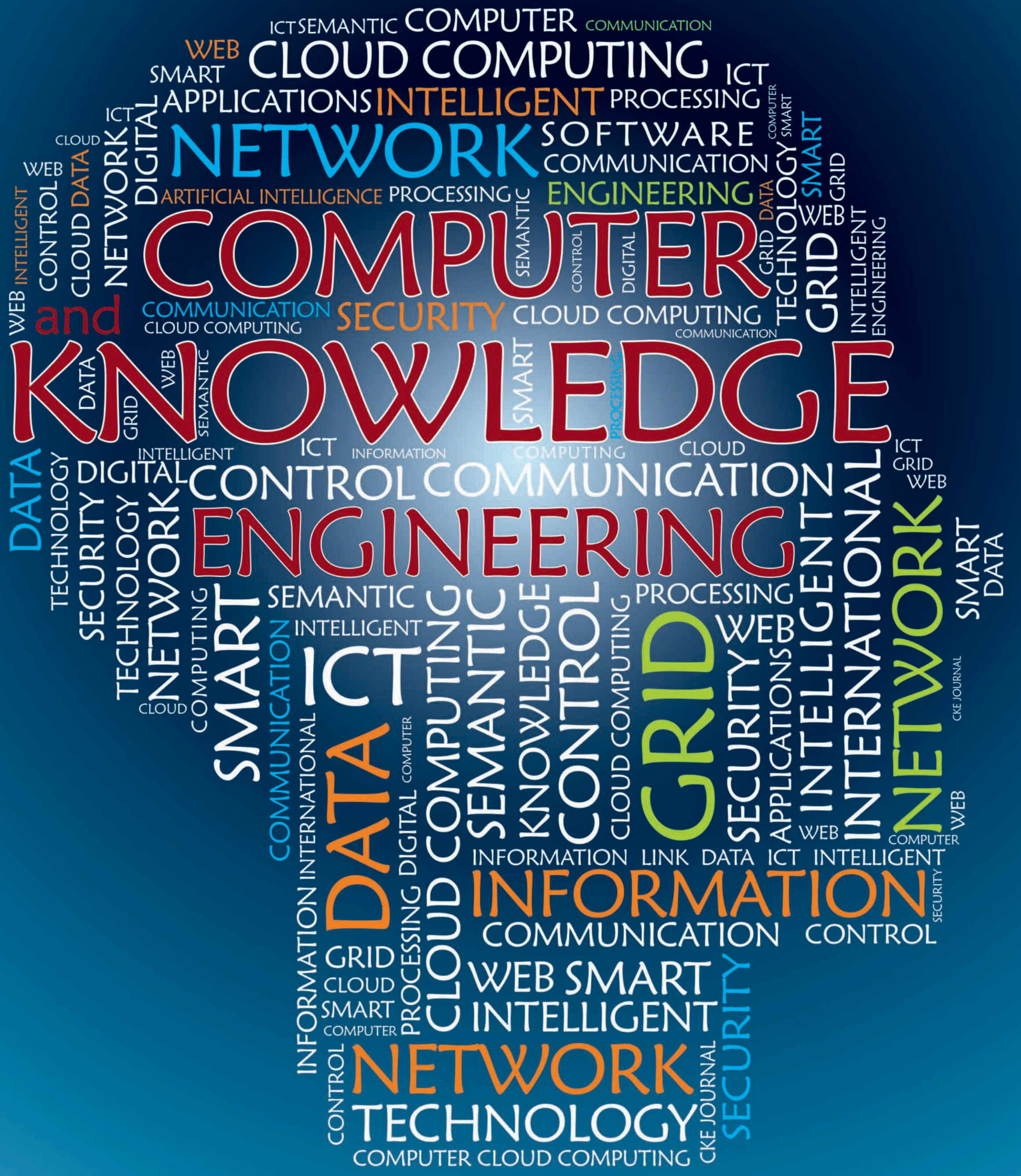
FACULTY OF ENGINEERING

Journal of Computer and Knowledge Engineering

Vol. I, No.I. 2017, ISSN : 5238-5453



Dept of Computer Engineering, Ferdowsi University of Mashhad

Serial No. 1

Editor's Letter

I am honored to present to you the very first issue of the journal of **Computer and Knowledge Engineering (CKE).** With the high coopetition in this field and the hard work needed to compose a set of high quality papers, we couldn't be more excited to have made it to this point. Please take some time to get to know our editorial board members, the focus and scope of the journal and the high quality papers that are selected for this issue.

CKE is founded by the Department of Computer Engineering, Faculty of Engineering, Ferdowsi University of Mashhad. It will be published twice a year and the journal materials will be available both on-line and in printed forms.

Submission of manuscripts not published before and not being under review by any other journals are encouraged. The authors are welcomed to upload their articles in either pdf or doc file format using the journal's website, i.e. http://cke.um.ac.ir/ . All submitted papers are single-blind peer reviewed by at least two different reviewers.

The journal is very vigilant against any kind of plagiarism and all papers will be checked by similarity finding software.

The final version of all accepted papers will be formatted by the thoughtful people of the faculty. This is a small gesture of help by the journal to let you focus on what you are doing best, that is research and development.

We appreciate your support and are so happy to have you as a reader of the journal of Computer and knowledge Engineering.

With warmest thanks,

Professor Mahmoud Naghibzadeh
Editor-in-chief

CONTENTS

Journal of Computer and Knowledge Engineering, Vol. 1, No. 1
DOI: 10.22067/cke.v1i2.56021

An Energy Efficient Data Dissemination Scheme for Distributed Storage in the Internet of Things

Mojtaba Rajabi, Gholamhossein Ekbatanifard[✦]

Abstract. In the emerging field of the Internet of Things (IoT), Wireless Sensor Networks (WSNs) play a key role in sensing and collecting data from the surrounding environment. In the deployment of large-scale monitoring systems in remote areas, when there is not a permanent connection with the Internet, WSNs are called upon for replication and distributed storage techniques that increase the amount of data storage within the WSN and reduce the probability of data loss. Unlike conventional network data storage, WSN-based distributed storage is constrained by the limited resources of the sensor nodes. In this paper, we propose a low-complexity distributed data replication mechanism to increase the capacity of WSN-based distributed storage, optimizing communication and decreasing energy usage. As the simulation results show, the proposed method has been able to attain acceptable responses and prolong network lifetime.

Keywords: The Internet of Things, Distributed Storage, Energy Efficient.

1. Introduction

The Internet of Things is a new concept that has emerged in recent decades. The concept is based on the theory that useful things have a permanent IP connection to the internet. The coverage of things in IoT encompasses different systems from radio frequency identification (RFID), machine to machine (M2M) to WSNs. Various IoT-based business applications have been presented, e.g. smart clusters with smart infrastructures [1]. WSNs for IoT monitoring systems include free nodes (supervision free) for sensing the environment and usually a sink node for gathering data and a gateway to the internet. Connections among sensor nodes and sink node are not instant namely in isolated WSNs in which sink node is not always present. In addition, while real-time gathering is not required, data storage and transmission units can be used to reduce radio transmissions and increase sensor lifetime [1].

Regarding the applications of WSNs in different domains, the considered system is required to have distributed storage capacity and data redundancy for long term usage in remote areas for distributed data storage. This capability will be attained through distribution and storage of multiple replications of data in a WSN. This redundancy leads to subsequent communicational and system storage overheads. Thus, in order to present an efficiently distributed storage algorithm with data replication scheme, a balance among node capacity, communicational optimization, and energy usage is required. In this paper, an IoT-based distributed storage scheme with data replication will be presented which aims at boosting storage capacity and optimizing communication and minimizing energy usage of the whole system.

Since there is no persistent communication between sensor and sink nodes in wireless sensor networks, nodes are required to store data and provide it to sink node when needed. Storing data in a node will be subject to missing, e.g. certain nodes may leave the network for certain reasons such as battery dying or physical destruction like a bomb explosion. In order to prevent similar cases, sensor nodes must be distributed in the network, i.e. sensors with a low data capacity of memory are required to provide their memories with other sensors as support memory.

The main purpose of this work is to design a system for distributing data, data replication, in different nodes which aim at balancing storage space usage, system stability, energy usage and communicational overheads. In the proposed method, by considering nodes energy in case of an error occurrence in multiple neighbors, network storage capacity, the required time to attain this capacity, data loss, and system stability will be calculated. Time intervals of network runtime in the classified algorithm divide system runtime into specific rounds and equal *T*- intervals. The division is such that in each interval, the selected set will be activated in the amount of *T* and other nodes will remain inactive and idle accordingly. Calculation of the *T* will vary according to the classification type and nodes remaining energy and estimated the lifetime of the network. Its duration can be evaluated according to the network requirements and physical parameters of the used sensor. In the presented scenario, the sensor nodes have been set randomly in the network. Activity timetable of sensor nodes must be such that it guarantees the following requirements:

1. Each active node which is available in any selected *Cj* sets must be connected to sink in each round and must at least have one path to sink for transmitting data.
2. Normal nodes are equipped with *E* initial energy, R_c communicational range and R_s sensory range ($R_c >= R_s$).

The rest of the paper is structured as follows: The network and energy model used in the paper are presented in Section 2. Related works are reviewed in Section 3. In Section 4 the proposed method is given. Section 5 expresses simulation results and finally, Section 6 concludes the paper.

2. Network and Energy model

In our network model, each node has an individual ID. Nodes are aware of their position. The considered network

Manuscript received May 24, 2016; revised October 17, 2016; accepted November 26, 2016.
Mojtaba Rajabi, Department of Computer Engineering, Rasht Branch, Islamic Azad University, Rasht, Guilan, Iran.
Gholamhossein Ekbatanifard, Department of Computer Engineering, Lahijan Branch, Islamic Azad University, Lahijan, Guilan, Iran.
[✦]The corresponding author's e-mail is: ekbatanifard@liau.ac.ir

is a combination of manager nodes, sensor nodes, and targets. Synchronizing manager nodes takes place via central station and then synchronizing other nodes take place via manager nodes, based on distance decrease or an increase of nodes they are capable of transmitting and adjusting sender power, moreover, they can distinguish the distance according to received signal strength. The number of targets with the constant position will be in the covered area.

The energy model which consists of sending and receiving l bit data has been considered according to LEACH model [2]. Given a distance from sender to receiver, if the distance from the sender to the receiver (*d*) is beyond d0, multiple routes method (path coefficient equals to 4), otherwise open space model is used (route coefficient equals to 2). The following relation [2] will be held for transmitting l bits to a distance *d*:

$$ETX(l, d) = E_{TX\text{-}elec}(l) + E_{TX\text{-}amp}(l, d) = \begin{cases} l \times E_{elec} + l \times \varepsilon_{fs} \times d^2 & d < d_0 \\ l \times E_{elec} + l \times \varepsilon_{mp} \times d^4 & d \geq d_0 \end{cases} \quad (1)$$

where $E_{TX\text{-}elec}(l)$ is the energy that radio dissipates to run the transmitter, and $E_{TX\text{-}amp}(l, d)$ represents the power amplification triggering the energy to send l bits. In this regard, E_{elec} equals the required energy for activating electronic circuits and ε_{fs} and ε_{mp} denote power amplification activating energy for multiple routes and open space respectively. A more general scheme of this relation can be stated with a constant *p* and *q* coefficients as relation 2.

$$ETX(l, d) = p + q \times d^n \quad (2)$$

Energy consumption for receiving *l* bit data on the receiver side will be in the form of relation 3.

$$ERX(l, d) = E_{RX\text{-}elec}(l) = l \times E_{elec} = p \quad (3)$$

3. Related Works

In recent years, multiple patterns regarding data distribution and replication in WSNs have been presented. In distributed storage patterns, WSN nodes cooperate in distributing data in the network. Generally, two certain methods can be presented: Data-Centric Storage (DCS) and Wholly Distributed Storage (WCS). In DCS the related data are stored by their names in a node and the queries with a specific name can be sent directly to its related stored data, while no flooding dissemination is needed [3].In [4] a load balanced distribution method has been presented that stores data in order to prevent data loss in sensory crowded regions.

The presented method in [5] stores data in terms of spatial and temporal similarities in order to reduce overhead and query delay. Since the said method is node-cooperative based, it is not WDS, since specific nodes store the content of other nodes. In WDS, nodes are equal in sensing and storing. Data are first stored locally, then immediately, by filling their local memories, they devolve new data storages to other nodes. The initial struggle at this point is to create data frames. In [6] a distributed mechanism with periodical data recycling scheme has been presented and cost model is used in order to measure energy usage and it shows how choosing the accurate storage nodes to help optimizing system capacity and mitigating transmission costs. In this method, a network with tree topology is assumed in which each sensor node knows of the return path to the sink node.

In [7] authors discussed energy usage and presented an energy efficient distribution data pattern according to data dissemination from low to high power nodes. In [8] a load balancing method has been presented and more focus is being put on redistribution of data, while sensor node storage space goes beyond the threshold. In the presented method, each node has a local memory table which is used to store neighbor nodes memory. In order to transmit data from a crowded environment to a less crowded one and sending stored data to the sink node, a mobile node is used. Data replication strategies are presented to solve node failure problem. The purpose of this strategy is to replicate data in other nodes to increase network flexibility. ProFlex method [9] is a storage protocol of distributed data from limited to strong nodes. One advantage of this method is its high communicative range and using long links to improve the distribution and data replication in node failure risk model.

In the presented method in [10], the replication node is selected according to certain parameters, e.g. connectivity, access memory and the remainder of node energy. In TinyDSM method [11], a reactive replication method is presented that distributes replicates randomly in the received replication area according to the number and replication density.

4. The Proposed Method

In order to apply and investigate the proposed method, it is assumed that sensor network nodes are continually collecting data. Data are collected periodically by the sink and removed from their memories. This periodical recycling allows limited memory usage, but data recycling is not the focus in this paper. In order to prevent data loss due to node failure or memory limitation, nodes operate in the following method. Based on a greedy distribution storage scheme, each sensor node reports its memory condition to other nodes. Each memory condition message consists of the following measures, which are related to sender node: (1) sensor node ID, (2) recent access memory space, (3) sensory rate and (4) an ordinal number that introduces the message. Each node keeps a local memory table which records the latest position of the reception memory of neighboring nodes.

Local memory table includes an entry for each neighbor that indicates the latest access memory space and corresponding awareness time. Each node updates its memory immediately after receiving news from a neighbor and prevents replicated or expired news by utilizing ordinal numbers. An individual node stores its *best* neighbors in case of memory table space expiration that commonly occurs in dense networks with above one-step neighbors. The *best* neighbor is also generous since it offers its memory to other nodes. If no *best* neighbor is found for a node, the node must delete formerly available data in its memory. In the proposed method, the size of the memory table is assumed to remain constant since dimensions may increase in dense networks. But in the case of finding the *best* neighbor, one element will be inserted in the table. Each sensor node makes an update decision immediately after receiving update data from neighbor nodes.

In this method, maximum *R* versions (instances) of each data will be stored in sensor's memory and each sensor node keeps at least one version of similar data. By creating a data item, sensor node can keep one version of each in its memory and R-1 will keep other versions in the neighbor node's memory $\left\{1,\ldots,V_1^{(i)}\right\}$. The best neighbor for node *i*, the generous one, as indicated as $D_r^{(i)}$, will be achieved based on relation 4:

$$D_r^{(i)} = arg\,max_{j\in\{1,\ldots,V_1^{(i)}\}} \frac{B_j(t_j)}{t-t_j} \tag{4}$$

where $t_j<t$, memory space $B_j(t_j)$ of node *j* has been received by node i and t_j equal the latest updating of neighbor's memory table.

In the decision-making process, nodes with the maximum empty memory and newest entry in the table will be chosen. In the absence of a proper node, i.e. $B_j\left(t_j\right) = 0, \forall j \in \{1,\ldots,V_1^{(i)}\}$ redundant data will be discarded. After receiving data, it will be stored in the buffer and based on relation 5 in *r*th replication, a generous neighbor will be chosen.

$$D_r^{(i)} = arg\,max_{j\in\left\{1,\ldots,V_1^{(i)}\right\}\setminus S^{r-1}} \frac{B_j(t_j)}{t-t_j} \tag{5}$$

where S^{r-1} is a set of *r*-1 generous nodes of the previous version. After choosing the generous node, i.e. *r*th node, one version of the data will be sent for this node and one unit will be subtracted from the whole version. Replication process will continue either to save the latest version or not receive any generous node. If no other generous node is received, the number of real stored versions will be less than *R*.

The only effective element in determining neighbor nodes for replicating wireless sensor data is D generous parameter. Prioritizing neighbor nodes will be based on the amount of their empty memory. The right option for data replication will be made if a node has sufficient empty memory and its level of energy is appropriate. In the proposed method, an integration of memory space and remaining energy will be utilized according to the three following parameters. Time of data updating, data lateness in the network, is the most important factor in decision making. Thus, our intention of *time* is the difference between present time and the latest updating time which is attained and normalized through the following relationship:

$$\Delta t_{total} = \sum_i(\Delta t)_i = \sum_i(t-t_i)$$
$$\widehat{\Delta t}_i = \frac{(\Delta t)_i}{\Delta t_{total}} = \frac{(\Delta t)_i}{\sum_i(t-t_i)} \tag{6}$$

where $(\Delta t)_i$ is the difference between current and the latest update time of *i*th neighbor. Δt_{total} is the sum of temporal (time) differences between the current and the latest update time for all neighbors. $\widehat{\Delta t}_i$ is the normalized time difference. The second parameter is the rate of free memory space of each neighbor node, in which free memory of all neighbor nodes are calculated and their reverse sum determines the normalization coefficient of memory for prioritizing. Thus, according to relation 7, the amount of normal empty memory space for an *i*th node will be attained.

$$B_{total} = \sum_i B_i(t_i)$$
$$\hat{b}_i = \frac{B(t_i)}{B_{total}} = \frac{B(t_i)}{\sum_i B_i(t_i)} \tag{7}$$

where $B_i(t_i)$ is the rate of empty memory of ith node at the latest update time of data from neighbor nodes, B_{total} is the sum of empty memories of all neighbor nodes and $\hat{b}_i$ is the rate of normal empty memory of the *i*th node at the latest update time. The third considered parameter is *energy*, which is the sum of normalized energy of all neighbor nodes as the choice factor, which is normalized according to relation 8:

$$E_{total} = \sum_i E_i(t_i)$$
$$\hat{e}_i = \frac{E(t_i)}{E_{total}} = \frac{E(t_i)}{\sum_i E_i(t_i)} \tag{8}$$

The final decision parameter has been introduced as the weighted sum of the three parameters, which is demonstrated as the following formula:

$$D_i(t) = w_t\widehat{\Delta t}_i + w_b\hat{b}_i + w_e\hat{e}_i \tag{9}$$

The last considered parameter is the impact factor (coefficient) which is computed as below:

$$D_i(t) = \frac{1}{3}(\widehat{\Delta t}_i + \hat{b}_i + \hat{e}_i) \tag{10}$$

The $\frac{1}{3}$ the coefficient is chosen such that generous coefficient is in [0-1] interval.

5. Simulation results

The common viewpoint in this study assumes that the latest data are more valuable regardless of the source from which they are produced. Intuitionally, it seems acceptable that newer sensors of the environment are more valuable. It is worth noting that during the designing phase, the nodes intermittence connection was taken into consideration so that sensors' memory data does not overflow. In such cases, data replications require more memory.

In this section, the results of the proposed method will be investigated and a comparison will be made based on the condition in which this method is not adopted too. In order to present the outcome results of the proposed method, Matlab (version 2013) was used. For simulating network parameters, table 1 parameters are used for the suggested system. For simulating purpose, network nodes were changed from 10 to 100, to investigate the rate of outputs in different conditions.

The memory of each node is an important parameter in data network storage. Investigating memory nodes and homogenous usage of reliable positions in the network are considered vital in data storage. Figure 1 represents nodes' memory space in comparison with the simulation time in different modes.

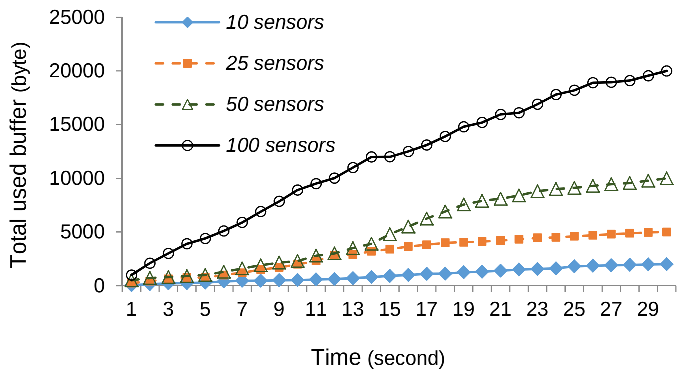


Fig. 1. Total used memory for the proposed method for different number of sensor nodes

Table 1. The simulation parameters

Symbol	**Amount**	**Unit**	**Description**
N	100-10	Scalar	Number of nodes
A	200*200	M2	Area surface
D	80	M	Transmission range
$V_1^{(i)}$	-	Scalar	Number of one-step neighbors
B_i	250	Scalar	Size of nth buffer $i\epsilon\{1,\dots,N\}$
$T_{sense,i}$	1	S	Node nth sensing interval $i\epsilon\{1,\dots,N\}$
$_{sense,i}$	10	S^{-1}	Node nth sensing rate $i\epsilon\{1,\dots,N\}$
P_t	0.01 to0.5	mw	Node transmission power
T_{adv}	10	S	Memory notification period
R	3	Scalar	Maximum number of replication in each sensory unit
T	10	S	Data recovery period

As the figure shows, buffer usage procedure is depicted according to time passage. It is noted that balanced buffer capacity of nodes will be used by time passage which is observable in this network with different numbers of nodes. Investigating the balance feature reveals that it is one of the advantages of the proposed method in which energy parameter has been involved.

