two entity sets are called n-ary relations. Binary relationships have two cardinality constraints [9] of the form (x; y), where x, y are natural numbers, x specifies the minimum cardinality and y the maximum participation constraint. Let us consider two entity sets E_1 and E_2 and a binary relation R between them with left cardinality constraint (x_1; y_1) and right cardinality constraint (x_2; y_2) denoted with: $E_1 \xleftrightarrow{(x_1,y_1)} R \xleftrightarrow{(x_2,y_2)} E_2$.

Based on their maximum cardinality constraints binary relationships are:

(i) One-to-One, if both roles have maximum cardinality 1

(ii) One-to-Many, if one role has maximum cardinality 1 and the other one has maximum cardinality N

(iii) Many-to-Many, if both roles have maximum cardinality N

The same E-R conceptual schema can be mapped into a different schema for semi-structured data. We consider two alternative ways of mapping E-R conceptual schemas into schema for semi-structured data: a relational-style (flat) design methodology and a nesting (or embedding) approach. In the flat schema model each entity is at the same level of the hierarchy and uses references to another entity as foreign keys in relational databases, the schema never nests. The nested schema embed entities as much as possible. M. Franceschet et al. in [10] proved that both validation of data and query processing are globally more efficient with nested schemas than with flat ones. Highly nested XML schemas reduce the number of expensive join operations.

To optimize application performance and reliability, a NoSQL schema must be driven by the application's intended purpose; it is about our data and how it is used. In the design process one has to decide whether flat or nested structuring optimizes one's schema. Both designs have advantages and disadvantages. The main advantage of the nested data model is that one can retrieve the complete class master information with one query. The main disadvantage of it is that there is no way of accessing the nested details as stand-alone entities.

In our method we will first create the Object-Object context of our data, build the concept lattice related to that context and read its background knowledge. From the graphical representation of the built concept lattice, one can discover and understand the conceptual relationships within a given set of data. Reading the built concept lattice we can decide if nesting is feasible or not based on Algorithm 1. We distinguish between the nodes of the Object-Object lattice the top node, the bottom node and intermediate nodes. The concept nodes of the Object-Object lattice are labeled with the keys of the entities.

Algorithm 1.

```
Input: a concept lattice
Output: nesting is possible or not
begin
if intermediate nodes are situated in one level then
   if intermediate nodes are labeled with utmost one key from $E_1$
     and utmost one key from $E_2$ then
        the relationship is One-to-One;
        if top node has labels and bottom node has labels then
              nesting is not possible;
        else
              nesting is possible;
        end;
   end;
   if intermediate nodes are labeled with more keys from $E_1$
     and utmost one key from $E_2$ then
        the relationship is Many-to-One;
        if top node has labels from $E_1$ or bottom node has labels from $E_1$ then
              nesting is not possible;
        else
              nesting is possible; // $E_1$ nested in $E_2$;
        end;
   end;
else
   the relationship is Many-to-Many;
   nesting is not possible;
end;
end;
```

Next, we will discuss in detail each of these three basic forms of relations: One-to-One, One-to-Many and Many-to-Many.