The other required parameters in examining the efficiency of data storage are investigating buffer condition of active nodes. Figure 2 shows buffer condition for each node and its neighbor's data storage at the end of simulation time which is distinguished by two colors. As it is indicated, the proposed method has been able to create a good balance between node data storage and neighbors' data storage. It can be observed that in most cases this balance is distributed among nodes.

Another criterion for investigating and examining an algorithm or protocol is how energy is used and how a balance is made among network nodes. Although the main proposal and initiative of this study have been based on the mentioned parameter, examining this section seems very important. As figure 3 shows, the remaining energy of each applied network is illustrated. These cases have been investigated and evaluated for modes in which the number of nodes has experienced change. As it is shown in the diagrams, the remaining energy of nodes is indicative of the balanced usage of network nodes which finally leads to a prolonged network lifetime.

Considering energy not only leads to heightening reliability of data storage in neighbor nodes but has been able to promote the efficiency of the network. It is such that in the previous state, as figure 4 shows, energy usage was heterogeneous whilst energy was not considered.

In certain cases, the network attenuation in high energy level is seen while no efficient usage of nodes is being discovered.

The figure indicates that increasing the number of sensor nodes causes an increase in the supportive sensors. Therefore the missing data could be extracted by the larger number of nodes. In this regard, increasing the number of nodes is inversely related to the amount of missing data.

In figure 5, because the energy parameter is not considered in selecting nodes, it is possible that the recipient node is dead or has died immediately after receiving the data before delivering it to the sink. This issue is considered in the proposed method and consequently better results are obtained at higher nodes scenarios.

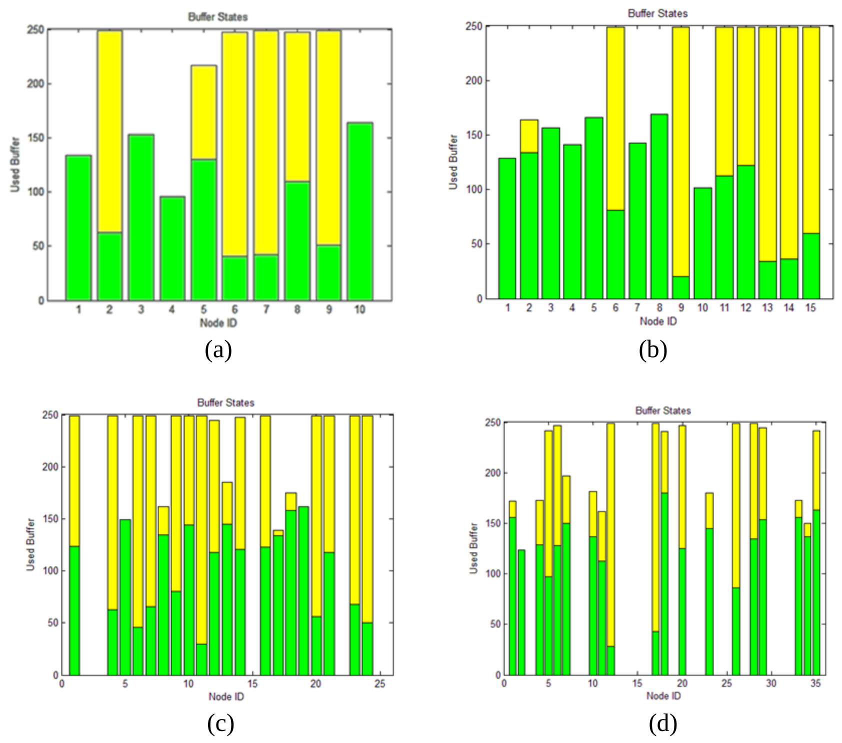


Fig. 2. The condition of each node’s buffer at the end of simulation for the proposed method. (Green: occupied by sensor data, yellow: occupied by neighbors data) (a) For each 10 sensor nodes (b) for 15 nodes (c) for 25 nodes (d) for 35 nodes

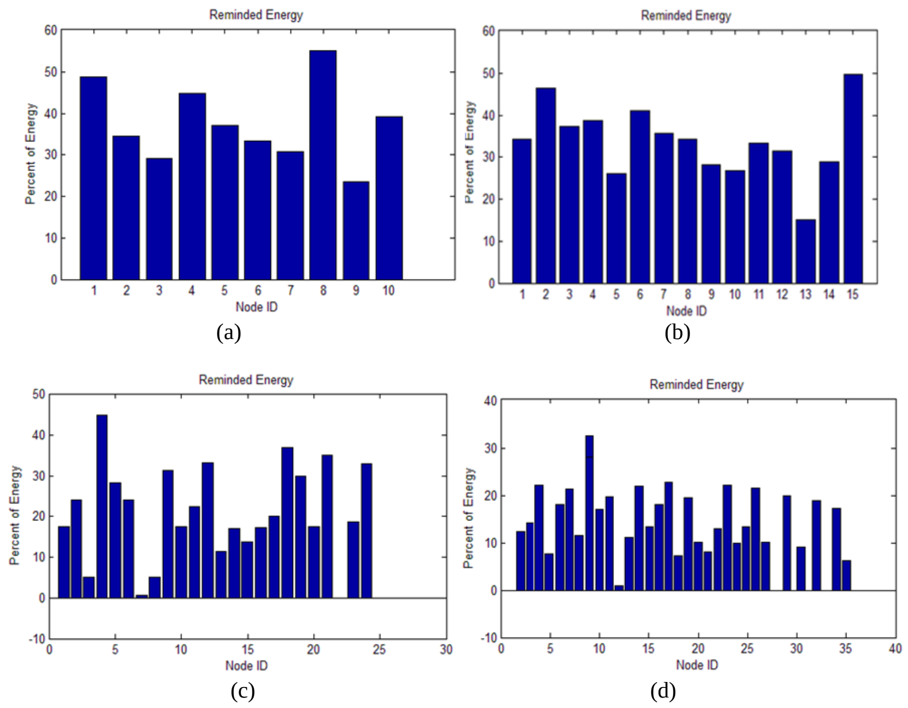


Fig. 3. The remained energy of sensor nodes at the end of simulation for the proposed method. For (a) 10 nodes (b) 15 nodes (c) 25 nodes (d) 35 nodes

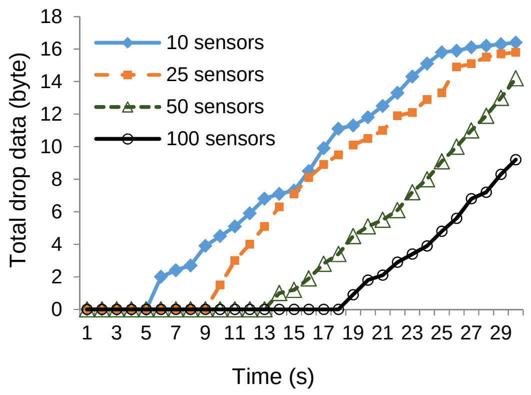


Fig. 4. Sum of the waste data during simulation in the proposed method.

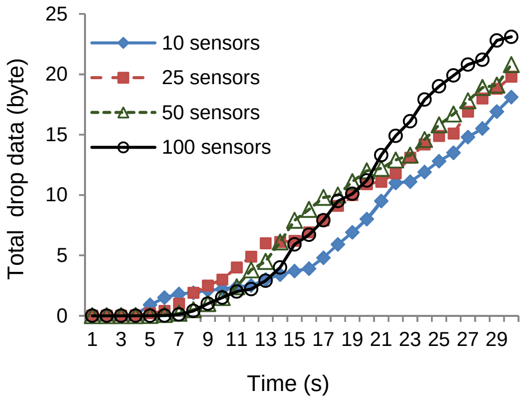


Fig. 5. Sum of the waste data during simulation in the related work [1].

6. Conclusion

In this paper, the problem of distributing redundant data in monitoring IoT–based systems has been discussed and a method with low complexity and overhead for distribution and data replication has been proposed. The simulation of the proposed method has been simulated with Matlab software. The goal is to create a balance between storage space, system stability, and energy consumption while communicational efficiency is high. By comparing and investigating the proposed method, the relative improvement in energy usage, lifetime and a balance in data storage in neighbor nodes have been noticed, while other parameters, e.g. efficiency and stability has been equal to related works. In certain cases, relative improvements have been observed which are indicative of the positive effect of the design and performance of the proposed method.

References

[1] P. Gonizzi, G. Ferrari, V. Gay, J. Leguay, "Data dissemination scheme for distributed storage for IoT observation systems at large scale", Information Fusion, Vol. 22, pp. 16-25, 2015.

[2] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocols for Wireless Microsensor Networks", Proceedings of the 33rd Hawaiian International Conference on Systems Science (HICSS), 2000.

[3] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, F. Yu, "Data centric storage in sensornets with ght, a geographic hash table", Mobile Networks and Applications, Vol. 8, pp. 427–442, 2003.

[4] M. Albano, S. Chessa, F. Nidito, S. Pelagatti, "Dealing with non-uniformity in data centric storage for wireless sensor networks", IEEE Transactions on Parallel and Distributed Systems, Vol. 22, pp. 1398–1406, 2011.

[5] H. Shen, L. Zhao, Z. Li, "A distributed spatial–temporal similarity data storage scheme in wireless sensor networks", IEEE Transactions on Mobile Computing, Vol. 10, pp. 982–996, 2011.

[6] A. Omotayo, M. Hammad, K. Barker, "A cost model for storing and retrieving data in wireless sensor networks", IEEE 23rd International Conference on Data Engineering Workshop (ICDE), pp. 29–38, 2007.

[7] M. Takahashi, B. Tang, N. Jaggi, "Energy-efficient data preservation in intermittently connected sensor networks", IEEE 30th Conference on Computer Communications Workshops (INFOCOM’11), pp. 590–595, 2011.

[8] L. Luo, C. Huang, T. Abdelzaher, J. Stankovic, "Envirostore: a cooperative storage system for

disconnected operation in sensor networks", IEEE International Conference on Computer Communications (INFOCOM’07), pp. 1802–1810, 2007.

[9] G. Maia, D.L. Guidoni, A.C. Viana, A.L. Aquino, R.A. Mini, A.A. Loureiro, "A distributed data storage protocol for heterogeneous wireless sensor networks with mobile sinks", Ad Hoc Networks, Vol. 11, Issue. 5, pp. 1588–1602, 2013.

[10] J. Neumann, N. Hoeller, C. Reinke, V. Linnemann, "Redundancy infrastructure for service-oriented wireless sensor networks", IEEE International Symposium on Network Computing and Applications (NCA),pp. 269 - 274, 2010.

[11] K. Piotrowski, P. Langendoerfer, S. Peter, "tinyDSM: A highly reliable cooperative data storage for wireless sensor networks", International Symposium on Collaborative Technologies and Systems (CTS’09), pp. 225–232, 2009.

Journal of Computer and Knowledge Engineering, Vol. 1, No. 1.
DOI: 10.22067/cke.v1i2.60404

Collusion-resistant Worker Selection in Social Crowdsensing Systems

Masood Niazi Torshiz[✦], Haleh Amintoosi

Abstract. The main idea behind social crowdsensing is to leverage social friends as crowdworkers to participate in crowdsensing tasks. A main challenge, however, is the identification and recruitment of well-suited workers. This becomes especially more challenging for large-scale online social networks with potential sparseness of the friendship network which may result in recruiting participants who are not in direct friendship relations with the requester. Such recruitment may increase the possibility of collusion among participants, thus threatening the application security and affecting data quality. In this paper, we propose a collusion-resistant worker selection method which aims to prevent the selection of colluders as suitable participants. For each participant who is considered to be selected as suitable, the proposed method is aimed to prevent any possible collusion. To do so, it determines whether the selection of a new participant may result in the formation of a colluding group among the selected participants. This has been achieved through leveraging the Frequent Itemset Mining technique and defining a set of collusion behavioral indicators. Simulation results demonstrate the efficacy of our proposed collusion prevention method in terms of selecting efficient collusion indicators and detecting the colluding groups.

Keywords: worker selection; collusion; data quality.

1.Introduction

1.1. Background

The widespread prevalence of mobile computing devices such as sensor-rich smartphones has propelled the emergence of a novel crowdsourcing [1] paradigm, known as mobile crowdsensing or participatory sensing [2]. In mobile crowdsensing, ordinary citizens volunteer to use their mobile phones for collecting sensor data from their nearby environment. The aim of such sensor data gathering methods include computing the aggregate statistics about a phenomenon, thus increasing the global awareness of the issues of interest. A plethora of applications have been recently proposed based on this revolutionary paradigm, ranging from personal health [3, 4] and prices of customer goods [5] to environmental monitoring such as road conditions [6] and noise pollution [7].

The involvement of people in the sensing process, however, brings about new challenges. Accepting to contribute to a task will inherently require that the worker devotes some time and effort towards it. Moreover, collecting and uploading the sensor data consumes the mobile phone battery and communication bandwidth. Most importantly, engaging in such crowdsourcing activities may lead to potential privacy threats such as the disclosure of home/work address or private conversations [8, 9, 10]. With all these in mind, a participant may be hesitant to contribute to a sensing task. This may result in a lack of adequate number of workers, which in turn may compromise the fidelity of the obtained information and ultimately render the application to be not very useful.

One potential solution to address this challenge is to leverage online social networks (constituting hundreds of millions of subscribers with various skills and expertise) as the underlying publish-subscribe infrastructure for crowdsensing applications [11, 12]. In a typical social crowdsensing system, social network members can act as service requesters and utilize social friends and friends-of-friends as crowdworkers to contribute to their tasks. A pertinent example of such a system is Jelly[1] which is built on top of existing social networks like Facebook[2] and Twitter[3]. When the users encounter something unusual, they can take a picture of the object, formulate a query and submit it to their social network. Another instantiation of the concept of social participatory sensing is found in [13] where Twitter is used as the underlying social network substrate. The authors proposed two mobile applications: (i)

Manuscript received November 21, 2016; accepted January 4, 2017.
Masood Niazi Torshiz, Department of Computer Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran.
Haleh Amintoosi, Department of Computer Engineering, Engineering Faculty, Ferdowsi University of Mashhad, Mashhad, Iran.
[✦]The corresponding author's e-mail is: niazi@mshdiau.ac.ir

[1] http://blog.jelly.co/post/72563498393/introducing-jelly
[2] http://facebook.com
3 http://twitter.com

a weather radar application in which, Twitter members send tweets indicating the weather condition and (ii) a noise-mapping application where members gather sound samples via their mobile phones and contribute the noise level via Twitter.

1.2. Problem Statement

One of the important challenges in the success of social crowdsensing is the selection of suitable participants as crowdworkers. Leveraging well-suited participants is bound to increase the quality of obtained contributions, since suitable participants have better knowledge and expertise relevant to the task requirements.

In the context of mobile crowdsensing, tasks are normally location-based (i.e. contributions should be collected from a specific place) and are to be completed within a specific time period. As such, the suitability of the participant for a campaign is typically related to the participant's geographical and temporal availability as well as the participant's reputation [14]. The geographical and temporal availability is extracted from collecting and analysing the time-stamped location traces of the participants. The reputation of the participant is measured based on the quality of the contributions of the participant in the past.

In social crowdsensing, the existence of public profile information of participants (who are social network members as well), and the social links between them adds new dimensions to the evaluation of a participant's suitability. Through public profile information, access to the participant's interests, expertise and domain specific knowledge is possible. Moreover, the participant's social reputability can be derived from his social relations and interactions. These valuable pieces of information can be used to identify well-suited participants, and hence, overcome the challenge of suitability.

Another important issue that should be considered when evaluating the suitability of participants is the likelihood of their involvement in collusive groups. A group of malicious participants might form a colluding group in such a way that they are recruited in preference to other potentially high-quality workers. The colluding group would then have the power to sway the outcome of the task in accordance with their agenda. So, it is important to identify potentially collusive members and prevent them from being selected as suitable participants.

In order to prevent collusion in mobile crowdsensing, a series of works [15, 16] utilise a trusted platform module (TPM) [17]. TPM is a micro-controller provided with each sensor device to attest the integrity of sensor readings. This local integrity checking makes the system resistant to collusion. However, TPM chips are yet to be widely adopted in mobile devices. In other research that eschews TPM, such as [18, 19], the collusion detection is achieved by leveraging reputation management systems and outlier detection algorithms. The aim is to identify and revoke the colluders by investigating their behaviour and assigning a low reputation score to them.

In the context of social crowdsensing, the existence of social ties between members facilitates the formation of colluding groups. Colluders can easily communicate via the social network communication facilities. They are also able to establish social communities by creating groups in the social network and manage collusive attacks by collaboratively contributing to a series of tasks. They can also easily share their corrupted contributions with other group members and hence propagate the bias. So, selecting the participants in such a way that the probability of collusion among the selected members is very low is important for achieving high quality contributions.

1.3 Contributions and Outline

In this paper, we propose a collusion-resistant worker selection method, which is aimed at preventing the selection of colluding members as suitable crowdworkers. In other words, we intend to identify whether the addition of each new participant to the previously selected group will result in the formation of a group of colluders within the selected participants.

Colluders are like-minded people who collaborate with each other on a specific agenda to obtain an objective by defrauding or gaining an unfair advantage. Their objective may be earning monetary or non-monetary profits. Colluders usually form a group which is large enough to make a considerable impact [20]. Moreover, group members usually target a considerable number of tasks and collaborate together in contributing to these tasks. Their contributions are typically similar to each other (in order to overwhelm the task with similar faulty contributions) and deviate from the other (genuine) participants (so as to change the task's outcome). Finally, the colluders may prefer to connect with each other in the form of social groups to facilitate their communications. Based on these collaborative behaviours, the collusion prevention method considers the following collusion indicators: (i) group size (i.e. number of colluders), (ii) group target size (i.e. number of tasks in which colluders have collaborated in the past), (iii) group deviation (i.e. an indicator to show the deviation of content produced by the colluders from those of other honest participants), (iv) group connectivity degree (an indicator to show to what extend the colluders are socially connected to each other), an (v) group content similarity (i.e., the degree of similarity of content produced by the group members). By considering all these indicators, the collusion prevention method determines a collusion probability for each participant and prevents the selection of the colluding participants.

In summary, the main contributions of this paper are as follows:

- We propose a method which is responsible for calculating a collusion possibility for each eligible participant to prevent any possible collusion on the task .
- We introduce five collusion indicators that resemble the behavior of colluding group members. We then provide equation to quantify each indicator and combine them to reach to a single value as the possibility of collusion .
- The accuracy and usability of the proposed techniques have been tested using real world datasets from the Advogato social network and Wikipedia Adminship Election and simulated experiments. The evaluation results show superiority of our method over the other common recruitment methods.

The rest of the paper is organised as follows. Related work is discussed in Section 2. We present the details of our collusion detection scheme in Section 3. Simulation results are discussed in Section 4. Finally, Section 5 concludes the paper.

2. Related Work

Social participatory systems can be regarded as a subset of collective intelligence systems, which are defined broadly as groups of individuals doing things collectively that seem intelligent [37]. Due to the openness of such systems, the selection and recruitment of well-suited participants has always been a great concern [38]. In the following, we will have a short review on the related works on the issue of collusion prevention among selected participants and will discuss the state-of-the-art.

Collusion detection has been widely studied in P2P systems [39, 40]. A comprehensive survey on collusion detection in P2P systems can be found in [39]. In their work, the authors extensively study and classify the reputation systems and micro-payments systems (MPS) as two main approaches in P2P systems against collusion. Reputation management systems are also targeted by collusion. Colluders in reputation management systems try to manipulate reputation scores by collusion. Much effort is put into detecting collusion using majority rules, weight of the worker and temporal analysis of the behavior of the users [41], but none of these methods are strong enough to detect all sorts of collusion [41].

In [20], Mukherjee et al. have proposed a model for spotting fake review groups in online rating systems. The model analyzes textual feedback cast on products in Amazon's online market to find collusion groups. They use eight indicators to identify colluders and propose an algorithm for ranking collusion groups based on their degree of spamicity. However, their proposed method is still vulnerable to some attacks. For example, if the number of attackers is much higher than honest raters on a product the model cannot identify this as a potential case of collusion.

In the domain of participatory sensing, the authors in [18] aim at detecting the collusion by leveraging a reputation management system and outlier detection algorithms. In [15], a trusted platform module (TPM) is provided with each sensor device to attest the integrity of sensor readings. This local integrity checking makes the system resistant to collusion. To the best of our knowledge, the collusion prevention has not been discussed in social participatory sensing, and the methods proposed for participatory sensing are not applicable to this domain.

3. Collusion Detection

An online social network is best represented as an undirected graph with the set of nodes representing participants and the set of friendship relations between nodes. Each participant has a profile containing his attributes and related information. Some attributes represent the participant's personal information such as name and address. Others include the outcome of participant's social behaviour. Examples are the participant's reputation score, the history of his previous transactions, the pairwise trust scores, etc. A participatory task or simply a task is represented by θ_i, and Θ is the set of all the tasks to be solved ($\Theta = \{\theta_i\}$. The owner of the task is also called the *requester*. Ψ is the set of *participants* who contribute to the task ($\Psi = \{\psi_i\}$). They provide the requester with a set of contributions represented by κ.