3.1.1 One-to-One Relationship Mapping

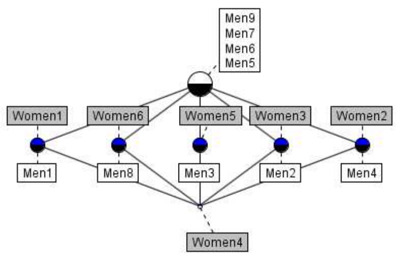
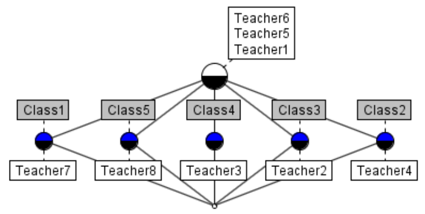
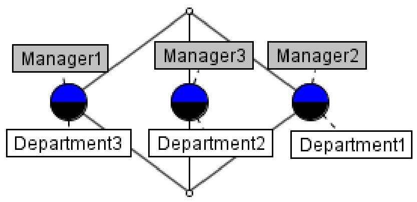
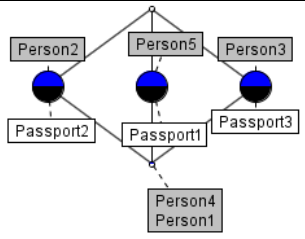
In general, the nested data model is the most advantageous, but when modeling One-to-One relationships, we have to carefully think through the structure of our data. Let $\boldsymbol{E_1}$ and $\boldsymbol{E_2}$ be two entity sets and R a One-to-One relationship between them. The question is how to nest them: $\boldsymbol{E_1}$ in $\boldsymbol{E_2}$ or $\boldsymbol{E_2}$ in $\boldsymbol{E_1}$? If we are using a nested data model and there are elements of $\boldsymbol{E_1}$ which are not related to any element of $\boldsymbol{E_2}$, or elements of $\boldsymbol{E_2}$ which are not related to any element of $\boldsymbol{E_1}$, then we may lose some elements of $\boldsymbol{E_1}$ or $\boldsymbol{E_2}$. Our method, using RCA helps one to decide if nesting is possible and provides the answer as how the entities should be nested.

In order to decide which data model to use we will firstly create the Object-Object context of our data and build the related concept lattice. The concept lattice will be the basis of further analysis. (We will present the Object-Object contexts only in case of One-to-Many relationships).

Depending on its cardinality the One-to-One relationship has four cases. Table 4 gives examples of these cases. We can observe, that for every concept (excepting the top and bottom of the lattice) there exists one element from $\boldsymbol{E_1}$ and one element from $\boldsymbol{E_2}$. The concept lattices vary only in bottom and top elements, but these are decisive in the nested data scheme. The elements of $\boldsymbol{E_1}$ and $\boldsymbol{E_2}$, which are not related to each other, will appear in the top or bottom of the lattice.

Table 4

Concept lattices of Object-Object context in case of One-to-One relationships

 Men9 Men7 Men6 Men5 Women1 Women6 Women5 Women3 Women2 Men1 Men8 Men3 Men2 Men4 Women4	 Teacher6 Teacher5 Teacher1 Class1 Class5 Class4 Class3 Class2 Teacher7 Teacher8 Teacher3 Teacher2 Teacher4
a) *Men* $\overset{(0,1)}{\leftrightarrow}\boldsymbol{spouse}\overset{(0,1)}{\leftrightarrow}$ *Women* There are dangling entities in $\boldsymbol{Men}$ and $\boldsymbol{Women}$, the top and bottom of the lattice contains non empty intent/extent, thus nested design is not possible. The inclusion of $\boldsymbol{Women}$ in $\boldsymbol{Men}$ would lead to the loss of $\boldsymbol{Women}$ elements which are not related to elements of $\boldsymbol{Men}$ and vice-versa.	b) *Teacher* $\overset{(0,1)}{\leftrightarrow}\boldsymbol{responsible}\overset{(1,1)}{\leftrightarrow}$ *Class* Every element of *Class* is related with one element of $Teacher$. The nested design is possible, including *Class* in $Teacher$, but not the inverse.
 Manager1 Manager3 Manager2 Department3 Department2 Department1	 Person2 Person5 Person3 Passport2 Passport1 Passport3 Person4 Person1
c) *Department* $\overset{(1,1)}{\leftrightarrow}\boldsymbol{managed}\overset{(1,1)}{\leftrightarrow}$ *Manager* The intent of the top element and extent of the bottom element of the concept lattice are empty, thus both nesting is correct, we can include $\boldsymbol{Department}$ in $\boldsymbol{Manager}$ or $\boldsymbol{Manager}$ in $\boldsymbol{Department}$.	d) *Passport* $\overset{(1,1)}{\leftrightarrow}\boldsymbol{managed}\overset{(0,1)}{\leftrightarrow}$ *Person* This case allows the existence of elements of $\boldsymbol{Person}$ not related to elements of $\boldsymbol{Passport}$, but every element of $\boldsymbol{Passport}$ is related to one element of $\boldsymbol{Person}$. The nested design is possible, $\boldsymbol{Passport}$ can be included in $\boldsymbol{Person}$, but not the inverse.

When one knows how to read a concept lattice, it can indeed provide valuable information. In [14] we have shown that an XML database can be translated into a power context family and that the functional dependencies of such a database correspond to FCA implications in a certain formal context. We are also able to visualize the structures of the different normal forms in a lattice. In Table 4 the possibility of nested scheme design can be read from the lattices. At this point, FCA proves to be a valuable tool for the design of such schemas. It gives an expressive graphical representation of the relationships between entities. This paper does not claim that these visualizations solve any computational problem or create new means for practical implementations. Instead, the visualizations are meant to serve as explanatory aids, to help us to decide which schema design to choose. NoSQL data modeling often requires a deeper understanding of data structures and algorithms than relational database modeling does. To the best of our knowledge, currently there are no applications which help in choosing between NoSQL data modeling techniques. Thus an FCA based visual aid is beneficial.

3.1.2 One-to-Many Relationship Mapping

In general, the N-side of a relationship is nested, if there is no need to access the embedded object outside the context of the parent object or one can use an array of references to the N-side objects if the N-side objects must stand alone, but we have to carefully think through the structure of our data.

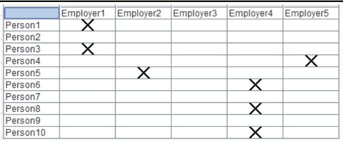
In order to decide which data model to use we will first create the Object-Object context of our documents and build the related concept lattice. Then we analyze the obtained lattice, reading the background knowledge from it, using Algorithm 1, to determine if nesting is possible.

Depending on its cardinality the One-to-Many relationship has four cases. We present three of them in Table 6. We denote $\boldsymbol{E_1}$ the entities of the left hand side of the One-to-Many relation and $\boldsymbol{E_2}$ the right hand side respectively, as a working example to generate the concept lattice, $\boldsymbol{E_1}$ representing the N side of the One-to-Many relationship. If we analyze the obtained lattices (Table 5) we can observe, that for every concept (excepting the top and bottom) there exists one element from $\boldsymbol{E_2}$ and N elements from $\boldsymbol{E_1}$. This illustrates the relationship between elements of $\boldsymbol{E_1}$ and elements of $\boldsymbol{E_2}$.

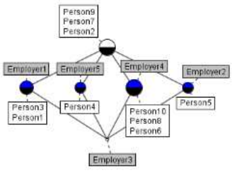
For the sake of simplicity, in the following examples we have presented very small concept lattices, but it has been shown in [3, 12] that readable lattices can be produced from real data sets with a straightforward process of creating sub-contexts.

Table 5

Object-Object contexts and their concept lattices in case of One-to-Many relationships

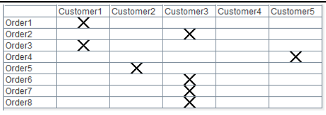


	Employer1	Employer2	Employer3	Employer4	Employer5
Person1	X				
Person2					
Person3	X				
Person4					X
Person5		X			
Person6				X	
Person7					
Person8				X	
Person9					
Person10				X	

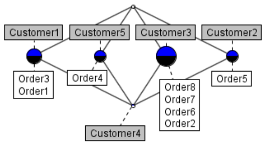


e) *Persons* $\overset{(\mathbf{0,1})}{\leftrightarrow}$***Employed*** $\overset{(\mathbf{0,}\boldsymbol{N})}{\leftrightarrow}$ *Employers*

There are elements of ***Persons*** which are not related to elements ***Employers*** being displayed at the top of the lattice, as well as elements of ***Employers*** which are not related to elements of Persons, appearing at the bottom of the lattice. Having elements in ***Persons*** representing the $\boldsymbol{N}$ side not related to a parent, hierarchical design is not possible. There can be persons who are not employed and employers who have no employed person.