As mentioned above, selecting well-suited participants is important for acquiring high quality contributions since these participants have better knowledge and expertise relevant to the task requirements. In our previous works [21, 22, 23, 24], we addressed the challenge of well-suited participant selection. Specifically, we proposed schemes and procedures for crawling through the social network starting at the requester and identifying well-suited participants. We define the suitable participants as those who can satisfy the task requirements. In order to evaluate the participant's suitability, we defined and quantified a set of suitability parameters. These parameters include the participant's expertise to satisfy the task's skill requirements, his locality for location-based tasks, his reputation score, etc. The suitability parameters are evaluated for each eligible participant and combined to form a suitability score for him.

Once the participant ψ_i is considered to be suitable for being selected, a final check should be done to ensure that the selection of ψ_i will not result in potential collusion. In particular, we aim to identify whether the addition of ψ_i to the set of previously selected participants will result in the formation of a group of colluders.

Collaborative attacks which are also called collusion attacks are those in which, a number of individuals form a clique and collaborate on changing the results of a task [20]. For example, colluders may collaborate as they wish to produce poor quality contributions that severely impact the goal of the task.

We define a *group g* consisting of a set of participants Ψ^g and a set of tasks Θ^g. In other words, g = $\{\Psi^g, \Theta^g\}$. All the participants in Ψ^g have contributed to all the tasks in Θ^g.

Identifying the collusive groups requires two steps. In the first step, all existing collaborative groups (that fit within the above definition) are identified. In the second step, the potential collusive groups are detected among the identified groups. The detection of collusive groups is carried out based on a set of indicators. In the following section, we discuss these steps in detail.

3.1 Identifying Potentially Colluding Groups

In order to identify all collaborative groups among the selected participants, the collusion prevention method employs the Frequent Itemset Mining (FIM) technique [25]. FIM is a method for market basket analysis. It aims at finding regularities in the shopping behaviour of customers of supermarkets, mail-order companies, on-line shops etc. More specifically, FIM intends to find sets of products that are frequently bought together. There are multiple applications for the identified frequent item sets such as improving arrangement of products in shelves, on catalogue pages etc., supporting cross-selling (suggestion of other products), product bundling and fraud detection [26, 27, 28]. Identified patterns are typically expressed as association rules, e.g., if a customer buys bread and butter, then this customer will probably buy cheese, too. The performance and accuracy of the FIM technique is discussed in [29]. FIM is one of the major group detection algorithms which have been extensively used for collusion detection in online rating systems [20]. Hence, in our collusion prevention method, we make use of the FIM algorithm to find potential collusive groups.

The description of the FIM is as follows [29]: Let I = $\{i_1, i_2, \dots, i_n\}$ be a set of items and D be a multiset of transactions, where each transaction T is a set of items such that $T \subseteq I$. For any $X \subseteq I$, we say that a transaction T contains X if $X \subseteq T$. The set X is called an itemset. The count of an itemset X is the number of transactions in D that contain X. The support of an itemset X is the proportion of transactions in D that contain X. An item set X is called *frequent* if its support is greater than or equal to some given percentage s, where s is called the minimum support. In our context, the set of items *(I)* is the set of all selected participants for the current task. The set of transactions *(D)* is the set of all tasks that a participant has been involved in the past. By mining frequent itemsets, we find groups of participants who have contributed to multiple tasks together.

3.2 Collusion Indicators

Most existing collusion detection techniques rely on 'behavioural' indicators to identify colluding groups [20, 30, 31]. These indicators reflect suspicious behaviour from a group of participants which indicates the possibility of collusion. Colluders usually form a group which is typically large enough to gain the majority and make a considerable impact [20]. Moreover, group members usually target a considerable number of tasks and collaborate together in contributing to these tasks. We also claim that the colluders prefer to connect with each other in the form of groups (such as social groups in OSNs) to facilitate their communications.

Group connivance is also represented by some 'content-related' indicators. Colluders normally report contributions with typically similar (duplicate or near duplicate) contents in order to ensure that the task outcome is different from the true consensus. Moreover, their contributions deviate from the other (genuine) participants in order to change the task outcome. In order to have a better view of content-based collusion indicators, we provide an illustrative example. Recall the PetrolWatch application [5] in which, participants are recruited to take photos of fuel price billboards. The photos are then aggregated in the server and the fuel prices are extracted. The cheapest fuel price for each area is then identified (for example by leveraging majority consensus). People are then able to query the server to access the cheapest fuel price in their area of interest. Consider a situation in which, a service station operator is aware that there is a contest between the nearby stations to have more costumers. The operator is aware that PetrolWatch uses majority consensus and comes up with a plan to game the system with the aim of attracting more customers to his business. The service station operator asks several of his social friends to collusively report false data for the competing service stations by uploading old pictures of higher fuel prices. If the false prices reported by his friends are more than the correct prices reported by other people, the collusion attack will be successful.

Table 1. Fuel prices of three different service stations uploaded by eight participants The abbreviation "inc." is used to denote incorrect prices (e.g., due to not being able to successfully recognise the price in the image).

Participants	Station1	Station2	Station3
1	123.0	119.0	Inc.
2	123.0	119.0	Inc.
3	123.0	119.0	121.3
4	123.0	119.0	Inc.
5	123.0	119.0	121.3
6(m)	123.0	125.0	124.5
7(m)	123.0	125.0	124.5
8(m)	123.0	125.0	124.5
correct ¢	123.0	119.0	121.3
majority consensus ¢	123.0	119.0	124.5

Table 1 (taken from [18]) is an example of this scenario that represents the fuel prices reported by 8 participants. We assume that the malicious operator owns service station 1 and that participants ψ_6 , ψ_7, and ψ_8 (denoted by 'm' in the table) are his workers who have formed a collusive group and report higher prices for service stations 2 and 3. As shown in Table 1, all the colluding members report the same data (¢125.0 for station 2 and ¢124.5 for station 3) to set a higher price for these stations. Also, the price reported by the group members deviates from the prices forwarded by other genuine participants 1-5 in order to change the outcome of majority consensus. The result obtained from the majority consensus (the last row of Table 1) shows that the colluding group is successful in falsifying the genuine price of service station 3 since they constitute the majority and hence, their reported price is selected as the true price. This example illustrates the need to examine certain features that suggest the likelihood of the existence of a colluding group.

Similar to the concepts discussed above, in our collusion prevention method, we consider a set of indicators. These indicators suggest that a colluding group is likely to exist among the selected participants. Note that these indicators reflect the likelihood of collusion only when they all occur together. In the following, we explain each indicator in detail and discuss how they identify possible collusive activities.

- **Group Size (GS).** The first indicator is the group size which is proportional to the *number of colluders* who have collaborated as a group in similar tasks. Group size (normalised) for a group g *(GS^g)* is calculated as follows,

  $$GS^g = \frac{|\Psi^g|}{\max(|\Psi^g|)} \tag{1}$$

  Where $\max(|\Psi^g|)$ is the largest group size of all found groups. *GS* is a parameter in the range of (0, 1], i.e. $0 < GS \leq 1$, showing how large the group is in comparison with other groups.

- **Group Target Size (GTS).** While the group size measures the **number** of group members, group target size measures the *number of tasks* in which the group members have targeted to collaborate in the past. Groups with a high value of target size are more likely to be colluding as the probability of a group of random people to have attended the same tasks together is rather small. For a group g, *GTS^g* is calculated as follows.

  $$GTS^g = \frac{|\Theta^g|}{\max(|\Theta^g|)} \tag{2}$$

  Where max $(|\Theta^g|)$ is the largest target size of all found groups. GTS is a number in range (0, 1], i.e. $0 < GTS \leq 1$.

- **Group Deviation (GD).** The third indicator is group deviation which is an indicator to show the difference between the contents contributed by the colluders and those reported by other (honest) participants. In order to calculate the group deviation, we first calculate the deviation of the contents produced by group members from those of other participants for a single task $t \epsilon \Theta^g$. For each task $t \epsilon \Theta^g$, the deviation of the group (GD_t^g) is calculated as follows:

  $$GD_t^g = \left| \overline{\underset{\iota \epsilon \Psi^g}{\kappa_{\iota,t}}} - \overline{\underset{j \notin \Psi^g}{\kappa_{j,t}}} \right|, for\ all\ i, j \epsilon \Psi^g \tag{3}$$

  Where $\overline{\kappa_{\iota,t}}$ and $\overline{\kappa_{j,t}}$ are the average of contents for task t given by members of group g and by other participants not in g, respectively.

  Now, for a group g, the group deviation, denoted by *GD^g*, is the maximum of all group deviations for all tasks in Θ^g. In other words, *GD^g* is computed as:

$$GD^g = \max_{t\epsilon\Theta^g}(GD_t^g) \quad (4)$$

GD is a number in range (0, 1], i.e. $0 < GD \leq 1$.

- **Group Connectivity (GC).** The fourth indicator which is specifically suited for social communities is the group connectivity degree which is an indicator to show to what extent the colluders are connected to each other. For a group g, we first calculate the number of links between group members and denote it by link count (LC^g). LC^g is calculated as:

$$LC^g = \sum_{i\epsilon\Psi^g}\sum_{j\epsilon\Psi^g} T_{i,j} \; for\ all\ i, j\epsilon\Psi^g \quad (5)$$

Where,

$$T_{i,j} = \begin{cases} 1 & if\ i \rightarrow j\ (there\ is\ a\ link\ from\ i\ to\ j) \\ 0 & otherwise \end{cases}$$

GC^g is then computed as follows.

$$GC^g = \frac{LC^g}{\max(LC^g)} \quad (6)$$

GC is a number in range (0, 1], i.e. $0 < GC \leq 1$.

- **Group Content Similarity (GCS).** The fifth indicator is group content similarity which indicates the degree of similarity of contents produced by group members. In order to evaluate this similarity, we first calculate the pairwise content similarity between every pair of members in the group. Pairwise content similarity between ψ_i and ψ_j, denoted by $GCS_{i,j}^g$, shows to what extent ψ_i and ψ_j have reported similar contents. In order to calculate the pairwise similarities between group members, we use the cosine similarity model, a well-known model for similarity detection [32]. Specifically, $GCS_{i,j}^g$ will be the cosine of the angle between two vectors containing the contribution contents of ψ_i and ψ_j and is a value in the range (0, 1). The value 1 for $GCS_{i,j}^g$ means completely the same while 0 means completely different. $GCS_{i,j}^g$ is calculated as follows.

$$GCS_{i,j}^g = \frac{\sum_{t\epsilon\Theta^g}\kappa_{i,t}\times\kappa_{j,t}}{\sqrt{\sum_{t\epsilon\Theta^g}(\kappa_{i,t})^2}\times\sqrt{\sum_{t\epsilon\Theta^g}(\kappa_{j,t})^2}} \quad (7)$$

We then calculate an overall degree of similarity for the group to show how all members are similar in terms of contents they have contributed. Group content similarity for every group g, denoted by GCS^g is the minimum amount of pairwise similarities between group members. In other words,

$$GCS^g = \min_{i,j\epsilon\Psi^g}(GCS_{i,j}^g) \quad (8)$$

GCS is a number in range (0, 1], i.e. $0 < GCS \leq 1$.

3.3 Possibility of Collusion

It is often difficult to determine with certainty whether a group is collusive [20]. Therefore, we define a metric called *Possibility of Collusion (PoC)* to show to what extent a group is potentially collusive. *PoC* is an aggregation of five collusion indicators. Since the importance of these indicators may be different in various applications, the collusion prevention method enables the applications to assign weight to each indicator based on its importance.

Suppose that W_{GS}, W_{GTS}, W_{GD}, W_{GC} and W_{GCS} are corresponding weights for indicators GS^g, GTS^g, GD^g, GC^g and GCS^g. The weights are initialised in a way that: $W_{GS} + W_{GTS} + W_{GD} + W_{GC} + W_{GCS} = 1$ and are set based on the application settings and nature, as described above. *PoC* is then calculated as:

$$PoC(g) = GS^g \times W_{GS} + GTS^g \times W_{GTS} + GD^g \times W_{GD} + GC^g \times W_{GC} + GCS^g \times W_{GCS} \quad (9)$$

PoC is a number in range (0, 1]. For each eligible participant ψ_i to be selected, we calculate the possibility of collusion (*PoC(g)*). If greater than a certain threshold, it implies that the selection of ψ_i may lead to potential collusion, and hence, the participant will not be selected.

4. Experimentation and Evaluation

In this section, we conduct a simulation-based evaluation to analyse the behaviour of our proposed collusion prevention method. First, we explain the experimentation set up and the datasets we used in experiments in Section 3.1. Then in Section 3.2, we investigate the efficiency of our proposed collusion prevention method.

4.1. Simulation Set-up

Our simulations have been conducted on a PC running Windows 7:0 Professional and having 4GB of RAM. We used Matlab R2012 for developing the simulator.

4.1.1 Datasets

In order to evaluate the performance of our proposed collusion prevention method, we set two experiments. In the first experiment, we aimed at utilising a real dataset for which, the possibility of collusion exists due to gaining benefits. Hence, we utilised the Wikipedia voting dataset. In Wikipedia, the voting process is used to elect administrators[4]. Every registered user can nominate himself or another user as an administrator in Wikipedia and initiate an election. The other users participate in the election and cast their votes on the eligibility of nominee. If the majority of users recognise a user as eligible, this user then will become a Wikipedia administrator. In order to incorporate this dataset in the context of our method, we employ the following mapping. The requester is the nominee, the worker is the voter, the task is evaluating the

[4]http://en.wikipedia.org/wiki/Wikipedia:Requests_for_adminship

eligibility of the nominee as an administrator in Wikipedia and the contribution is the worker's vote. We use the log of Wikipedia Adminship Election[5] which was collected by Leskovec et al. for behaviour prediction in online social networks [33], referred to as WIKILog. WIKILog contains about 2800 elections (tasks) with around 100000 total votes and about 7000 users participating in the elections either as a voter or a nominee. We use the WIKILog to demonstrate the efficacy of our proposed method to detect collusion.

The dataset that we use for the second experiment is the real web of trust of Advogato.org [34]. Advogato.org is a web-based community of open source software developers in which, site members rate each other in terms of their trustworthiness. Trust values are one of the three choices master, journeyer and apprentice, with master being the highest level in that order. The result of these ratings is a rich web of trust, which comprises of 14019 users and 47347 trust ratings. In order to conform it to our framework, we map the textual ratings to the range of [0, 1] as master = 0.8, journeyer = 0.6, and apprentice = 0.4. Advogato web of trust can be regarded as a social participatory sensing system with users as the potential participants and trust ratings as the friendship relations. In order to better investigate the performance of our method, we artificially created collusive groups among Advogato members. We then investigated whether the proposed collusion prevention method is able to identify these groups.

4.2 Collusion Prevention Analysis

As mentioned in Section 3.1, to evaluate the performance of the collusion prevention method, we set two experiments. The experiments differ in their employed datasets. In the following, we explain the results of each experiment in detail.

4.2.1 Wikipedia Adminship Election Dataset

In the first experiment, we use the Wikipedia adminship election dataset to investigate the performance of our proposed collusion prevention method. The dataset contains the information related to 2794 tasks. The average number of participants in these tasks is 40. In order to obtain reliable results, we consider the tasks with number of participants greater than the average as the sample data, and randomly select 100 tasks from these. We then test our proposed method to identify any potential colluding group among the participants. As mentioned in Section 2.2, we consider five indicators for detecting potential collusion. Among these indicators, the two indicators Group Size (GS) and Group Target Size (GTS) are the most important indicators as they are the basic conditions for the f*ormatio*n of a group. Basically, a group *g* is created when at least th_1 members of *g* have collaborated in at least th_2 tasks. So, we first run a short experiment to define the optimal values for th_1 and th_2.

In order to find the optimum value for th_2, we set an experiment in which, the target size (i.e. number of the tasks for which the group members have collaborated in the past) is changed. For each target size, we measure the number of groups identified, together with their size. As can be seen in Fig. 1, the maximum size of identified groups is decreased by increasing the target size. This is rational since the probability of finding groups whose members have collaborated in a greater number of tasks is smaller. We believe that the best setting is the one which results in the identification of the largest groups to make a considerable impact. As derived from the figure, this situation is related to the case where the target size is 6. So, we set th_2 to be equal to 6. For the sake of simplicity, we assume that th_1 equals to th_2.

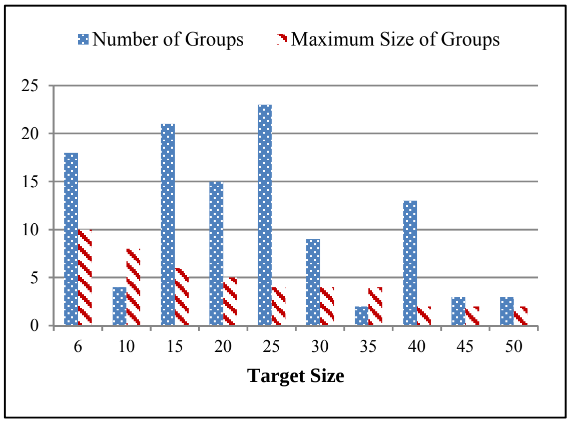


Fig. 1: Evolution of number of groups and their maximum size according to the target size.

In order to investigate the performance of our proposed collusion prevention method, we first utilise the FIM technique to find the candidate groups among the participants. The outcome is the discovery of 18 candidate groups with at least 10 members. We then employ our collusion prevention method and identify 9 of these 18 groups as collusive. To evaluate the efficiency and accuracy of our method, we examine a number of statistical metrics. At first, we measure the ratio of the tasks targeted with the colluding groups. The result shows that 14% of the tasks were affected by these 9 colluding groups. This means that

[5] http://snap.stanford.edu/data/wiki-Elec.html

our collusion prevention method is able to prevent these tasks from being targeted by the colluders. We then calculate the success ratio of the tasks targeted by the colluding groups as well as all 100 tasks. In the Wikipedia adminship election dataset, a task (an election) is successful if it results in the selection of the user as an administrator (note that the results are available in the dataset). By success ratio, we mean the ratio of the tasks that have resulted in a desired decision (i.e. resulted in the selection of a user as an admin), to the total number of tasks. We observe that overall success ratio of the tasks in our dataset is 71%. This ratio is 83% for the groups identified by our collusion detection method. This means that there is a high probability that the groups identified by our method are colluding groups, since their collaboration has resulted in a considerably high success ratio. This is a significant indication that the identified groups are much more likely to be collusive.

4.2.2 Advogato Dataset

In the second experiment, we use Advogato dataset. We first create a set of candidate groups among the Advogato members, and then, we define some of these candidate groups as collusive. In order to create candidate groups, we first select 90 Advogato members with at least 30 trust relations (i.e. 30 friends). Each of these members along with 20 out of his 30 friends forms a candidate group. When a task is released, a set of Advogato members are considered as eligible to contribute (by using the aforementioned suitability assessment and eligibility assessment techniques). Each candidate group with at least 10 eligible members is considered as collusive. The collusive group members contribute polluted data while other eligible members contribute genuine data. Specifically, we assume that the genuine data (d) is a random number in [0,1], while the polluted data is a random number in ($d - \mu$, $d + \mu$). Greater values for μ result in polluted values with great deviation from the genuine values, which makes the collusion detection easier. In our experiments, we set μ to be 0.2. Note that for each task, all the collusive members report the contaminated data, while others report the genuine data. We run the experiment for 10 rounds. In each round, 20 tasks are released. At the end of each round, we utilise the FIM technique to find the groups. The outcome is the set of all groups among the eligible participants (who have collaborated in at least 5 tasks). Then, for each group identified by FIM, the possibility of collusion (*PoC*) is computed by utilising Equation 9. While we believe that the threshold for *PoC* should be application-specific, in our experiments, we assume that groups with *PoC* > 0.5 are identified as collusive (In Equation 9, for simplicity, we assume that all the indicator weights are equal to 0.2).

In order to evaluate the efficiency and the accuracy of our proposed method in identifying the colluding groups, we utilise two criteria. For the evaluation of accuracy, we use the well-known measures of precision and recall [35]. Precision measures the quality of the identification results, and is defined by the ratio of the correct identification of colluding groups, to the total number of groups identified by our method. Recall measures coverage of the identification results, and is defined by the ratio of the collusive groups identified correctly to the total number of all correct colluding groups that should be found. These two definitions are summarised in the following equations:

$$Precision = \frac{number\ of\ collusive\ groups\ identifised\ correctly}{total\ number\ of\ identified\ groups}$$

$$\boldsymbol{Recall = \frac{number\ of\ collusiveroups\ identifised\ correctly}{total\ number\ of\ existing\ collusive\ groups}}$$

These two measures are usually expressed as percentages. For an approach to be effective, it should achieve a high precision and high recall. However, in reality these two metrics tend to be inversely related [36]. This means that the improvements in precision come at a cost of reduction in recall, and vice versa.