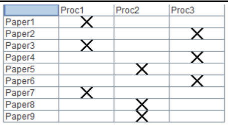


	Customer1	Customer2	Customer3	Customer4	Customer5
Order1	X				
Order2			X		
Order3	X				
Order4					X
Order5		X			
Order6			X		
Order7			X		
Order8			X		

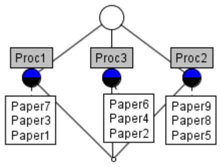


f) *Orders* $\overset{(\mathbf{1,1})}{\leftrightarrow}$***Ordered*** $\overset{(\mathbf{0,}\boldsymbol{N})}{\leftrightarrow}$ *Customers*

There are elements of $Customers$ which are not related to elements of $Orders$, appearing in the bottom of the lattice, but every element of $Orders$ is related with one element of $Customers$. Hierarchical design is possible, including $Orders$ in $Customers$.



	Proc1	Proc2	Proc3
Paper1	X		
Paper2			X
Paper3	X		
Paper4			X
Paper5		X	
Paper6			X
Paper7	X		
Paper8		X	
Paper9		X	



g) *Papers* $\overset{(\mathbf{1,1})}{\leftrightarrow}$***Appeared*** $\overset{(\mathbf{1,}\boldsymbol{N})}{\leftrightarrow}$ *Proceedings*

There are no dangling entities: every element of ***Papers*** is related to one element of ***Proceedings***, and also every element of ***Proceedings*** is related with elements of ***Papers***. Nested structure is possible.

3.1.3 Many-to-Many Relationship Mapping

Concepts of conceptual lattices which represent Many-to-Many relationships are on more hierarchical levels. The top and bottom of the concept lattice are labeled if there are dangling elements in the entity sets A and B.

Example 3. Consider the Many-to-Many relation between Students and Courses: Students $\overset{(0,N)}{\leftrightarrow}$Choose $\overset{(0,N)}{\leftrightarrow}$ Courses. One course can be chosen by 0 or more students and there can be students who choose 0 or more courses.

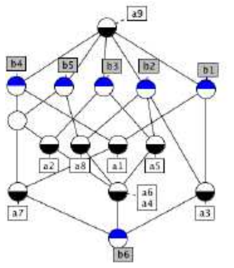


Figure 2
Concept lattice of Many-to-Many relationship

Nested design is not possible in this case.

In the case of Many-to-Many relationships, one can use bi-directional referencing if you are willing to pay the price of not having atomic updates or use of application-level joins, they are barely more expensive than server-side joins if you index correctly and use the projection specifier.

4 Experimental Evaluation

In the following we will use MongoDB for our discussion as it is one of the leading open-source NoSQL databases due to its simplicity, performance, scalability, and active user base. It has to be emphasized that our aproach is available in all semistructured datastores, not only in MongoDb.

We use the DBLP [6] bibliography dataset imported in MongoDB to test queries on flat versus nested structure of the documents. Our experiments were conducted on three computers, both with i7 processors and 4 GB RAM, running with a Windows 7 (64bit) OS.

3.2 Data Structure of Experimental Data

The structure of DBLP data is described in [7], where the flat representation of XML data is used. We import 1.559.572 conference proceedings and 1.232.729 journal articles from DBLP dataset. An *inproceeding* document is designed for the paper and a *proceeding* document for the volume. The journals are also stored in the proceeding collection with key value beginning with journal. We have 25593 proceedings and journal documents. Journal articles are also stored in the inproceeding collection.

The relationship between *papers* and *proceeding* is presented in Table 5 case g) from 3.1.2. Using Algorithm 1 we can see that nesting is possible.

We import the DBLP XML data in MongoDB in flat style first. In case of flat representation the *key* and *crossref* fields were used in order to map the one to many relationship between *proceeding* and the *papers* published in it, see Example 4. Then we constructed the nested representation in MongoDB using as input the flat MongoDB data. We used indexes for *crossref* field of *inproceeding* collection in flat style data to improve the retrieval of *papers* for one *proceeding*.