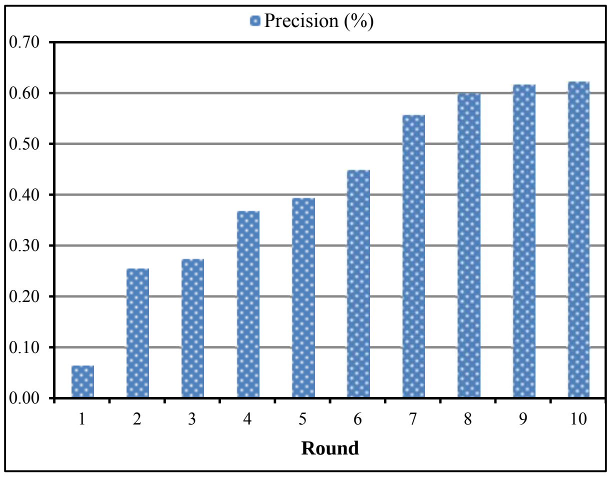


Fig. 2. Evolution of precision (%) in different rounds.

Fig. 2 shows the evolution of precision in different rounds. As displayed in this figure, the collusion detection method achieves a precision of 63%. This means that our collusion prevention method is able to prevent 63% of the tasks from being targeted by the colluders. This is due to the suitability of the indicators, which correctly model the collusive behaviour of group members. Note that it may be possible to achieve greater precision but would result in a drop in recall. As can be observed in this figure, the precision values evolve in a constantly in-creasing manner. A lower value of precision in the first rounds is due to the lack of adequate history related to the colluders' behaviours. In other words, due to the small number of released tasks in the first rounds of the experiment, the collusion prevention method does not have the required information (e.g., content similarity, target size, etc.) at hand. As time goes by, collusive members collaborate in more tasks which results in the availability of more behavioural information such as number of the tasks they have collaborated on, the contributions they have re-ported to these tasks, etc. This helps the collusion prevention method to better detect the collusive behavioural pattern.

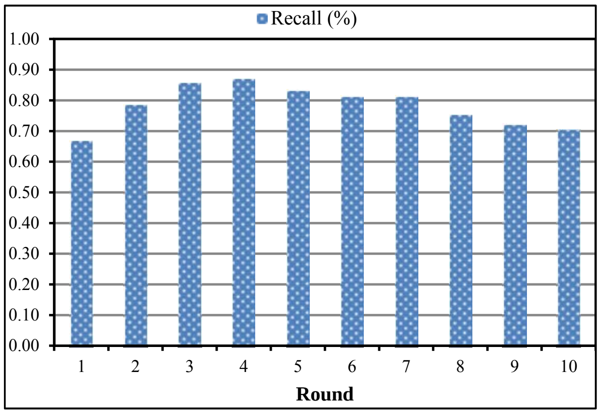


Fig. 3: Evolution of recall (%) in different rounds.

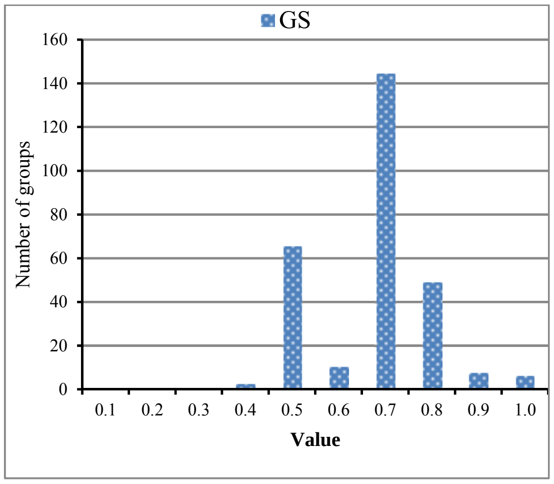
Fig. 3 depicts the evolution of recall in various rounds. As can be seen in this figure, our method also achieves a high percentage of recall (86%), which denotes that our collusion prevention method is successful in detecting 86% of the existing collusive groups. It can be observed that there is a slightly descending growth in recall after the fourth round which, as mentioned above, is natural in real systems, since precision and recall typically evolve inversely [36].

As mentioned above, a group is identified as collusive if the possibility of collusion (*PoC*) for this group is above 0.5. The possibility of collusion is obtained by averaging the indicator values. However, in order to ensure that the indicators are selected correctly, we calculate the distribution of values of each indicator in all collusive groups identified by our method. Figures 4.a. to 4.e depict the distribution of values calculated for collusion indicators. The values calculated for indicators are almost always higher than 0.5. This illustrates that the identified indicators are suitable and effective for detecting collusion in social participatory sensing.

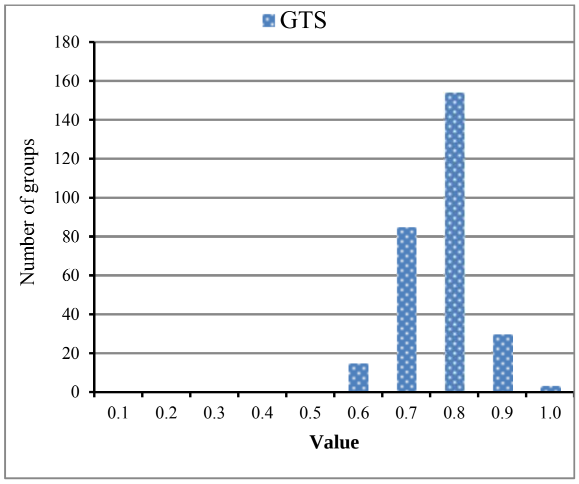
In a nutshell, the results show that our proposed collusion prevention method is successful in preventing the formation of colluding groups among the selected participants with high accuracy.

5. Conclusion

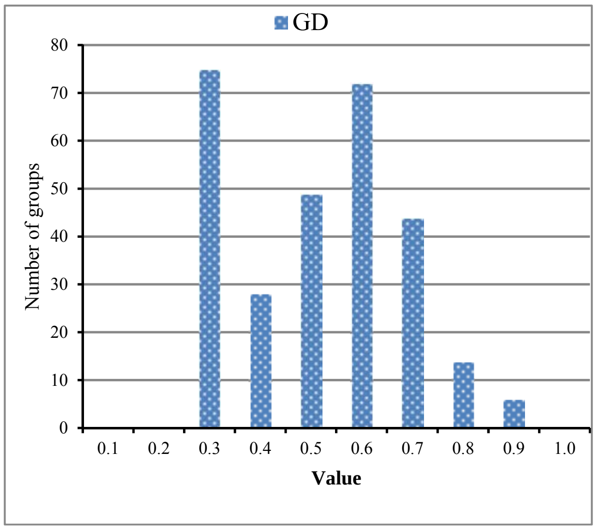
In this paper, we proposed a collusion-resistant participant selection method with the goal to prevent the formation of colluding groups within the selected suitable participants. The method investigates the possibility of collusion upon each eligible participant. This decision is made based on a set of indicators that are related to the common approaches utilised by colluders to arrange a collusive attack. Colluders normally form a large group and collectively collaborate on a large number of tasks. They normally contribute similar content which deviate from the genuine contributions provided by honest participants. They may also benefit from the social groups to better manage their communications. We then calculated the possibility of collusion based on these indicators. In order to measure the performance of the collusion prevention method, we set up two experiments in which, the datasets Wikipedia adminship election and Advogato were employed. The result of these experiments showed that our proposed method is able to detect the collusive groups with high precision. The results also demonstrated the correctness and effectiveness of proposed indicators.



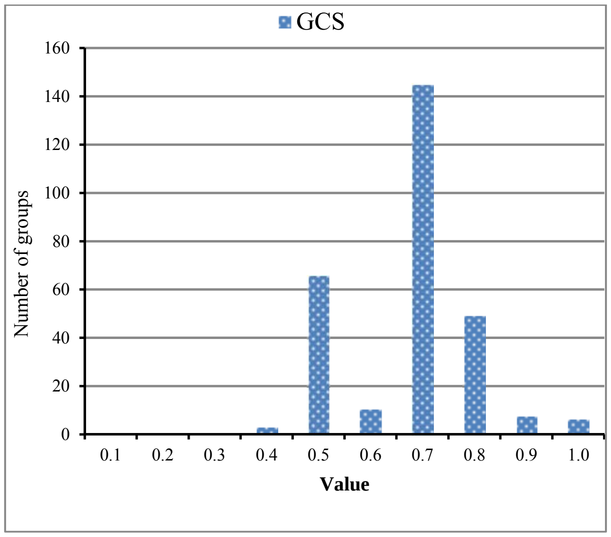
(a)



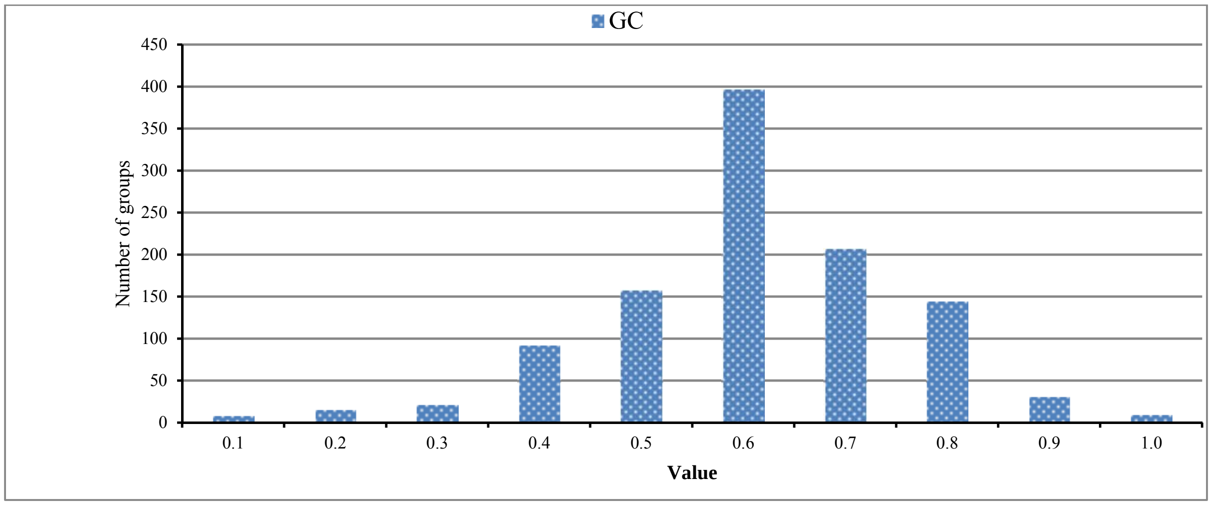
(b)



(c)



(d)



(e)

Fig. 4. Distribution of the values of indicators in collusion attacks

References

[1] A. Doan, R. Ramakrishnan, and A. Y. Halevy, "Crowdsourcing systems on the world-wide web," Communications of the ACM, vol. 54, no. 4, pp. 86–96, 2011.

[2] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and B. Srivastava, "Participatory sensing," in Proceedings of the 1st Workshop on World-Sensor-Web (WSW), 2006, pp. 1–5.

[3] S. Reddy, A. Parker, J. Hyman, J. Burke, D. Estrin, and M. Hansen, "Image browsing, processing, and clustering for participatory sensing: Lessons from a dietsense prototype," in Proceedings of the 4th ACM Workshop on Embedded Networked Sensors, 2007, pp. 13–17.

[4] E. P. Stuntebeck, J. S. Davis II, G. D. Abowd, and M. Blount, "Healthsense: classification of health-related sensor data through user-assisted machine learn-ing," in Proceedings of the 9th workshop on Mobile computing systems and applications. ACM, 2008, pp. 1–5.

[5] Y. F. Dong, S. S. Kanhere, C. T. Chou, and R. P. Liu, "Automatic image capturing and processing for petrolwatch," in Proceedings of the 17th IEEE International Conference on Networks (ICON), 2011, pp. 236–240.

[6] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, Balakrishnan, and S. Madden, "Cartel: a distributed mobile sensor comput-ing system," in Proceedings of the 4th international conference on Embedded networked sensor systems. ACM, 2006, pp. 125–138.

[7] R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu, "Ear-phone: an end-to-end participatory urban noise mapping system," in Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), 2010, pp. 105–116.

[8] P. Johnson, A. Kapadia, D. Kotz, N. Triandopoulos, and N. Hanover, "People-centric urban sensing: Security challenges for the new paradigm," TR2007-586, Dartmouth College, Computer Science, Hanover, NH, Tech. Rep., 2007.

[9] K. Shilton, "Four billion little brothers?: Privacy, mobile phones, and ubiqui-tous data collection," Communications of the ACM, vol. 52, no. 11, pp. 48–53, 2009.

[10] D. Christin, A. Reinhardt, S. S. Kanhere, and M. Hollick, "A survey on privacy in mobile participatory sensing applications," Journal of Systems and Software, vol. 84, no. 11, pp. 1928–1946, 2011.

[11] I. Krontiris and F. C. Freiling, "Integrating people-centric sensing with social networks: A privacy research agenda," in Proceedings of the 8th IEEE Inter-national Conference on Pervasive Computing and Communications Workshops (PERCOM), 2010, pp. 620–623.

[12] I. Krontiris and F. Freiling, "Urban sensing through social networks: The tension between participation and privacy," in Proceedings of the International Tyrrhenian Workshop on Digital Communications (ITWDC), 2010.

[13] M. Demirbas, M. A. Bayir, C. G. Akcora, Y. S. Yilmaz, and H. Ferhatosman-oglu, "Crowd-sourced sensing and collaboration using twitter," in Proceedings of the IEEE International Symposium on World of Wireless Mobile and Multi-media Networks (WoWMoM), 2010, pp. 1–9.

[14] S. Reddy, D. Estrin, and M. Srivastava, "Recruitment framework for partici-patory sensing data collections," in Pervasive Computing, ser. Lecture Notes in Computer Science, 2010, vol. 6030, pp. 138–155.

[15] A. Dua, N. Bulusu, W.-C. Feng, and W. Hu, "Towards trustworthy participa-tory sensing," in Proceedings of the Usenix Workshop on Hot Topics in Security (HotSec), 2009, pp. 8–8.

[16] S. Saroiu and A. Wolman, "I am a sensor, and i approve this message," in Proceedings of the 11th ACM Workshop on Mobile Computing Systems and Applications (HotMobile), 2010, pp. 37–42.

[17] "Trusted computing group," https://www. trustedcomputinggroup.org/home.

[18] K. L. Huang, S. S. Kanhere, and W. Hu, "On the need for a reputation system in mobile phone based sensing," Ad Hoc Networks, vol. 12, no. 0, pp. 130–149, 2014.

[19] S. Ganeriwal, L. K. Balzano, and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," in ACM Transactions on Sensor Networks (TOSN), 2008, vol. 4, no. 3, p. 15.

[20] A. Mukherjee, B. Liu, and N. Glance, "Spotting fake reviewer groups in consumer reviews," in Proceedings of the 21st ACM International Conference on World Wide Web, 2012, pp. 191–200.

[21] H. Amintoosi and S. Kanhere, "A reputation framework for social participa-tory sensing systems," in Mobile Networks and Applications (MONET), 2014, vol. 19, no. 1, pp. 88–100.

[22] H. Amintoosi and S. S. Kanhere, "A trust-based recruitment framework for multi-hop social participatory sensing," in Proceedings of the 9th IEEE Inter-national Conference on Distributed Computing in Sensor Systems (DCOSS), 2013, pp. 266–273.

[23] H. Amintoosi and S. Kanhere, "Privacy-aware trust-based recruitment in so-cial participatory sensing," in 10th International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services (MobiQuitous), 2013, pp. 262–275.

[24] H. Amintoosi, S. S. Kanhere, and M. Allahbakhsh, "Trust-based privacy-aware participant selection in social participatory sensing," Journal of Infor-mation Security and Applications, vol. 20, pp. 11–25, 2015.

[25] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in Proceedings of the 20th International Conference on Very Large Data Bases (VLDB), vol. 1215, 1994, pp. 487–499.

[26] T. Brijs, B. Goethals, G. Swinnen, K. Vanhoof, and G. Wets, “A data mining framework for optimal product selection in retail supermarket data: the generalized profset model,” in Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2000, pp. 300–304.

[27] T. Brijs, G. Swinnen, K. Vanhoof, and G. Wets, “Using association rules for product assortment decisions: A case study,” in Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999, pp. 254–260.

[28] W. Lee, S. J. Stolfo, and K. W. Mok, “Adaptive intrusion detection: A data mining approach,” Artificial Intelligence Review, vol. 14, no. 6, pp. 533–567, 2000.

[29] G. Grahne and J. Zhu, “Fast algorithms for frequent itemset mining using fp-trees,” IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 10, pp. 1347–1362, 2005.

[30] M. Allahbakhsh, A. Ignjatovic, B. Benatallah, S.-M.-R. Beheshti, E. Bertino, and N. Foo, “Collusion detection in online rating systems,” in Proceedings of the 15th Asia Pacific Web Conference (APWeb), 2013, pp. 196–207.

[31] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw, “Detecting product review spammers using rating behaviors,” in Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM), 2010, pp. 939–948.

[32] G. Salton, C. Buckley, and E. A. Fox, “Automatic query formulations in in-formation retrieval,” Journal of the American Society for Information Science, vol. 34, no. 4, pp. 262–280, 1983.

[33] J. Leskovec, L. Adamic, and B. Huberman, “The dynamics of viral market-ing,” ACM Transactions on the Web (TWEB), vol. 1, no. 1, pp. 1–39, 2007.

[34] R. Levien and A. Aiken, “Attack-resistant trust metrics for public key cer-tification,” in Proceedings of the 7th USENIX Security Symposium, 1998, pp. 229–242.

[35] H. R. Motahari Nezhad, G. Y. Xu, and B. Benatallah, “Protocol-aware matching of web service interfaces for adapter development,” in Proceedings of the 19th ACM International Conference on World Wide Web, 2010, pp. 731– 740.

[36] G. Chowdhury, Introduction to Modern Information Retrieval. Facet pub-lishing, 2010.

[37] T. Malone, R. Laubacher, and C. Dellarocas, “Harnessing crowds: Mapping the genome of collective intelligence,” in MIT Sloan Research Paper, 2009.

[38] S. S. Kanhere, “Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces,” in Distributed Computing and Internet Tech-nology. Springer, 2013, pp. 19–26.

[39] “Collusion in peer-to-peer systems,” in Computer Networks, 2011, vol. 55, no. 15, pp. 3517 – 3532.

[40] Q. Lian et. al., “An empirical study of collusion behavior in the maze p2p file-sharing system,” in 27th International Conference on Distributed Comput-ing Systems. IEEE Computer Society, 2007, pp. 56–.

[41] Y. Sun and Y. Liu, “Security of online reputation systems: The evolution of attacks and defenses,” in Signal Processing Magazine, IEEE, 2012, vol. 29, no. 2, pp. 87 –97.

Journal of Computer and Knowledge Engineering, Vol. 1, No. 1.
DOI: 10.22067/cke.v1i2.60500

Designing Optimized Scheduling QoS-Aware RPL for Sensor-Based Smart Grid Communication Network

Mohammad Alishahi[✦], Mohammad Hossein Yaghmaee Moghaddam and Hamid Reza Pourreza

Abstract: Various applications with different requirments are rapidly developed in the smart grid. The need to provide Quality of Service (QoS) for such a communication network is inevitable. However, recently a protocol called RPL (Routing Protocol for Low Power and Lossy Network) has been standardized and is known as the main solution for last mile communication network of smart grid. In this paper, by studying the existing methods and identifying the shortcomings, we propose a customized version of RPL which we call OMC-RPL (Optimized Multi Class-RPL). Two principal advantages of the proposed method are: a holistic objective function including distinctive metrics related to QoS; and supporting the data classification which is an important requirement in this context. The main contribution of this paper is to make different objective functions proportional to the number of classes by using weighting parameters. The best values of these coefficients are determined by an optimization algorithm. OMC-RPL is evaluated from different aspects. Simulation results show that the new idea significantly decreases the end-to-end delay and increases lifetime of the nodes that have limited source of energy. It seems that OMC-RPL could be a good substitution for the available methods.

Keyword: RPL, DODAG, QoS.

1. Introduction:

One of the basic issues in Smart Grid (SG) is a reliable and secure communication network that can support SG applications such as Advanced Metering Infrastructure (AMI), Demand Response (DR), Distribution Automation (DA) etc.. Within the communication network associated with the power grid, the SG Neighbor Area Network (NAN) and Home Area Network (HAN), as shown in

Fig. *1*, are faced with substantial communication challenges because of their size and traffic variation [1-3].

The IETF ROLL working group had a mission to propose a routing protocol for LLN (Low Power and Lossy Networks) which leads to RPL (Routing Protocol for LLN) standard in 2012 [4, 5]. LLNs are made up of a large number of embedded devices with limited power, memory, and resources that connect to each other using various communications protocols, such as IEEE 802.15.4, Wi-Fi, and power-line communication (PLC) [6]. RPL is the main candidate for acting as the standard routing protocol for IP smart object networks in NAN. This popularity is because of two reasons, one is its flexibility to adapt to different topologies, and the other is its capability of QoS support [7]. Distinctive types of applications in the smart grid, especially in NAN and HAN are experiencing the same situation as LLN. This last mile network is made of highly limited devices interconnected by fairly unstable low-quality links that cause different QoS requirements, which is not the same as the traditional IP networks [8]. Eke the QoS is an essential component of the overall architecture in the smart grid [9]. Some data such as alert or control signals have real-time requirements. Ergo the networking infrastructure somehow should guarantee the quality of service, for example, decreasing the end-to-end delay. Due to the necessity of QoS in SG and usability of RPL, in this paper, we propose an optimized QoS-aware RPL, which is completely suited for SG communication network (SGCN). The remaining sections of this paper are organized as follows. In section 2, the RPL is explained. In section 3 the related work on QoS in smart grid and especially those methods using RPL, are studied. Section 4 explains the proposed method. Finally, the last two sections are about simulation results and conclusion.