Example 4. Consider the flat structure: one conference *proceeding* and two *inproceeding* documents:

```
PROCEEDING:
{   "_id" : ObjectId("54d61a311ba3b50d0c1f5a20"),
    "key" : "conf/cla/2007",
    "editor" : ["Peter W. Eklund", "Jean Diatta", "Michel Liquiere"],
    "title" : "Proceedings of the Fifth International Conference on Concept Lattices
     and Their Applications, CLA 2007, Montpellier, France, October 24-26, 2007",
    "booktitle" : "CLA",
    "publisher" : "CEUR-WS.org",
    "volume" : "331",
    "year" : "2008",
    "series" : "CEUR Workshop Proceedings",
    "url" : "db/conf/cla/cla2007.html" }

INPROCEEDING:
{   "_id" : ObjectId("54d61a311ba3b50d0c1f59e4"),
    "author" : ["Radim Belohlvek", "Bernard De Baets", "Jan Outrata", "Vilm
                          Vychodil"],
    "title" : "Inducing Decision Trees via Concept Lattices.",
    "year" : "2007",
    "ee" : "http://ceur-ws.org/Vol-331/Belohlavek3.pdf",
    "crossref" : "conf/cla/2007",
    "url" : "db/conf/cla/cla2007.html#BelohlavekBOV07" }
{   "_id" : ObjectId("54d61a311ba3b50d0c1f59f4"),
    "author" : ["Laszlo Szathmary", "Amedeo Napoli", "Sergei O. Kuznetsov"],
    "title" : "ZART: A Multifunctional Itemset Mining Algorithm.",
    "year" : "2007",
    "ee" : "http://ceur-ws.org/Vol-331/Szathmary.pdf",
    "crossref" : "conf/cla/2007",
    "url" : "db/conf/cla/cla2007.html#SzathmaryNK07" }
```

Example 5. Consider the nested structure: two *inproceeding* documents nested in a conference *proceeding*:

```
NESTEDPROCEEDING:
{   "_id" : ObjectId("54d61a311ba3b50d0c1f5a20"),
    "key" : "conf/cla/2007",
    "editor" : ["Peter W. Eklund", "Jean Diatta", "Michel Liquiere"],
    "title" : "Proceedings of the Fifth International Conference on Concept Lattices
```

```
 and Their Applications, CLA 2007, Montpellier, France, October 24-26, 2007",
"booktitle" : "CLA",
"publisher" : "CEUR-WS.org",
"volume" : "331",
"year" : "2008",
"series" : "CEUR Workshop Proceedings",
"url" : "db/conf/cla/cla2007.html",
"inproceedings" :
  [ { "author" : ["Radim Belohlvek", "Bernard De Baets", "Jan Outrata", "Vilm
                  Vychodil"],
      "title" : "Inducing Decision Trees via Concept Lattices.",
      "year" : "2007",
      "ee" : "http://ceur-ws.org/Vol-331/Belohlavek3.pdf",
      "url" : "db/conf/cla/cla2007.html#BelohlavekBOV07" },
    { "author" : ["Laszlo Szathmary", "Amedeo Napoli", "Sergei O. Kuznetsov"],
      "title" : "ZART: A Multifunctional Itemset Mining Algorithm.",
      "year" : "2007",
      "ee" : "http://ceur-ws.org/Vol-331/Szathmary.pdf",
      "url" : "db/conf/cla/cla2007.html#SzathmaryNK07" }] }
```

3.2.1 Testing the Queries

We test ten queries for a single server MongoDB configuration and the same queries for three servers. In the case of three servers the queries were executed by one and three different users. The queries were executed with different parameters in different order more times. In the graphic representation we consider the average of the execution times.

The first five queries do not involve both *proceeding* and *inproceeding* type documents, only one of them. In the last five queries both - *proceeding* and *papers* needs to be accessed. In the case of nested design, the paper documents are embedded in the containing proceeding document. We create index on *crossref* field of *inproceeding* collection in case of flat representation to improve the 'join' operation, which has to be solved programmatically.

The queries were:

Query 1: Select the conference papers and journal articles for one author.

Query 2: Select the journal articles written by author1 and written by author2 in a given year or articles written by author1 and not written by author3 in a given year.

Query 3: Select the conference papers and journal articles in 3 given year.

Query 4: Select conference proceeding or journal information given a *booktitle* value. The *booktitle* is the same for a series of conferences which are held every year or a journal which appears more times in a year.

Query 5: The same as Query 4 complemented with a condition excluding a given year.

Query 6: Select conference or journal from proceedings collection given a *booktitle* value and a year value, together with papers from the selected proceeding.

Query 7: Given a paper title finds the paper and the proceeding where the paper was published.

Query 8: Given a part of a paper title finds the papers and the proceedings where the papers were published.

Query 9: Given a part of a proceeding title finds the corresponding proceedings and the papers published in these proceedings.

Query 10: Select journal articles of a given author in a given journal, in a given volume, for a given year and the corresponding journal.

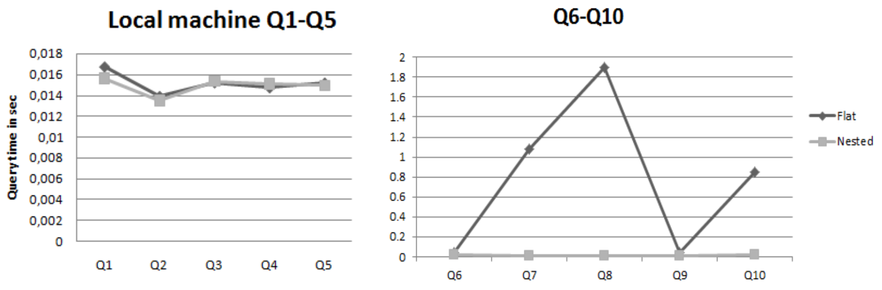


Figure 3
Execution time of queries on a local machine

The execution time for every query in nested design mode of data is between 0.01 and 0.02 sec in every case, using local machine or 3 servers. The execution time in flat structure of data for queries 6-10 is between 0.04 and 1.89 sec. Queries 1-5 are executed nearly at the same time in embedded and non-embedding cases.

In Figure 3 we can see the average execution time on a local machine for the ten queries in flat (Example 4) and nested (Example 5) structure of the data. The experiments for queries 6-10 show that the nested design mode of data is more suitable. The execution of the first five queries on one machine does not depend on the structure of the data.

Figure 4 presents the results of query execution on three servers with one user. In this case, the results are nearly the same, for queries 1-5 the structure of the data is not relevant, but for queries 6-10 the nested design of the data is more appropriate.

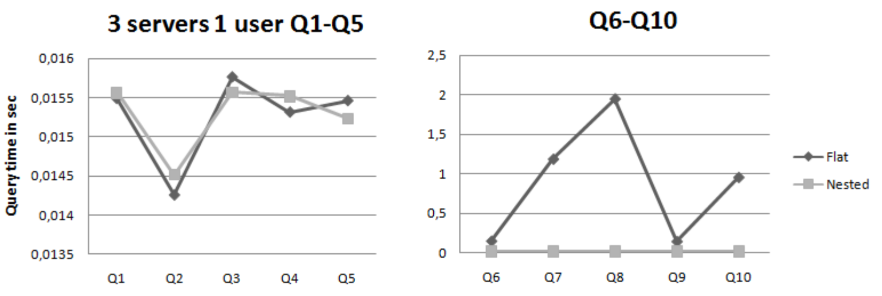


Figure 4
Execution time of queries on three servers one user

Figure 5 illustrates the execution time of the same queries on three servers by three different users concurrently. The results are nearly the same, nested structure of data is superior in this case as well.