I. RPL

RPL is a distance-vector protocol that is based on the concept of a topological Directed Acyclic Graph (DAG). DAG uses a tree structure in which each node can have more than one parent. Specifically, RPL organizes these nodes into Destination Oriented DAGs (DODAG) whose roots are destination nodes – e.g., sinks, concentrators, or network gateways. Fig. 2 shows a sample DODAG with a similar structure to a tree that specifies the conventional route between the LLN nodes [7].

DODAGs are created and managed based on the objective function (OF). The OF specifies routing metrics and optimization goals and can construct routes to satisfy any requirements, such as quality of service. To construct a DODAG, the root sends the objective function via a standard IPv6 message to neighboring nodes. The DODAG's creation is finalized when the nodes select , using a general algorithm, their preferred parent, and rank. The rank of a node [10] is also computed by the objective function, which expresses the distance of the node from its root in relation to the given metrics; nodes closer to the root should have lower ranks.

Manuscript received November 23, 2016; accepted January 4, 2017.

Mohammad Alishahi, Department of Computer Engineering, Fariman Branch, Islamic Azad University, Fariman, Iran, Mohammad Hossein Yaghmaee Moghaddam, Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran.

Hamid Reza Pourreza, Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran.

[✦]The corresponding author's e-mail is: alishahi@mshdiau.ac.ir

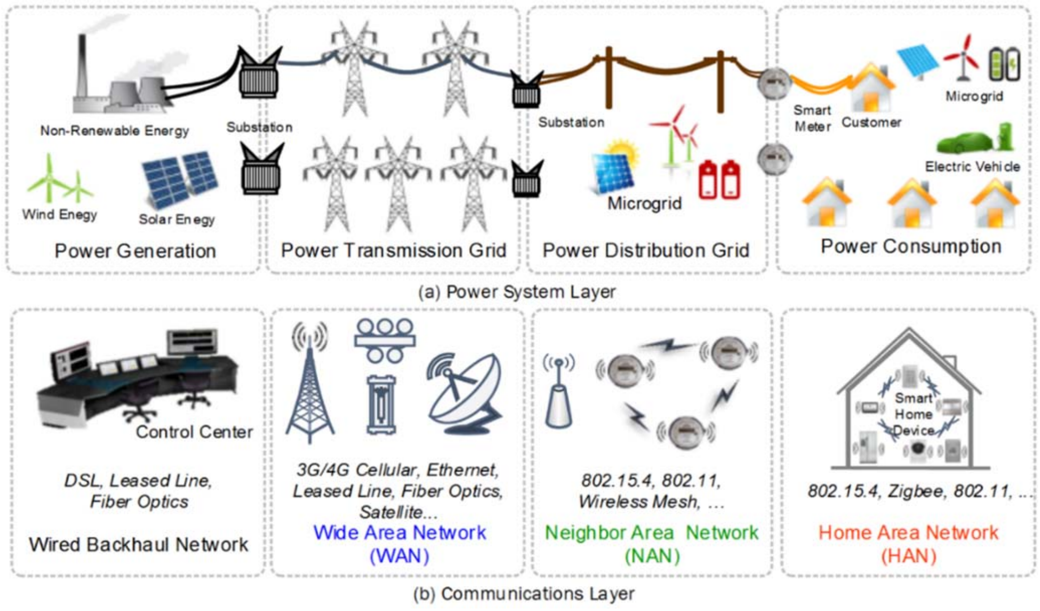


Fig. 1. Smart Grid System Architecture [3]

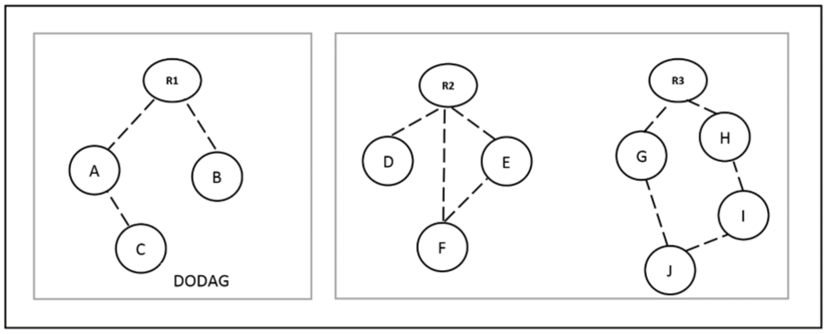


Fig. 2. A Sample of DODAG

The RPL protocol's process helps to create a self-configuring, self-healing, loop detecting system that will be suitable for NAN and HAN networks in the smart grid.

The specification of RPL does not force any routing metric and leaves it open to implementations. The proposed objective functions by the IETF presented in [10] and [11] have put forth some recommendations on how to implement OF without specifying usable routing metrics.

In RFC 6552 [10] the principle of the OF is described, which is called Objective Function Zero (OF0). As explained earlier, the two main duties of an objective function are choosing a proper rank and the preferred parents. RFC 6552 describes the principles and rules of defining an objective function based on the required metrics and constraints. For instance, there is a rule that says, a node with the lowest rank should be chosen as a preferred parent, but note that this document does not consider any routing metric specified in [12]. The proposed objective function in this paper is also based on these foundations.

One typical OF based on the metric of link quality is Expected Transmission Count (ETX) [11]. The main idea of this objective function is the probable amount of transmission to send a packet successfully. This OF is usually used in wireless environments. ETX has been widely used in recent research papers [13, 14].

2. Related Works

In this section, we investigate about the related works in two parts. The first part is about the concepts and the methods that try to ensure the QoS in smart grid, but the second part is only about the techniques that use RPL algorithm to achieve this goal.

A. QoS in Smart Grid

There are too many studies on QoS in the smart grid. Some of them focus on challenges and requirements of QoS in this area. For instance, [15] discusses that one of the most important requirements is that each system architecture should support a diverse set of QoS classes with a wide range of rate and delay requirements. Other studies in this area usually propose specific ways that somehow improve the QoS in the smart grid. For example, [16] uses the Differentiated Service (Diff-Serv) approach and some priority queues. [17] provides different services for various types of traffic in MAC layer for low-cost protocols like ZigBee. [18] studies about the scheduling and routing methods based on a Back-Pressure algorithm to guaranty the QoS.

B. Methods Using RPL to Guarantee QoS in Smart Grid

Although RPL has been released recently, several research studies have been presented to investigate about the subjects that are left open by the working group. In [19], two MAC-based routing metrics are used. The first one checks the ETX and the packet losses due to the MAC contention. The second metric selects the routes that have the acceptable traffic load by considering the power consumption, and the application required reliability. The proposed method is implemented by the author in a real testbed composed of seven Telos motes. The performance parameters in this paper are end-to-end reliability and the power consumption.

In [20] the impact of objective functions on the network topology is analyzed. LQL (Link Quality Level) is another objective function, which is based on the link condition. The author uses two objective functions (OF0 and LQL) for comparison. In [21] a combination of two routing metrics among hop counts, ETX, remaining energy, and RSSI is used. In fact, the first metric is responsible for choosing a parent with the lower rank. If the first values are equal, then the node with the lower rank of the second composition metric is selected as the preferred parent.

Another study [8] suggests a cross-layer QoS mechanism that merges a priority queue with multiple instances of RPL. The focus of this paper is on the MAC level QoS separation. Moreover, both RPL instances are based on the same objective function and root, but generate distinct DODAGs due to partitioning of the actual physical network (i.e., nodes are classified as regular or alarm, regular nodes are responsible for physical environment monitoring and generate data packets at a low rate; however, alarm nodes randomly generate small-size alert packets). This paper intends to extend the idea of QoS through multiple RPL instances by supporting priority traffic in MAC layer and exploring the effect of traffic differentiation at the network layer.

In [3] the QoS is guaranteed through traffic prioritization in MAC layer in a way that the random backoff mechanism is altered based on the traffic classes, and this is how they control the channel accessibility. The author compares the single instance RPL, multi-instance RPL and multi-instance RPL with prioritized channel backoff to see the effect of traffic differentiation at the network layer.

The author in [9] proposes a network-MAC cross-layer protocol based on incorporation of RPL and SCSP (Sleep Collect and Send Protocol) in wireless sensor networks. In fact, SCSP is a power-saving mechanism and media access control protocol. The RPL-SCSP guarantees fast transmission for critical data while reducing the energy consumption. In RPL-SCSP the preferred parent is slected based on the queue load; moreover, the nodes with the empty queue will be switching to an inactive state in order to extend the network lifetime.

Another study [22] believes that in order to optimize the path to the DODAG's root, the existing objective functions rely either on a single metric or on the combination of the two metrics. Thus a novel objective function based on the fuzzy parameters has been designed. Four different metrics, including end-to-end delay, hop counts, link quality and remaining energy are used to propose holistic objective function by using fuzzy logic. The proposed fuzzy system is a four-input controller with three membership functions for each input that leads to 81 rules. Eventually, by using centroid defuzzification method, control action based on several membership values is produced.

3. THE PROPOSED OMC-RPL (Optimized Multi-Class-RPL) PROTOCOL

As studied in related work section, the existing protocols that offer the QoS by using RPL are usually faced with two major shortcomings:

1) Most of the approaches do not provide a comprehensive and holistic objective function. For example, an OF may improve the end-to-end delay by finding the most proper path towards the sink, but as all packets try to use the same path, it is possible to have a bad effect on energy consumption.

2) The data classification, which is one of the most important requirements in assuring the QoS is not supported by available methods. The main reason is that if we categorize the data, then each class type has its own specification, and it should be treated in a distinctive way. This means that we need different objective functions for each class of data, which is a notable challenge. Although some studies use two classes of data or two different OFs, but it is for multiple instances RPL. In fact, some networks may run multiple instances of RPL concurrently, but logically these instances are independent.

In this paper, we propose a customized RPL with holistic objective function. The proposed protocol is named OMC-RPL (Optimized Multi-Class RPL) which can support data classification.

Although DODAG construction in RPL is clear, but as OMC-RPL should support data classification, we need a new procedure. OMC-RPL is able to support several types of traffics. In order to better understand the DODAG construction process we provide a flowchart in Fig. 3, that shows the steps in creating DODAG regarding just two classes of data with two ranks for each node.

In the first step, the root broadcasts a message (including default values for $rank_1$ and $rank_2$ and the two objective functions) to the nodes in its vicinity . When a node receives a message from the root for the first time, the algorithm calculates two ranks for the node based on two different OFs sent by the root and creates a list which includes pairs of ($rank_1$, parent) and ($rank_2$, parent); naturally for the first time the root is the parent. In fact, we need two OFs to make the difference for two distinctive classes. The receiving nodes then broadcast a message with new routing information to their neighbors. Consequently, if it is not the first time that a node receives a message, the algorithm performs steps to decide if the message comes from a better parent with lower rank or not. Note that in DODAG, each node can have several parents; one as preferred parent and the others as a replacements in the case of failure. Furthermore, in our proposed method, each node may have different preferred parents proportional to the number of classes. Each parent is just suitable for its relevant class.

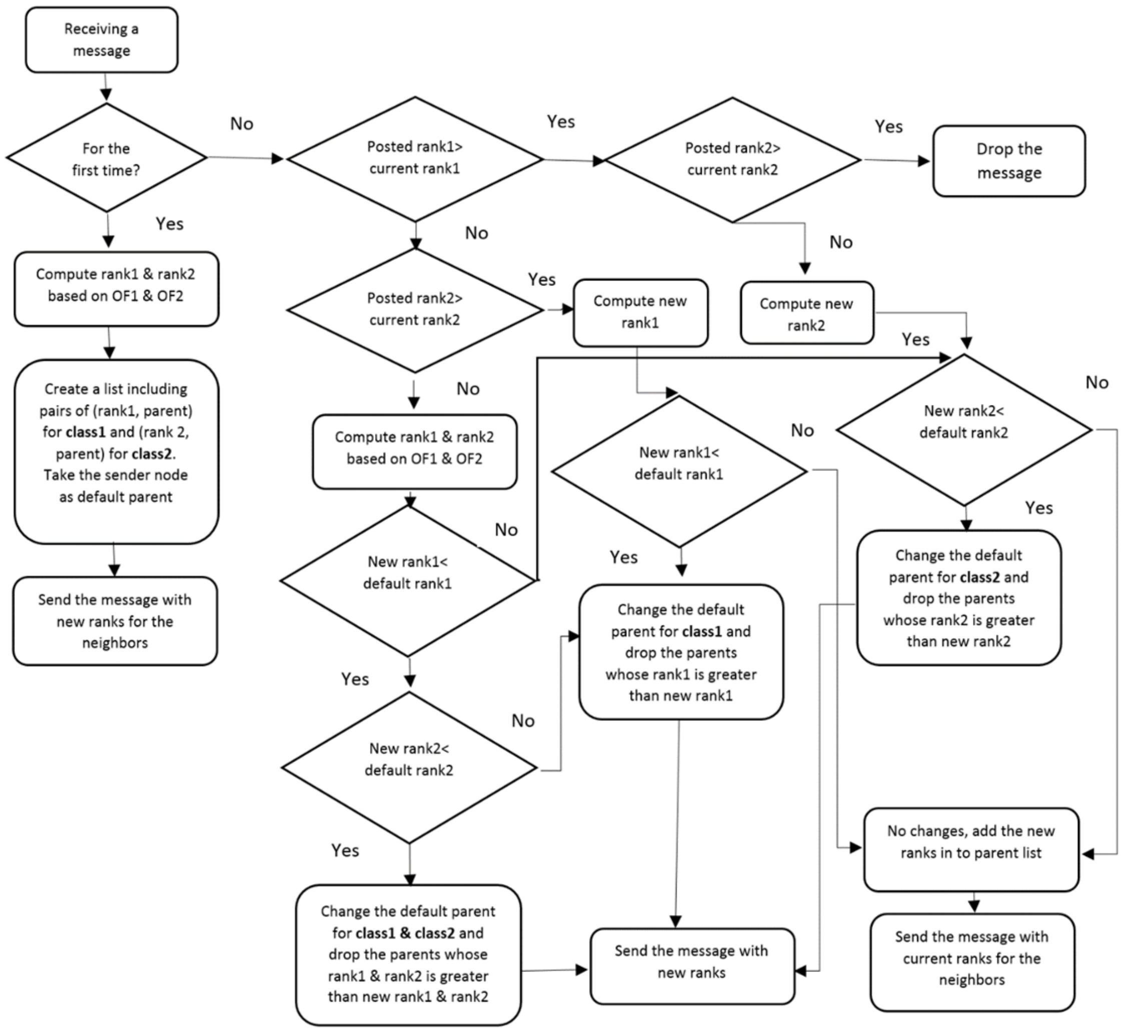

Receiving a message
For the first time?
No
Yes
Compute rank1 & rank2 based on OF1 & OF2
Create a list including pairs of (rank1, parent) for **class1** and (rank 2, parent) for **class2**. Take the sender node as default parent
Send the message with new ranks for the neighbors
Posted rank1> current rank1
Yes
No
Posted rank2> current rank2
Yes
No
Drop the message
Posted rank2> current rank2
Yes
No
Compute new rank1
Compute new rank2
Compute rank1 & rank2 based on OF1 & OF2
New rank1< default rank1
No
Yes
New rank2< default rank2
No
Yes
Change the default parent for **class1 & class2** and drop the parents whose rank1 & rank2 is greater than new rank1 & rank2
New rank1< default rank1
No
Yes
Change the default parent for **class1** and drop the parents whose rank1 is greater than new rank1
Send the message with new ranks
New rank2< default rank2
Yes
No
Change the default parent for **class2** and drop the parents whose rank2 is greater than new rank2
No changes, add the new ranks in to parent list
Send the message with current ranks for the neighbors

Fig. 3. OMC-RPL Algorithm for Constructing DODAG with Two Classes

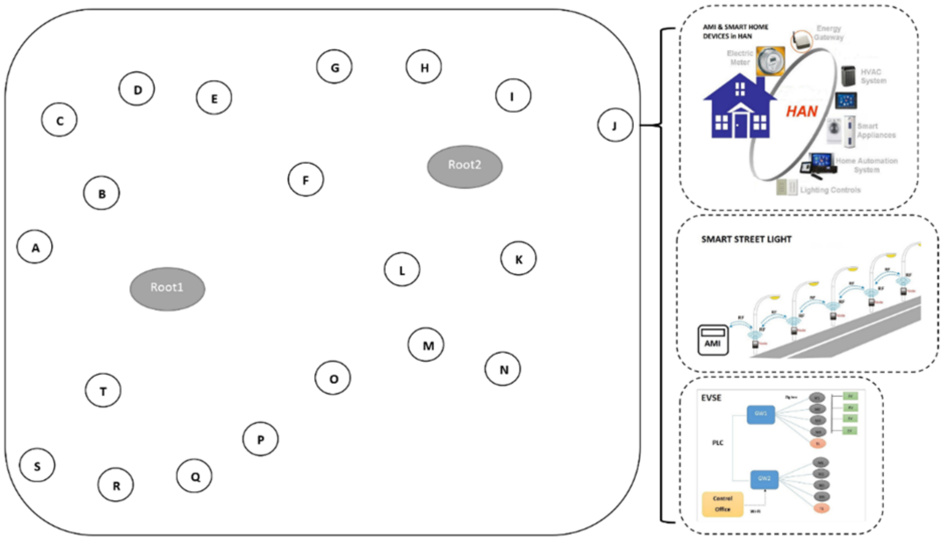


Fig. 4. A Sample Network of Nodes of Various Applications in HAN and NAN

Various steps taken by the algorithm in order to select the preferred parents are as follow:

The receiving node checks the posted rank for each class, and if it is not greater than the current rank, then it is reasonable to find new parents otherwise it drops the message.

The receiving node computes the new ranks based on the fresh information to see if it is really from a better suited parent; if so it updates the pairs of (rank$_i$ and parent) in its list and removes the parents with greater rank. To clarify the concern of overhead, we should be careful that there are not separate routing tables, we just keep different ranks and preferred parents for each node.

When the DODAG is constructed, the upward route is clear because the only routing process for the intermediate nodes is to send their packets based on their class to the preferred parents, which is chosen during the DODAG's construction procedure.

We provide an example to better describe the proposed algorithm. Fig. 4 shows a sample network of twenty nodes and two roots in HAN and NAN. Each node may be related to different applications such as DR (Demand Response), AMI (Advanced Metering Infrastructure), Smart Streetlight, Smart Home Devices, EVSE (Electric Vehicle Supply Equipment), etc. with various QoS requirements. The root could be a concentrator in a station. After performing the two classes OMC-RPL algorithm, a DODAG as shown in Fig. 5 is created. It is obvious that this exemplary DODAG is formed based on a specific OF. If we change the objective function, we willprobably face another DODAG. As mentioned earlier, according to two classes of data, each node has two preferred parents (the parents could be different from each other), and also each node could have none or several reserved parents.

Algorithm 1 is the generalized OMC-RPL algorithm for n classes of data. Note that the messages in OMC-RPL are standard IPv6 message. These messages are modified easily by adding some fields for extra ranks.

Now, the challenge is designing n objective functions for each class of data. According to [22] a good route should be real-time (low end-to-end delay), reliable (high delivery ratio) and energy efficient. Therefore, our goal is to propose comprehensive objective function that satisfies these properties. Instead of having n objective functions, we suggest weighting parameters that make the difference for each class of data based on its requirements. Three main components of our proposed objective function are: the quality of the node, the quality of the link (henceforth, we name them NR (Node Rank), LR (Link Rank) respectively) and the energy efficiency, that we evaluate by Remaining Energy (RE) in each node in percent. It is obvious that the RE is meaningful for the nodes that are supplied by battery and have the energy concern; for the other nodes, we consider the RE equal to one.

The proposed objective function is given by Equation 1:

$$\mathrm{R_n(i)} = \mathrm{R_n(p)} + \frac{(\alpha_\mathrm{n}\mathrm{NR} + \beta_\mathrm{n}\mathrm{LR})}{\mathrm{RE}} + 1 \qquad (1)$$

Table 1 shows all the parameters and their definitions, which are used in the equations. According to the proposed objective function, the rank of each node for each class of data is the sum of its parent's rank in the same class, plus the ratio of link and node quality (the NR/LR coefficients are changed equivalently to the class type) to RE, plus one hop count. The weighting parameters are used to control the effectiveness of NR and LR based on the class type.

Distinctive types of traffics and five classes of data related to LLN are presented in [23]. Each data class faces with different QoS requirements. This issue is satisfied by changing the weighting coefficients. In fact, the data classification scale is the amount of allowed delay times for various types of applications. Packets with very low allowance of a delay, belong to critical, real-time and high-priority classes. Assume a spectrum of applications from real-time to non-real-time, that can be divided into several classes; the first class is the most critical, and that last one is the most unimportant.

When there are real-time packets, the values of weighting parameters should be selected in a way that the objective function offers the best path (appropriate nodes and links) towards the root. The qualities of node and link are determined by the proposed equations 2-6. Equations 2, 4 and 5 compute the node quality by multiply the ratio of service rate to arrival rate and the ratio of queue length to buffer length. Equation 3 is used to keep the history of node quality. This parameter is calculated, using the current and old values of NR. Equation 6 is the ETX, that we use it for the link quality.

$$NR = \rho \times \omega \qquad (2)$$

$$\rho.\omega = (1-\gamma)(\rho.\omega)_{old} + \gamma(\rho.\omega)_{current} \qquad (3)$$

$$\rho = {}^{\lambda}/_{\mu} \qquad (4)$$

$$\omega = {}^{Q}/_{L} \qquad (5)$$

$$LR_{(i,j)} = {}^{m}/_{s} \qquad (6)$$

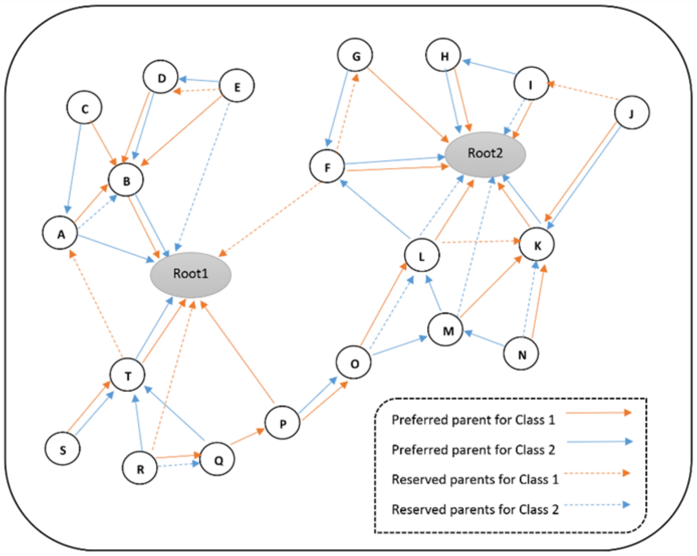

Root1
Root2
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
Preferred parent for Class 1
Preferred parent for Class 2
Reserved parents for Class 1
Reserved parents for Class 2

Fig. 5. An exemplary DODAG after Using OMC-RPL Based on Two Classes of Data

Algorithm 1: OMC-RPL Algorithm Based on *n* Classes

```
Input:
n classes of data
Objective Function {$OF_1$, $OF_2$ … $OF_n$}
Output:
List of {$rank_1$- parent, $rank_2$-parent … $rank_n$-parent} for each node
Method:
(1) Repeat
(2) If (receiving a message for the first time)
        {
        for (i = 0 to n)
            {
            Compute $Rank_i$ based on $OF_i$
            Create list ($rank_i$, parent)
            }
        Send message with new rank
        }
(3) else
        for (i = 0 to n)
        {
        if (posted $rank_i$ < current $rank_i$)
            Compute new $rank_i$
        if (new $rank_i$ < default $rank_i$)
            {
            Change the default parent for class i
            Drop the parents whose $rank_i$ is greater than new $rank_i$
            }
(4) until all nodes attach to DODAG
```

Table 1: Definition of parameters used in objective function.

Parameters	Definition
$\boldsymbol{R_n(i)}$	rank of node i for $n'th$ class
$\boldsymbol{R_n(p)}$	the rank of preferred parent for $n'th$ class
$\boldsymbol{\alpha_n}$	Weighting coefficient for $n'th$ class between (0-1)
$\boldsymbol{\beta_n}$	Weighting coefficient for $n'th$ class between (0-1)
$\boldsymbol{NR}$	The amount of this parameter shows the node quality
$\boldsymbol{LR}$	The amount of this parameter shows the link quality between two nodes
$\boldsymbol{RE}$	Remaining Energy in percent
$\boldsymbol{\gamma}$	Coefficient between (0-1)
$\boldsymbol{\lambda}$	Arrival rate
$\boldsymbol{\mu}$	Service rate
$\boldsymbol{Q}$	Queue length
$\boldsymbol{L}$	Buffer length
$\boldsymbol{m}$	data packets transmitted from node i to node j
$\boldsymbol{s}$	Number of successful network layer transmission

In order to find the best values for the weighting coefficients, we use the PSO (Particle Swarm Optimization) algorithm. PSO is a population-based algorithm that was introduced by Kennedy and Eberhart in 1995. It is based on the social behavior of a swarm of birds and fishes in search of for food [24]. This algorithm is an appropriate solution for a large-scale non-convex optimization problem. Searching rules in this algorithm are easy and yet meaningful, the computation time is low and there is no need for much memory space; these are the reasons why it has been used by many applications to solve several problems. The procedure of PSO algorithm in finding optimal values follows the animal behavior through using the best personal/global experience of particles. PSO consists of a swarm of particles, where each particle represents a potential solution [25]. Although recently, several modifications have been made to the original PSO, but the main idea of PSO asserts that position of the particle toward the optimized answer is influenced by a velocity vector. Let $x_i(t)$ denote the position of particle i in the search space at time step t (denotes discrete time steps). According to equation 7, the new position of the particle is obtained by adding a velocity $v_i(t)$ to the current position. The velocity is calculated based on equation 8 which is the outcome vector of previous velocity, local best and global best values [26].

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{7}$$

$$\begin{aligned} v_i(t) = v_i(t-1) + \ & c_1 r_1\big(localbest(t) - x_i(t-1)\big) \\ & + c_2 r_2\big(globalbest(t) - x_i(t-1)\big) \end{aligned} \tag{8}$$

In equation 8, $localbest(t)$ and $globalbest(t)$ respectively show the best personal and the best neighborhood experience of particles in time slot t, $c_1 and\, c_2$ are acceleration coefficient and $r_1 and\, r_2$ are random numbers between 0 and 1. After a certain number of repetitions, the algorithm will find the optimized answers in the search space.

Despite the non-real-time classes, the critical and sensitive classes of data need a fast path; with regard to this issue, the NR coefficient is considered greater than the LR and for the less important classes, it happens vice versa. This classification idea causes the spread of traffic through the network. This leads to congestion prevention and an increase in the network lifetime . Eventually, the PSO procedure of finding the optimized values for weighting parameters is as follows: the required initial values are determined; the random weighting coefficient values are selected for the particles; the PSO objective function which we consider as the average end-to-end delay is implemented; then the best personal and global experiences are identified; the algorithm is repeated based on equation 7 to find the optimized values. According to the aforementioned, and in order to have distinctive weighting coefficients for each class of data, The ranges for the coefficients, equivalent to the number of classes is specified in Fig. 6. For example, in a case that there are two classes of data, the optimized NR coefficient for the first class should be found between 0.5 and 1, and the optimized LR coefficient should be found between 0 and 0.5. In the next section using the simulation, the performance of the proposed method is evaluated.

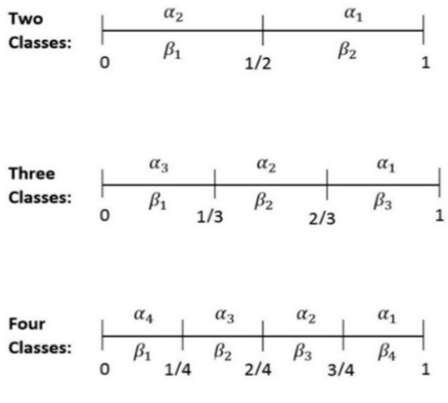


Fig. 6. Specified Range of Weighting Coefficient for Two, Three and Four Classes of Data

4. PERFORMANCE EVALUATION:

In order to evaluate the OMC-RPL, we create a sample network with several nodes and sinks. This network generates different traffic types related to the various applications and classes. Furthermore, to support distinctive applications, some of the nodes have energy concern and work with a battery. Table 2 shows the general information of simulation environment.

Table 2. The General Information of Simulation Environment

Simulator	Riverbed 18.0
Simulation area	250×250m
Number of nodes	88
Number of sinks	3
Number of nodes work with battery	7
Initial node energy for battery base nodes	50 joule
Test duration for each scenario	1 Hour
Traffic patterns	Is defined based on the classes and specifications in [23]

Before evaluating the proposed method, we need to find the optimized values for weighting coefficients. In the current study, we consider three different cases, including two, three and four classes of data; with that in mind, the described PSO is implemented inside the simulation software, and we need to run the algorithm for each class. If we have two classes, then we should run the algorithm two times. The achieved values of parameters at different cases is shown in table 3. For example, in the case of two classes of data and by using the achieved values, the objective functions for the first and second class are given in equations 9 and 10, respectively. Now the OMC-RPL is able to support different classes and has several OFs corresponding to the number of classes.

$$R_1(i) = R_1(p) + \frac{(0.87 \times NR + 0.33 \times LR)}{RE} + 1 \qquad (9)$$

$$R_2(i) = R_2(p) + \frac{(0.39 \times NR + 0.76 \times LR)}{RE} + 1 \qquad (10)$$

Table 3. The Values of Weighting Coefficients

Two Classes	$\alpha_1 = 0.87$	$\beta_1 = 0.33$
	$\alpha_2 = 0.39$	$\beta_2 = 0.76$
Three Classes	$\alpha_1 = 0.90$	$\beta_1 = 0.13$
	$\alpha_2 = 0.54$	$\beta_2 = 0.43$
	$\alpha_3 = 0.16$	$\beta_3 = 0.88$
Four Classes	$\alpha_1 = 0.81$	$\beta_1 = 0.18$
	$\alpha_2 = 0.66$	$\beta_2 = 0.42$
	$\alpha_3 = 0.37$	$\beta_3 = 0.70$
	$\alpha_4 = 0.22$	$\beta_4 = 0.91$

In the following, we evaluate the performance of OMC-RPL in distinctive scenarios. In all scenarios, four different cases, including OMC-RPL with two, three and four classes of data (which henceforth we call case A, case B and case C, respectively) and ordinary RPL that is based on ETX, are used for comparison. In the first scenario, we compare the end-to-end delay of these cases during the simulation time; the results of which can be seen in

Fig. *7*. It shows that the classification idea outperforms the RPL with single OF. Although in cases B and C, the results are almost the same, but both of them act better than case A. It seems that the diversity of applications are in a way that there is no difference between three and four classes of data.

In the second scenario, we again investigate the end-to-end delay, but this time some nodes are congested randomly. This scenario is illustrated in Fig. 8. The peaks in regular RPL show the congestions, and the results demonstrate that in all cases OMC-RPL acts better. In OMC-RPL the process of DODAG construction is continuously repeated and upon any changes in nodes and links, ranks are modified and a new DODAG is created, while in ordinary RPL, any changes in nodes are not effective and lead to the increase of the drop rate in case of congestions. That is why at these moments the end-to-end delay increases slightly in OMC-RPL and rises sharply in ordinary RPL.

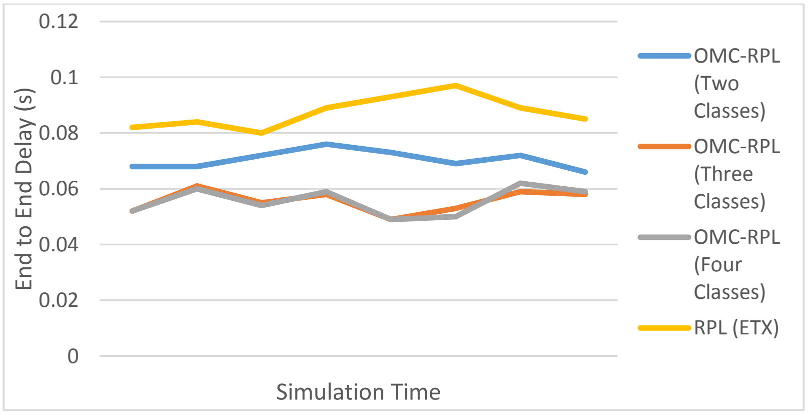


Fig. 7. End to End Delay During the Simulation Time (All Nodes Work Normally)

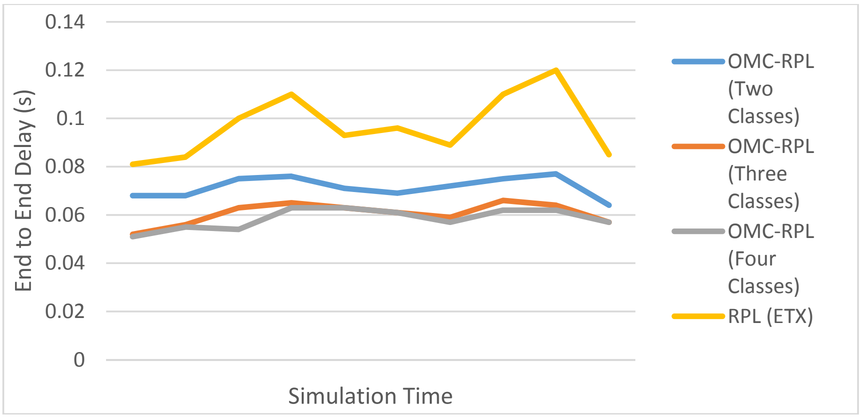


Fig. 8. End to End Delay During the Simulation Time (Some Nodes Are Congested)

Fig. *9* is about the third scenario which depicts the end-to-end delay in a situation that some nodes fail. When failure happens in ordinary RPL (ETX), the end-to-end delay increase significantly while this growth is negligible in each OMC-RPL case. We know that when a receiving node fails, the algorithm easily uses the reserved parents. The variation of end-to-end delay in cases B and C are more than the previous scenarios, then we can conclude that in the case of node failure, more classes of data could act better.

The average node's queue size is a QoS metric. For the fourth scenario, we choose four sample nodes.

Fig. *10* shows the average queue size in the selected nodes during the simulation. Looking at the chart, it is obvious that when using RPL (ETX) a node like node 2 is too busy but conversely, node 1 is rarely selected as the preferred parent and is idle. The average queue size in each case of OMC-RPL does not exceed more than half the capacity, in fact, OMC-RPL by using a holistic OF and various parameters balance the traffic load in all the nodes.

The goal of the fifth scenario is the assessment of energy consumption in the nodes that work with battery. In this scenario, we choose three of the seven available nodes that have the energy concern; two of these nodes are common with the previous scenario (node 1 and 2). Fig. 11 shows the remaining energy of nodes in percentages. The leftover energy in node 1 is around 90 percent, which means that this node is not used very often, unlike node 2 that lost its energy completely. The achieved results from both the last scenarios prove that the ordinary RPL is not able to find the appropriate paths due to the lack of proper objective function; therefore, some nodes are used immensely while others remain unused . The results for any case of OMC-RPL show that the remaining energies in nodes are acceptable and OMC-RPL can help to increase the network lifetime. Among different instances of the proposed method, case C outperforms in terms of energy consumption.

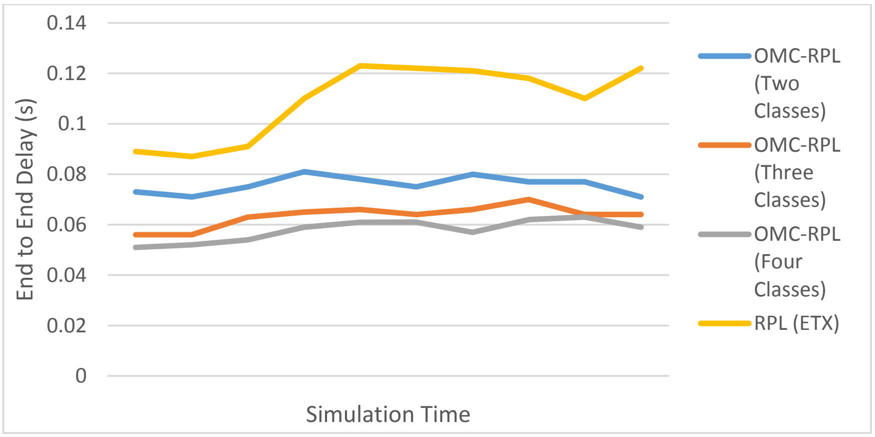


Fig. 9. End to End Delay During the Simulation Time (Some Nodes Are Failed)

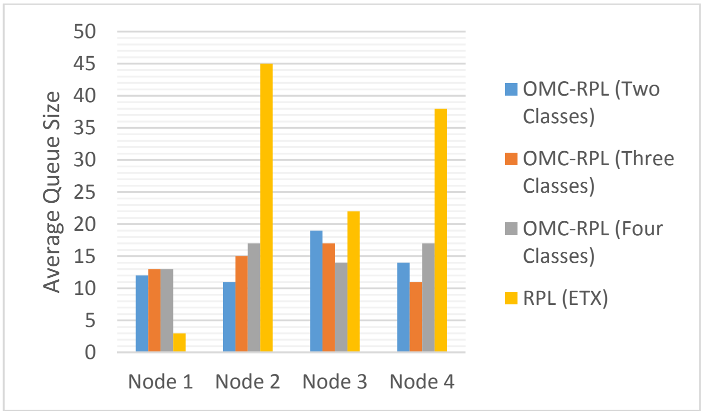


Fig. 10: Average Queue Size for Four Sample Nodes

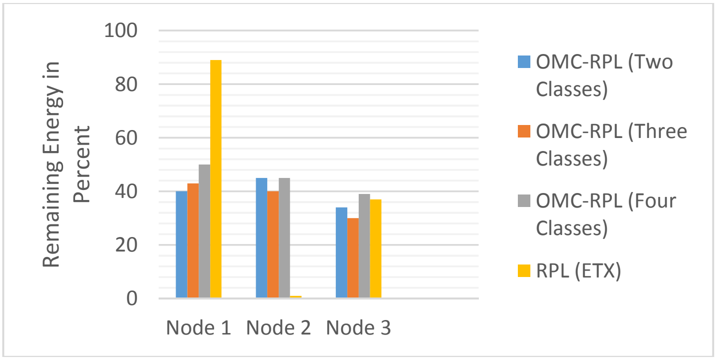


Fig. 11: Remaining Energy in Percent for Three Sample Nodes that Work with Battery

5. Conclusion

As mentioned earlier, recently RPL protocol has become one of the main solutions in the smart grid. Lots of researches have been done on this issue. In this paper by studying the shortcomings and challenges of RPL, we present a modified version of RPL with the approach of QoS. Since data classification is the main requirement of providing QoS, the proposed method with comprehensive objective function regarding the QoS metrics is able to support multi-classes of data; in this regard the PSO optimization algorithm is used to find the best values of coefficients used in OF. OMC-RPL with the different number of classes and various scenarios was simulated; the results in comparison with ordinary RPL are significant. Using OMC-RPL with any number of classes leads to decrease end-to-end delay, balance the traffic load in the network and increase the lifetime of battery supplied nodes. Although the results for three and four classes of data are very close, but in some scenarios, we still experience better outcomes for four classes of data. Thereupon using the idea of multi-class RPL can play an important role in the smart grid and can be used as an alternative solution. As future work, we can propose new OF with different metrics and approaches for specific cases; we can investigate about the stability of the network and also the optimized number of classes can be studied.

References

[1] Gungor, V.C., et al., A survey on smart grid potential applications and communication requirements. Industrial Informatics, IEEE Transactions on, 2013. 9(1): p. 28-42.

[2] Ho, Q.-D., Y. Gao, and T. Le-Ngoc, Challenges and research opportunities in wireless communication networks for the smart grid. Wireless Communications, IEEE, 2013. 20(3): p. 89-95.

[3] Rajalingham, G., et al. Quality of service differentiation for smart grid neighbor area networks through multiple RPL instances. in Proceedings of the 10th ACM symposium on QoS and security for wireless and mobile networks. 2014. ACM.

[4] Thai, P., Dissertation/Thesis Approval Form. 2011.

[5] Winter, T., RPL: IPv6 routing protocol for low-power and lossy networks. 2012.

[6] Vasseur, J., Terminology in low power and lossy networks. Work in Progress, 2011.

[7] Gaddour, O. and A. Koubâa, RPL in a nutshell: A survey. Computer Networks, 2012. 56(14): p. 3163-3178.

[8] Long, N.T., et al. QoS-aware cross-layer mechanism for multiple instances RPL. in Advanced Technologies for Communications (ATC), 2013 International Conference on. 2013. IEEE.

[9] Abdessalem, R.B. and N. Tabbane. RPL-SCSP: A Network-MAC Cross-Layer Design for Wireless Sensor Networks. in Proceedings of Ninth

International Conference on Wireless Communication and Sensor Networks. 2014. Springer.

[10] Thubert, P., Objective function zero for the routing protocol for low-power and lossy networks (RPL). 2012.

[11] Gnawali, O. and P. Levis, The ETX Objective Function for RPL. 2010.

[12] Vasseur, J.-P., et al., Routing metrics used for path calculation in low-power and lossy networks. 2012.

[13] Gaddour, O., et al. Simulation and performance evaluation of DAG construction with RPL. in Communications and Networking (ComNet), 2012 Third International Conference on. 2012. IEEE.

[14] Wang, D., et al. RPL based routing for advanced metering infrastructure in smart grid. in Communications Workshops (ICC), 2010 IEEE International Conference on. 2010. IEEE.

[15] Jeon, Y.-H., QoS requirements for the smart grid communications system. International Journal of Computer Science and Network Security, 2011. 11(3): p. 86-94.

[16] Deshpande, J.G., E. Kim, and M. Thottan, Differentiated services QoS in smart grid communication networks. Bell Labs Technical Journal, 2011. 16(3): p. 61-81.

[17] Sun, W., et al. Quality of service networking for smart grid distribution monitoring. in Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on. 2010. IEEE.

[18] Gharavi, H. and C. Xu, Traffic scheduling technique for smart grid advanced metering applications. Communications, IEEE Transactions on, 2012. 60(6): p. 1646-1658.

[19] Di Marco, P., et al. MAC-aware routing metrics for low power and lossy networks. in INFOCOM, 2013 Proceedings IEEE. 2013. IEEE.