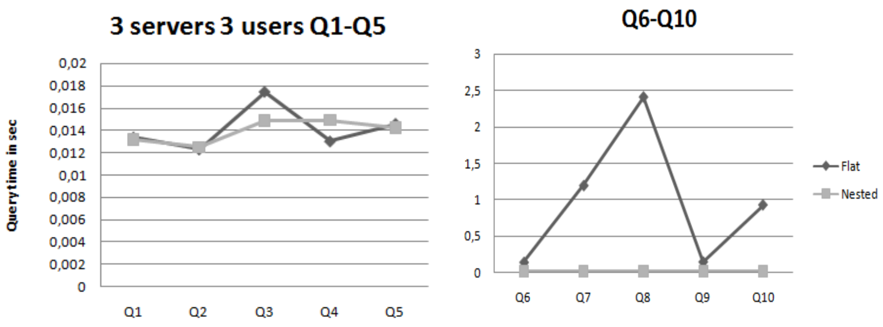


Figure 5
Execution time of queries on three servers with three users

Figure 6 presents the execution times for the three different architectures in the case of flat data structure. The differences are not relevant between the local machine and distributed architectures.

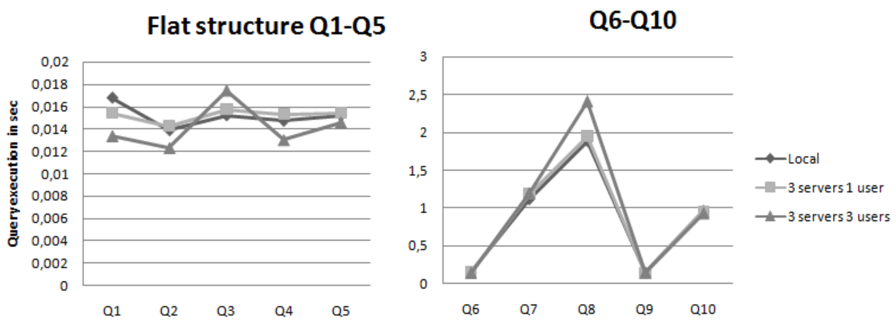


Figure 6
Execution time of queries for flat data structure

Figure 7 compares the query execution time for the nested structure of the data on different system architecture.

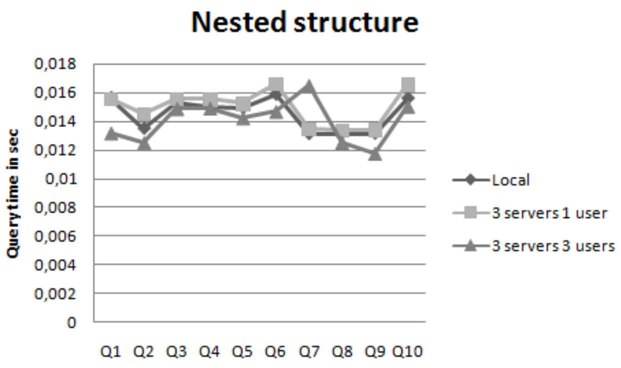


Figure 7
Execution time of queries for nested data structure

Our method of using RCA helps us to decide if nesting is feasible or not. We tested all queries in flat and nested versions and the results show that if we choose the design method, which is suggested by our algorithm, the execution time will be more efficient. Results of experiments using our method have proven that decisions affecting the modeling of data can affect application performance and database capacity.

Conclusions

In this paper we have proven that the more knowledge we have concerning a target domain, the better that certain tools can support the domain's analyses. Decisions that affect how we model data, can affect application performance and database capacity. We have covered the basics of data modeling for document based data stores. The three basic types of relations were discussed and illustrated with the help of examples.

We used FCA methods to visually analyze the schema of large-scale data. We proposed an RCA based approach to document based NoSQL database design. The possibility of nested scheme design can be read from the lattices. Results of experiments using our method have validated the feasibility of our approach.

References

[1] V. Abramova, J. Bernardino, NoSQL Databases: MongoDB vs Cassandra, Proceedings of the International C* Conference on Computer Science and Software Engineering, ACM, 2013, pp. 14-22

[2] M. Arenas, L. Libkin, A Normal Form for XML Documents. TODS 29(1), 2004, pp. 195-232

[3] S. Andrews, C. Orphanides, Analysis of Large Data Sets using Formal Concept Lattices, CLA 2010, pp. 104-115

[4] R. Cattell, Scalable SQL and NoSQL Data Stores. ACM SIGMOD Record 39.4, 2011, pp. 12-27