[20] Brachman, A., Rpl objective function impact on llns topology and performance, in Internet of Things, Smart Spaces, and Next Generation Networking. 2013, Springer. p. 340-351.

[21] Karkazis, P., et al. Design of primary and composite routing metrics for rpl-compliant wireless sensor networks. in Telecommunications and Multimedia (TEMU), 2012 International Conference on. 2012. IEEE.

[22] Gaddour, O., et al. OF-FL: QoS-aware fuzzy logic objective function for the RPL routing protocol. in Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), 2014 12th International Symposium on. 2014. IEEE.

[23] Shah, S. and P. Thubert, Differentiated Service Class Recommendations for LLN Traffic. 2012.

[24] Kennedy, J. and R. Eberhart. Particle swarm optimization. in Neural Networks, 1995. Proceedings., IEEE International Conference on. 1995.

[25] Rini, D.P., S.M. Shamsuddin, and S.S. Yuhaniz, Particle swarm optimization: technique, system and challenges. International Journal of Computer Applications, 2011. 14(1): p. 19-26.

[26] Engelbrecht, A.P., Fundamentals of computational swarm intelligence. 2006: John Wiley & Sons.

Journal of Computer and Knowledge Engineering, Vol. 1, No. 1
DOI: 10.22067/cke.v1i2.53554

Scheduling Data-Driven Workflows in Multi-Cloud Environment

Nafise Sooezi[+], Saeid Abrishami, Majid Lotfian

Abstract. Nowadays, cloud computing and other distributed computing systems have been developed to support various types of workflows in applications. Due to the restrictions onthe use ofone cloud provider, the concept of multiple clouds as been proposed.Inmultipleclouds, schedulingworkflowswithlarge amounts ofdata is a well-knownNP-Hard problem. The existing scheduling algorithms have not paid attention to the data dependency issues and their importance in scheduling criteria such as time and cost. In this paper, we propose a communication-based algorithm for workflows with huge volumes of data in a multi-cloud environment. The proposed algorithm changes the definition of the Partial Critical Paths(PCP) to minimize the cost of workflow executionwhile meeting a user defined deadline.

Keywords: Cloud Computing,Multi-cloud,Workflow Scheduling,Data Dependency, Communication.

1. Introduction

Multiple clouds have a special place in the modern-day models we use. An important reason for this is the increased use of clouds in recent years. One of the important features of clouds is the illusion of unlimited resources to cloud users. The number of users varies at different times of the day during the weeks or on weekends. If the providers upgrade their resources so as to meet the peak demand of users, these resources will remain partially unused during non-peak hours. However, providers can shut down unused nodes in order to eliminate the cost of maintenance of the equipment and resources; however, they still have to pay for the cost of buying and equipping these unused resources. To offset these costs, providers are forced to increase their prices, and this has resulted in the poor competition between cloud service markets. On the other hand, if the provider only supplies the needed resources to users in the average demand time, then it cannot provide service at peak time demands and this will lower the reliability of the provider and it will result in a reduction of the number of users of its services. Today, a cloud alone cannot meet the needs of users at all times and it is becoming more important to provide service using multiple clouds. Sharing resources between several providers might be the best solution to the problem.

In [1], more than 25 types of multiple clouds have been introduced among which multi-cloud and cloud federation are a few examples. In cloud federation, cloud providers will agree to share resources, which help them improve services to the users. However, since the cloud technology is in its early stages and there is no overall standard, the agreement between cloud providers is difficult due to the fact that each provider uses their own interface and protocol. Multi-clouds a type of multiple cloud system in which there is no agreement between the cloud providers, and a third party is responsible for the relationships, the dialogues, and monitoring the providers. Scheduling in a cloud environment has been one of the major challenges in the world of clouds.

Workflow is a collection of interconnected multiple tasks that must be performed in a specific order. Workflow structure indicates temporary dependencies between tasks [2]. Workflow scheduling is the problem of mapping each task to a suitable resource and of ordering the tasks on each resource to satisfy some performance criterion. This is an NP-Hard problem, so there is no known polynomial algorithm for it. In general, Multi-Criteria scheduling problems are very difficult to solve even in the single cloud case. Workflow scheduling is facing more challenges in the multi-cloud environment due to the increasing number of complex factors. One of the major problems facing the proposed scheduling algorithms in multiple clouds is the lack of attention to communication in workflow and its effect on the cost and execution time. The data transfer rate between the samples of a cloud is very high, (e.g. bandwidth between different samples in the Amazon EC2 is approximately between 300 Mbps to 4 Gbps) and this transfer is free, while the received speed of the Amazon cloud (inter-cloud speed) is between 400 Mbps to 20 Mbps and send speed is between 20Mbps and 80 Mbps[3-5]. In the proposed solution, we have tried to pay attention to minimize the cost and time due to it. In the proposed approach, we use the concept of Partial Critical Paths(PCP) introduced by Abrishami et al.[6]. In this way, we have changed the definition of the critical path and the amount of communication between the tasks to be included in this definition.

The rest of the paper is organized as follows. Section 2 presents related work. In section 3, we describe the generic application, objective, and platform models underneath our approach. Section 4 shows the proposed algorithm and scheduling policies. Section 5 presents and discusses the results. Finally, in Section 6 we present our conclusions and future work.

2. Related Work

So far, many algorithms have been proposed for workflow scheduling in a single cloud including: [7], [8], [6], [9], [10] and [11].The authors of most of these works

Manuscript received February 7, 2016; revised July 18, 2016; accepted November 24, 2016.
Nafise Sooezi and Saeid Abrishami, Department of Computer Engineering, Engineering Faculty, Ferdowsi University of Mashhad, Mashhad, Iran.
Majid Lotfian, Department of Computer Engineering, Engineering Faculty, Islamic Azad University of Mashhad, Mashhad, Iran.
[+]The corresponding author's e-mail is: nafise.sooezi@stu.um.ac.ir

have considered execution time and cost as their objectives. Ever since the concept of multi-cloud has been introduced in recent years, there have been a few algorithms in this field.

In 2007, Sakellariou et al. [12] presented an algorithm known as IC-LOSS. The IC-LOSS algorithm tries to minimize the execution time under a budget constraint. This algorithm consists of two phases: first, it tries to find an initial schedule for the input workflow with minimum execution time, and then it refines the initial schedule until its budget constraint is satisfied. In the first phase, it uses a well-known makespan minimization algorithm, called HEFT [13]. The HEFT algorithm for each task is looking for a version that has the earliest completion time for the task. The second part deals with the correction of the scheduling with the allocation of tasks to cheaper resources until the budget constraints are fulfilled:

LossWeight $(T_I,J) = \frac{\text{Tnew} - \text{Told}}{\text{Cold} - \text{Cnew}}$

In 2010, Van den Bossche et al.[14]solved the problem of multiple applications scheduling in several cloud providers using linear programming. The objective function in this algorithm is minimizing the total cost of data traffic and computational cost over all time slots within the schedule for all providers. The lack of consideration of the communication time between the clouds is one of the problems in this paper.

In 2011, Houidi et al. [15] presented an algorithm that aims to break several requests between cloud providers such that the user costs (total cost of each resource and the cost of communication) are minimal. They distributed the requests with the broker between the cloud providers. The broker is composed of three main components:

1. Cloud Request Splitting
2. Resource Provisioning
3. Inter-Cloud Network Provisioning.

After formulating the problem, they have solved it by using linear programming.

In 2011, Li et al. [16] presented an algorithm that aims to maximize capacity and minimize costs in accordance with the new conditions of the providers. They have also tried to minimize the overhead of scheduling under the new conditions as compared with the previous condition if some changes are made in the environment.

In 2013, Fard et al. [17] presented a method to prevent the selfish behavior of providers that use an auction pricing model instead of the pay as you go model. In this way, each task announces to the resources its workload (communicating with other tasks and the required input-output). The source suggests an approach to tasks. In this way, the solution is chosen so that the product of time and its cost is minimal. After winner resource is selected, if the time proposed by the source is greater than or equal to real-time, the cost of the provider is fully paid and if the time proposed by the source is less than real time, the resource is penalized using a given function. In this method, the Nash equilibrium is used that is a fundamental concept of the theory of games.

In 2012, Fardet al. [18] presented an algorithm that is one of the complete algorithms introduced in this field. The algorithm makes use of user-defined constraints about time, cost, power consumption, and reliability and then it estimates the optimal solution. In this paper, all the objectives are modeled. Then the algorithm approximates the optimum solution during threephases. In the first phase, it estimates the objectives' sub-constraints for each individual task using the user constraint vector. In the second phase, it assigns a rank to each task of the workflow and sorts them in an ascending order. Finally, in the third phase, the algorithm attempts to allocate the most appropriate resource to each activity with due consideration given to the estimated sub-constraints. A major problem with this algorithm is that it does nothing to improve communication. As was mentioned earlier in this paper, inter-cloud communication is one of the most important issues in the scheduling workflow in multi-cloud systems. Lack of attention to this point has affected the whole algorithm, and it is particularly inappropriate for communication-based workflows. The assumption of unlimited resources is another problem in this algorithm.

Duan et al. [19] offered a good algorithm in 2014. In their paper, time and cost are considered based on the limitations of communication bandwidth and storage space. One of the differences between this paper and the previous one [18]is that this paper considers two objectives and two conditions instead of four objectives. The other difference is that this algorithm has a lower time complexity as compared with [18]. In this paper, the problem is modeled with game theory. The algorithm is repeated as many times as needed by one condition and the nearly optimal solution is found. One of the advantages of this algorithms fasts convergence by using the information about the environment and the competitors. And the other advantage is that you can easily add a new objective to the problem. One of the major problems of this algorithm is that it is not suitable for applications with a high level of complexity just like the algorithm presented in the previous paper. Other problems can also be mentioned such as the following:

1. Initialization of the weight vector is done by the algorithm itself and this can lead to different results.
2. Tasks are broken vertically to transfer parallel tasks to one provider that cannot be useful because there is no data to transfer between them (The cost and time of data transmission within a provider are not comparable to the inter providers).

In 2014, Montes et al. [20] proposed an algorithm that allows execution of dynamic workflows in a multi-cloud environment. In addition, there is an ability to customize the scheduler for the user. One of the policies that provide this ability operates as follow: it assigns instances to each task so that the total execution time of tasks, task data receiving time to the desired instance, and the estimated time needed to perform the next task, is minimized. Another policy is based on a deadline that selects a minimum set of the resources that are needed to complete all tasks such as deadlines are met and the objective function is satisfied. They have considered four objective functions: performance optimization, data locality optimization, performance and data optimization, and cost optimization. One of the major problems of this algorithm is the lack of attention to the communication problems costs. As the article mentioned, communication has a very great impact on the cost and time and that should be focused on. One of the other problems is considering the workflow dynamically and separating its steps effectively.

Table 1. Comparison of different scheduling algorithms in a cloud environment

Paper	*Environment*	*Scheduling type*	*Method*	*Objectives*	*Advantages*	*Disadvantages*
BIP [14] 2010	Hybrid cloud and Multi-cloud	static	Mathematical model	• Minimize cost	-	• The lack of attention to communication and to be placed tasks in a single provider • Unsuitable for communication-based applications
[15] 2011	Multiple cloud	static	Mathematical model	• Minimize cost	•consider tasks placed in a single provider	• The lack of attention to communication and execution time and user deadline
[16] 2011	Multiple cloud	dynamic	Mathematical model	• Maximize the use of capacity • Minimize cost in new conditions	• Dynamic considering the price of instances, the type of instances, and the performance of the services	• The lack of attention to be placed tasks in a single provider and communication in workflows
[17] 2013	Multi-cloud	dynamic	Mathematical Model	• Minimize cost • Minimize execution time	• harness the selfish behavior of cloud providers	• The lack of attention to be placed tasks in a single provider and communication in workflows
[19] 2014	Hybrid cloud	static	Mathematical Model	• Minimize cost & time while fulfilling network bandwidth and storage requirements	• fast convergence by using competitors and environment information	• unsuitable for applications with the complex dependencies between tasks • initialize the weight vector by the algorithm itself • break tasks vertically
MOLS [18] 2012	Set of heterogeneous resources	static	Heuristic	• Minimize cost, time, and energy consumption • Maximize reliability	• low time complexity	• assuming resources are unlimited • The lack of attention to communication between the clouds
[20] 2014	Multi-cloud	Dynamic	Heuristic	• Minimize cost to satisfy the objective function and user deadline	• allows users to customize scheduling policies	• The lack of attention to communication and cost at the same time • high time complexity

3. The Model

3.1. The Application Model

Workflow is described by a Directed Acyclic Graph (DAG) in which each computational task is represented by a node, and each data or control dependency between tasks is represented by a directed edge between the corresponding nodes. W=(T,E) consists of a set of *n* tasks: T= $\bigcup_{i=1}^{n} T_i$ interconnected through a set of control flow and data flow dependencies: E=$\{(T_i, T_j, Data_{ij})|(T_i, T_j) \in T \times T\}$ As $Data_{ij}$ shows the amount of data to be exchanged between T_i and T_j.

We always add two dummy tasks T_{entry} and T_{exit} to the beginning and the end of the workflow, respectively. These dummy tasks have zero execution time and they are connected with zero-weight dependencies to the actual entry and exit tasks.

3.2. The Platform Model

A multi-cloud environment includes N providers: P_1, P_2,…,P_N. Each provider has certain characteristics that are shown by a property vector (B_{up}, B_{down}, C_{in}, C_{out}, $B_{internal}$, I), which include (in order) upload/download bandwidth, incoming/outgoing data transfer costs, internal bandwidth and set of provider's instances (I_{1i},I_{2i}, … I_{mi}). Each instance mi, has certain characteristics that are shown by a property vector I_{mi}=(V_{mi},C_{mi}), which include (in order) computational speed of the instance I_{mi} in millions of instructions per second (MIPS) and cost of instance *mi*.

3.3. The Objective Model

We want to schedule workflow so that the execution costs are minimized and user deadlines are satisfied. Time and cost have been formulated according to [18]. The execution time of task T_j on the instance I_{mi}can be computed as the sum of the longest input transfer time T_j (from all inputs to T_j) and the task computation time:

$$ET(T_j,I_{mi})= \text{Max}_{\ T_P\in pred(T_j)} \{\frac{data_{pj}}{B_{up}(mi)}\}+ \frac{work(T_j)}{V_{mi}} \quad (1)$$

Where$B_{up}(mi)$is the bandwidth between task$T_j$and T_p. The completion time or makespan of a task $T_j$can be recursively computed as follows:

$$ET_{final}(T_j,I_{mi})= \begin{cases} ET(T_j, I_{mi}) & pred(T_j) = \emptyset \\ \text{Max}_{\ T_p\in pred(T_j)} \{ET_{final}(T_p, sched(T_p)) + ET(T_j, I_{mi})\} & pred(T_j) \neq \emptyset \end{cases} \quad (2)$$

Consequently, the workflow makespan is given by the longest completion time of its tasks:

$$Total_{ET}(workflow)=Max_{j\in[1…n]}\{ET_{final}(T_j,sched(T_j))\} \quad (3)$$

The cost of task $T_j$in the instance mi is thesum of the computation and data transfer costs:

$C(T_j,I_{mi})$
$=ET(T_j,I_{mi})*C_{mi}+Input(T_j)*C_{in_{mi}}+Output(T_j)*C_{out_{mi}}$(4)
where $C_{mi}$is the computational cost in instance mi, $C_{in_{mi}}$and $C_{out_{mi}}$is the incoming and outgoing data transfer cost of instance I_{mi}'s cloud. Input (T_j) and Output (T_j) are the total amount of input and output data of node T_j that are received from tasks that have been scheduled in instances other than cloud of instance I_{mi}(because data transfer cost between tasks is zero if you have two tasks on the same instance or on instances that are available on one cloud). The execution costs for a workflow is equal to the sum of computation and communication costs for all tasks:

$$C_{final}(workflow)=\sum_{j=1}^{n} C(T_j, sched(T_j))\quad(5)$$

4. The Proposed Algorithm

At first, a brief look at basic concepts is presented. In the proposed scheduling algorithms, we have two notions of the start times of tasks, i.e. the earliest start time computed before scheduling the workflow, and the actual start time which is computed after the tasks are scheduled. The Earliest Start Time of each unscheduled taskT_i, EST(T_i), is defined as follows:

$EST(T_{entry}) = 0$
$EST(T_i)=Max_{T_p\in pred(T_i)}\{EST(T_p)+MET(T_p)+TT(T_p,T_i)\}$(6)

where the Minimum Execution Time of a task T_p, MET(T_p), is the execution time of task T_p on an instance $I_j\in I$ which has the minimum ET(T_p, I_j) between all available instances. Note that MET(T_{entry}) and MET(T_{exit}) equal zero.TT(T_p,T_i) is the data transfer time of the dependency $Data_{pi}$. Accordingly, the Earliest Finish Time of an unscheduled task T_i, EFT (T_i), can be defined as follows:

$EFT (T_i) = EST (T_i) + MET (T_i)$ (7)

The latest finish time for each unscheduled task is calculated as follows:

$LFT(T_{exit}) = D$
$LFT(T_i) =Min_{T_p\in child(T_i)}LFT(T_p)—MET(T_p) —TT(T_p,T_i)\}$
(8)

where LFT(T_i) is the latest time at which T_i can finish its computation such that the whole workflow can finish before the user defined deadline.

The general idea is that the proposed algorithm breaks workflow in such a way that tasks with the most dependency are scheduled to run on one cloud.

In the proposed algorithm, the critical paths are identified according to the new definition. EFT,EST, and LFT are computed for all nodes. Then we define the degree of dependence that is calculated for all nodes and finally for all paths. Nodes are ranked and scheduled so that the best possible cloud is assigned to each path. In the following, we explain the steps of the proposed algorithm.

4.1. Step One: Identify the Partial Paths with Minimal Communications

The Critical Parent of a node T_i is the unassigned parent of Ti that has the latest data arrival time at T_i. The partial critical path for each workflow graph is calculated as follows: we begin with T_{exit} and follow back the critical parents until we reach T_{entry}, and so we find the overall real critical path of the workflow graph .The proposed algorithm has changed this concept and the graph is broken into paths whose tasks together are the largest data exchange. One of the conditions of the generated paths is that all nodes have a maximum of one parent and one child in every path. Algorithm 1 shows how to break the workflow graph to paths with the mentioned conditions.

4.2. Step Two: Preprocessing

At this step, the Degree of Dependence (DOD) of each path to the other paths is calculated. Thus, because of the limitation of free capacity of every cloud, the algorithm specifies that the paths should be scheduled on one cloud. At first, DOD is calculated for tasks that their critical parents are located in another path according to Algorithm1. Thus, the DOD of T_1 to T_2 is equal to the start time of T_1 regarding the arrival time of data from T_2 in the other path, minus the start time of T_1 regardless of the arrival time of data from T_2 in the other path:

$$DOD(T_1,T_2) = (EST(T_2) + MET(T_2) + TT(T_1,T_2))-$$
$$(EST(T_3) + MET(T_3) + TT(T_1,T_3))\quad(9)$$

Algorithm 1: Finding Critical Paths

```
1. Input:$W=(T,E),T=\bigcup_{i=1}^{n} T_i,E=\{(T_i ,T_j, Data_{ij})|(T_i,T_j)\in T \times T\}$,
   $E_{ij}=(T_i,T_j,data_{ij})$;
2. Output:CriticalPaths=$\bigcup_{i=1}^{s} CP_i$, $CP_i$=(startNode$_i$, endNode$_i$,$E'_i$| $E'_i$C E);
3. coveredPaths← ∅, coveredNodes← ∅;   /*set them to empty*/
4. SE← Sort(E,Data);  /*Sort all E in descending  Data order */
5. forall ($E_{ij}\in$ SE) do
6. if($E_{ij}\notin$coveredPathsand$T_i\in$cøveredNodesand$T_j\in$coy̸eredNodes)then
7.         add ($T_i$,$T_j$,$E_{ij}$) to coveredPaths
8.         add $T_i$,$T_j$ to coveredNodes
9.    end if
10. if($E_{ij}\notin$coveredPathsand$T_i\in$cøveredNodesand$T_j\in$coveredNodes)then
11.           if( ∃$CP_e\in$coveredPaths | $CP_e=(T_j,T_{j'},E_{jj'})$) then
12.           $CP_e=(T_i,T_{j'},E_{ij})$
13.                add $T_j$ to coveredNodes
14.           end if
15.    end if
16. if($E_{ij}\notin$coveredPathsand$T_i\in$coveredNodesand$T_j\in$cov̸eredNodes)then
17.           if(∃$CP_e\in$coveredPaths |$CP_e=(T_{i'},T_i,E_{i'i})$) then
18.           $CP_e$= $(T_{i'},T_j,E_{ij})$
19.                add $T_i$ to coveredNodes
20.           end if
21.    end if
22. if($E_{ij}\notin$coveredPathsand $T_i\in$coveredNodesand$T_j\in$coveredNodes)then
23.           if((∃$CP_e\in$coveredPaths| $CP_e=(T_{i'},T_i,E_{i'i}$ ))and
              (∃$CP_{e'}\in$coveredPaths | $CP_{e'}=(T_j,T_{j'},E_{jj'})$)) then
24.           $CP_e$← $(T_{i'},T_{j'},E_{ij})$
25.                Remove $CP_{e'}$fromcoveredPaths
26.           end if
27.    end if
28. end for
29. Return coveredPaths
```

whereT_1 is the critical parent of T_2 that has been located in a different path with the path of T_1 according to Algorithm 1, and T3 is the parent of T_1 in the path produced by Algorithm 1.It should be noted that TT(T_1, T_2) is calculated by using the inter-cloud speed while $TT(T_1,T_3)$ is calculated by using the intra-cloud speed. In addition, DOD is calculated only for two nodes such that one is the critical parent of the other, but has been located in a different path from the path produced by Algorithm 1, and DOD for other pairs of nodes is considered to be zero. After calculating the DOD of all nodes, DOD of each path to the other path is calculated, i.e. that is equal to the sum of DOD of all nodes in the path to nodes in the other path and vice versa:

$$DOD(CP_1,CP_2)=\sum_{T_i\in T'_{CP_1}}\sum_{T_j\in T'_{CP_2}} DOD(T_i,T_j) +\sum_{T_j\in T'_{CP_2}}\sum_{T_i\in T'_{CP_1}} DOD(T_j,T_i) \quad (10)$$

where T'_{CP_1} and T'_{CP_2} respectively are the set of nodes in the path CP_1 and CP_2. Algorithm 2 shows how to calculate DOD of two paths.