[5] K. T. Janosi Rancz, V. Varga, XML Schema Refinement Through Formal Concept Analysis, Studia Univ. “Babeş-Bolyai” Cluj-Napoca, Informatica, vol. LVII, No. 3, 2012, pp. 49-64

[6] M. Ley, DBLP Computer Science Bibliography. http://dblp.uni-trier.de/

[7] M. Ley, DBLP — Some Lessons Learned, Proc. VLDB Endowment, Vol. 2, Nr. 2, 2009, pp. 1493-1500

[8] Ecma International. The JSON Data Interchange Format, 2013, www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

[9] R. Elmasri, S.B. Navathe: Fundamentals of Database Systems, Addison Wesley (2010)

[10] M. Franceschet, D. Gubiani, A. Montanari, C. Piazza: A Graph-Theoretic Approach to Map Conceptual Designs to XML Schemas, ACM Transactions on Database Systems, 2013, Vol. 38, pp. 6-44

[11] B. Ganter, R. Wille, Formal Concept Analysis: Mathematical Foundations, Springer, 1999

[12] D. V Gnatyshak, D. I. Ignatov, S. O. Kuznetsov, L. Nourin, A One-Pass Triclustering Approach: Is There any Room for Big Data?, Proceedings of the 11th International Conference on CLA, 2014, pp. 231-242

[13] C. Yu, H. V. Jagadish, XML schema refinement through redundancy detection and normalization. VLDB J. 17(2), 2008, pp. 203-223

[14] K. T. Janosi-Rancz, V. Varga, T. Nagy, Detecting XML Functional Dependencies through Formal Concept Analysis, ADBIS 2010, Novi Sad, Serbia, LNCS 6295, pp. 595-598

[15] M. Klettke , U. Störl, S. Scherzinger: Schema Extraction and Structural Outlier Detection for JSON-based NoSQL Data Stores, 16th Conference on Database Systems for Business, Technology, and Web (BTW), 2015

[16] B. Ganter, C. Meschke, A Formal Concept Analysis Approach to Rough Data Tables. Trans Rough Sets 2011, 14:37–61

[17] S. O. Kuznetsov, J. Poelmans, Knowledge Representation and Processing with Formal Concept Analysis, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, Vol. 3, Issue 3, pp. 200-215, 2013

[18] M. Rouane-Hacene, M. Huchard, A. Napoli, P. Valtchev, A Proposal for Combining Formal Concept Analysis and Description Logics for Mining Relational Data, ICFCA'07, France. Springer, LNAI 4390, pp. 51-65

[19] M. Rouane-Hacene, M. Huchard, A. Napoli, P. Valtchev, Relational Concept Analysis: Mining Concept Lattices from Multi-Relational Data, Annals of Mathematics and Artificial Intelligence 67, 1, 2013, pp. 81-108

[20] S. Tiwari, Professional NoSQL, O'Reilly, 2013

[21] S. Václav, H. Zdenek, A. Ajith, Understanding Social Networks Using Formal Concept Analysis, Proc. WI-IAT '08, Volume 03, pp. 390-393

[22] V. Varga, Ch. Sacarea, An FCA Driven Analysis of Mapping Conceptual Designs to XML Schemas, Studia Univ. "Babeş-Bolyai" Cluj-Napoca, Informatica, Vol. LIX, No. 1, 2014, pp. 46-57

[23] R. Belohlavek, Fuzzy Galois Connections. Math Logic Q 1999, 45:497–504

[24] B. Ganter, S. O. Kuznetsov, Pattern Structures and their Projections. Proc. of 9th ICCS'01. LNAI, 2120; July 30-August 3, 2001; Stanford University, CA. Berlin, Heidelberg: Springer 2001, 129-142

[25] D. Gnatyshak, D. I. Ignatov, S. O. Kuznetsov, L. Lourine: A One-pass Triclustering Approach: Is There any Room for Big Data?, CLA 2014, 231-242

[26] F. Lehmann, R. Wille, A Triadic Approach to Formal Concept Analysis. ICCS; August 14-18; Santa Cruz, CA. Berlin, Heidelberg: Springer; 1995, 32-43