4.3. Step three: Allocation of Resources to the Partial Paths

At this step, a degree is assigned to each node in the workflow. The degree is based on the estimated time required to execute tasks. In this case, we begin with the end of the workflow and calculate the degree of root nodes. The degree is equal to the execution time of these tasks in the fastest available instance. Then, we calculate the degrees of all parent nodes. The degree of each parent is equal to the sum of the maximum degree of the children and data transmission time from the parent to the child. Similarly, the degrees of all tasks in the workflow are calculated. These degrees are arranged in ascending order. Then tasks from the ordered list are traversed and scheduled onto the best cloud. Algorithm 3 shows the procedure. In this algorithm, we begin from the node with the highest degree and based on the assigned degree, the operation is repeated for each node. The node that is to be scheduled is now called the current node. Among all the paths identified by Algorithm 1, the path that includes the current node is called the current path. First, we examine if there is a node on the current path that is scheduled. If the response is positive, we try to find the best instance preferably in the cloud in which the parent of the current node is placed(First we will search for available instances at the desired cloud, and if a good instance could not be found we create a new instance in that cloud). If the response is negative, we consider if there is a scheduled node on the paths that depends on the path of the current node. If affirmative, we try to find the best instance in the cloud on which the path that depends on to the current node is scheduled. If there are several options, we select a cloud that has a greater volume of transactions (according to the calculated DOD by using Algorithm 2) in the current path. If the response is negative, we introduce an appropriate cloud based on our preliminary estimates and create the best instance on it. The preliminary estimates check that if all workflow nodes are separately scheduled on one cloud, which cloud would be the least expensive one (Any node of the workflow that cannot be executed even in the fastest instance of a cloud is not considered). So, we select a cloud for which the ratio of the cost to the number of instructions of executable tasks is the least. After scheduling this node, the node with the next degree is chosen.

Algorithm 2: Computing Degree Of Dependency of two paths(DOD)

```
1. Input:W=(T,E), $T=\bigcup_{i=1}^{n} T_i$ , $E=\{(T_i,T_j, Data_{ij})|(T_i,T_j)\in T\times T\}$,
        $E_{ij}=(T_i,T_j,data_{ij})$, CriticalPaths=$\bigcup_{i=1}^{s} CP_i$;
2. Output:$\bigcup_{c=1}^{s(s-1)}\{DOD(CP_i,CP_j)|CP_i,CP_j \in$ CriticalPaths$\}$;
3. for all ($T_i\in T$) do
4.    Compute EST ($T_i$), EFT ($T_i$) and LFT ($T_i$);
5. end for
6. for all ((Ti , Tj)$\in$ ($T\times T$)) do
7.  $CP_{T_i} \leftarrow$ FindCriticalPath($T_i$);    /*Path of $T_i$ form CriticalPaths*/
8. $CP_{T_j} \leftarrow$ FindCriticalPath($T_j$);    /*Path of $T_j$ form CriticalPaths*/
9. CiriticalParent$_{T_i} \leftarrow$ Max$_{T_p\in pred(T_i)}\{EFT(T_p)+TT(T_p,T_i)\}$;
          /*node that is normal critical parent of $T_i$ according [6]*/
10.    if($T_j$=CiriticalParent$_{T_i}$and $CP_{T_i} \neq CP_{T_j}$)
11. $DOD(T_i,T_j) = (EST(T_j) + MET(T_j) + TT(T_i,T_j)) – (EST(T_3) +$
              $MET(T_3) + TT(T_i,T_3))$;
12. else
13. $DOD(T_i,T_j)=0$;
14. end if
15. end for
16. for all ($CP_i, CP_j \in$ CriticalPaths)do
17. $DOD(CP_i,CP_j)=\sum_{T_m\in T'_{CP_i}}\sum_{T_n\in T'_{CP_j}} DOD(T_m,T_n)+$
              $\sum_{T_n\in T'_{CP_j}}\sum_{T_m\in T'_{CP_i}} DOD(T_n,T_m)$
18. end for
```

5. Performance Evaluation

In this section, experiments to verify the performance of the algorithm are proposed. The proposed algorithm is examined from two aspects:

a. Specifying the criteria for evaluating the quality of the proposed algorithm as compared with the other existing algorithms. For this purpose, two measures are used to compare quality: 1) Compare the shortest execution time of the algorithms, 2) Compare the financial cost of scheduling and the cheapest possible scheduling of each algorithm.

b. Evaluate the performance of the algorithm about workflows in comparison with the other existing algorithms. For this purpose, several workflows are studied.

5.1. Experimental Setup

For each experiment, we assume 10 clouds with each cloud having 10 separate services with different processor speeds and different prices. The processor speeds are selected randomly so that the fastest service is roughly five times faster than the slowest one, and accordingly, it is roughly five times more expensive. The average bandwidth between the computation services is set to 1 Gbps and the average bandwidth between clouds of 100 Mbps has been assumed. One-hour time slots are used. In the experiments, normal cost is calculated as follows:

$$NC=\frac{\text{total schedule cost}}{Cc} \quad (11)$$

where Cc is the cost of executing the same workflow with the cheapest strategy (scheduling of all nodes on a separate version of the cheapest available service) and normal makespan or execution time is calculated as follow:

$$NM =\frac{\text{schedule makespan}}{M_H} \quad (12)$$

whereM_H is the time of executing the same workflow with the fastest strategy (scheduling of all nodes on a separate version of the fastest available service). The final deadline is changed by a factor of α in the experiments. In this case, the deadline is calculated from the product of αat execution time of the fastest strategy.

5.2. Experimental Results

The proposed algorithm is tested for three common workflows presented in Table 2. We compared the proposed algorithms with IC-LOSS scheduling algorithm described in Section II because this algorithm has the same criteria as the proposed algorithm. For the Sipht, CyberShake, and Epigenomics workflows the proposed algorithm has a better performance, as shown in (Fig. 1, 2 and3). Experiments show that the proposed algorithm has a better performance than others when the workflow tasks have more data dependency and dispersal communication between them.

Table 2. Workflows Classes

Workflow	Small	Medium	Large
CyberShake	30-50	100	1000
Sipht	30-60	100	1000
Epigenimics	24-46	100	997

Algorithm 3: Extended PCP

```
19. Input:W=(T,E), T=$\cup_{i=1}^{n} T_i$ , E={$(T_i, T_j, Data_{ij})|(T_i,T_j)\in T \times T$},
        $E_{ij}$=(Ti, Tj, dataij), CriticalPathes=$\cup_{i=1}^{s} CP_i$;
20. Output:$\cup_{i=1}^{n}\{(T_i, sched(T_i)\}$,workflow schedule;
21. ranks=ComputeRank(T);/*the ranks are calculated in a bottom-up
        direction in workflow*/
22. ST ←Sort(T, ranks);                /*Sort all T in descending ranks
        order*/
23. for all (Ti∈ST) do
24.  bestCloud ← ∅,T′ ← ∅,T″ ← ∅;   /*set them to empty*/
25.    $CP_{T_i}$ ← FindCriticalPath($T_i$);   /*Path of $T_i$ form CriticalPaths*/
26. T'←FindScheduleNodes($CP_{T_i}$); /*Nodes of $CP_{T_i}$ that is scheduled*/
27.    for all(CP ⊂ CriticalPaths |DOD($CP_{T_i}$,CP)≠ 0) do /*T" is
        scheduelednodes in all depended on paths of $CP_{T_i}$*/
28. T"←T"∪FindScheduleNodes(CP);/*add Nodes of CP that is
        scheduelede to T"*/
29.   end for
30. if(T'≠ ∅) then
31. bestCloud←FindBestCloudInPath($T_i$); /*bestCloud for $T_i$ according
        $CP_{T_i}$*/
32. else  if(T"≠ ∅)then
33. bestCloud←FindBestCloudInDependedPath($T_i$); /*Best Cloud for $T_i$
        According DependedPaths*/
34. else
35. bestCloud←PreCloudAssign() ;/*Estimate bestcloud  */
36. end if
37.    end if
38.    if(bestCloud≠ ∅andbestcloud is not full)then/*if bestcloud can
        accept more requests according to its strategy*/
39. AssignBestInstance($T_i$ ,bestCloud);/*Assign $T_i$ to BestInstance in
        bestCloud*/
40. else
41. AssignBestInstanceInFree($T_i$);/*Assign $T_i$ to BestInstance that exists
        in all Clouds*/
42. end if
43.   Mark $T_i$ as scheduledNode;
44. end for
```

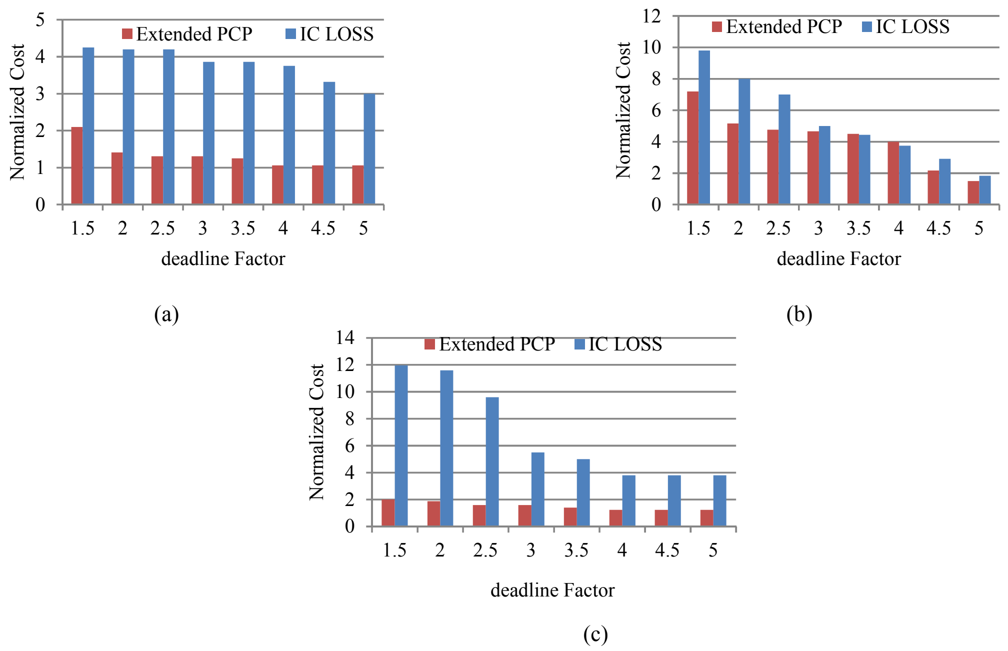
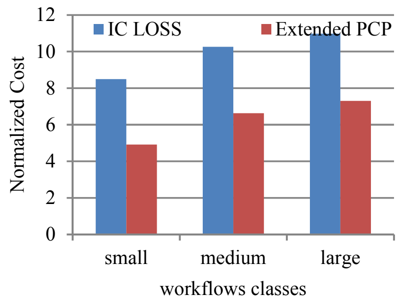
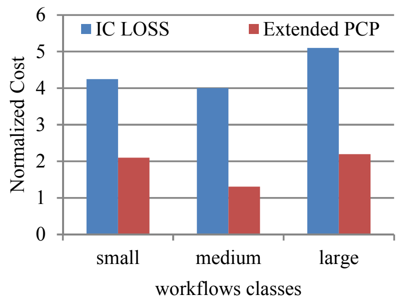


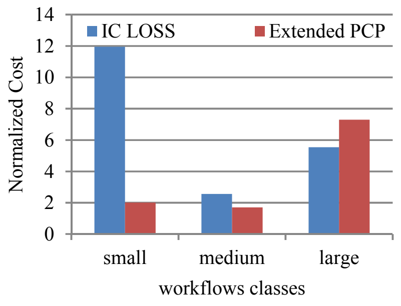
Fig. 1. The Normalized Cost of scheduling workflows with Extended PCP and IC-LOSS with the time interval equal to 1 h. (a). CyberShake. (b).Sipht. (c). Epigenomics.



(a)

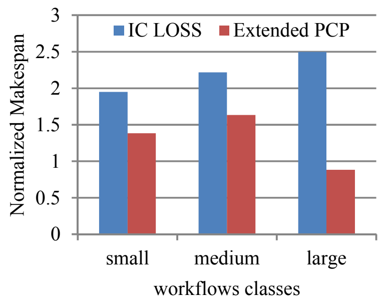


(b)

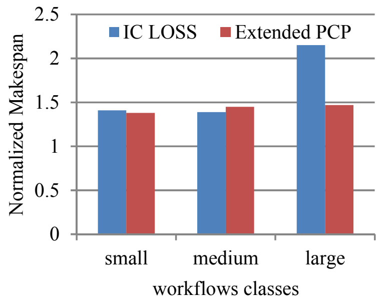


(c)

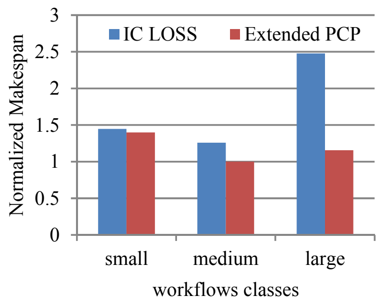
Fig. 2. The Normalized Cost of scheduling workflows with Extended PCP and IC-LOSS for different workflows classes. (a). CyberShake. (b). Sipht. (c). Epigenomics.



(a)



(b)



(c)

Fig. 3. The Normalized Makespan of scheduling workflows with Extended PCP and IC-LOSS for different workflows classes.(a). CyberShake. (b). Sipht. (c). Epigenomics.

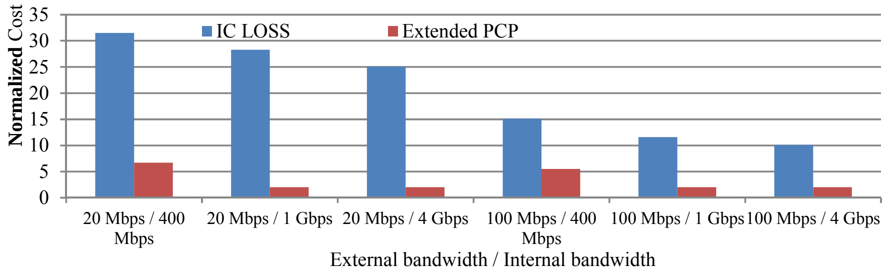
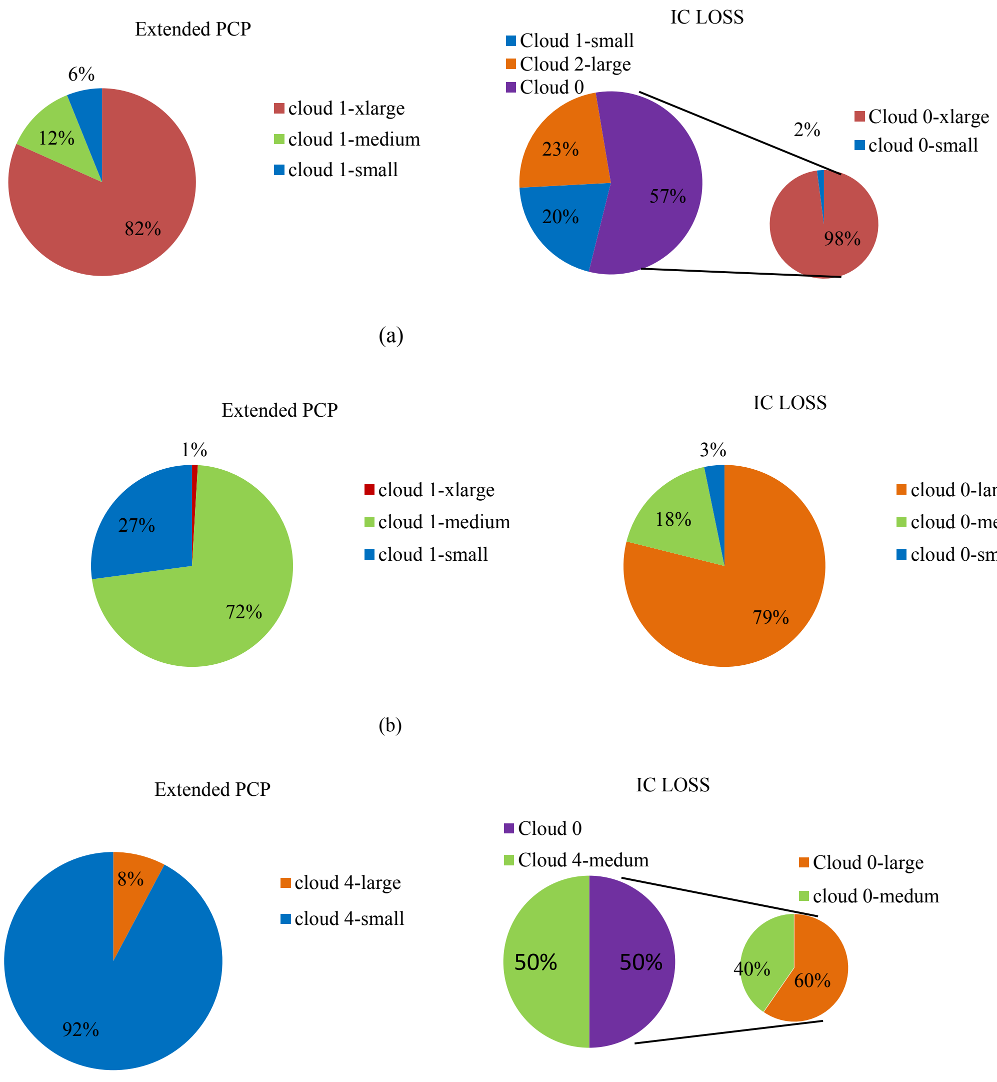


Fig. 4. The Normalized Cost of scheduling Epigenomics workflow for different external/internal bandwidth values



(c)

Fig. 5. Used instances (percentage of each instance type. a) CyberShake. b) Sipht. c) Epigenomics.

Fig 4 shows the Normalized Cost of scheduling Epigenomics workflow for different external/internal bandwidth values. As seen in Fig 4, the proposed algorithm shows more reaction to internal bandwidth and the cost is reduced by increasing it. Fig5. Shows the percentage of each instance type which is used by IC-Loss and Extended PCP algorithms. We can see from Fig.5 that the proposed algorithm tries to use the instances from one cloud, while IC-LOSS uses instances from different clouds. It leads to minimize the amount of communication between different clouds in our algorithm and so the cost and execution time are minimized.

6. Conclusions and Future Work

In this paper, a scheduling algorithm for data-driven workflows in a multi-cloud environment, called Extended PCP is proposed. In this proposed algorithm, we break a workflow into the paths whose tasks have the largest data exchange together and have the minimum data exchange between each other. A new criterion for controlling data flow, called Degree of Dependence or DOD, which is calculated for all nodes and paths is defined. This criterion will help us minimize the amount of communication between different paths by selecting the tasks that are more appropriate to be with each other. This algorithm considers communication as a very important factor that influences the whole scheduling. This makes the algorithm suitable for data-driven workflows in which a large amount of data is transferred between their tasks. The time complexity of the algorithms is $O(n^2)$, where n is the number of workflow tasks. The polynomial time complexity makes it a suitable option for the cases with large workflows. The proposed algorithm is evaluated by comparing its performance on scheduling three synthetic workflows that are based on real scientific workflows with different structures and different sizes, with IC-Loss [12]. The results show that the proposed algorithm has a better performance in workflows with a high amount of communication.

In the future, we intend to improve the proposed algorithm for multi-workflows scheduling in a multi-cloud environment and by taking into account other criteria such as fairness between providers. The next planned future work is extending the algorithm for dynamic environments where there is a possibility of changing the conditions of providers at any time in order to maximize the profits of users and providers at the same time.

References

[1] Petcu, D.: "Consuming resources and services from multiple clouds", *Journal of Grid Computing*, 12, (2), pp. 321-345*,* 2014.

[2] Yu, J., and Buyya, R.: "A taxonomy of scientific workflow systems for grid computing", *ACM Sigmod Record*, 34, (3), pp. 44-49*,*2005.

[3] Chen, R., Yang, M., Weng, X., Choi, B., He, B., and Li, X.: "Improving large graph processing on partitioned graphs in the cloud", in Editor (Ed.)^(Eds.): "Book Improving large graph processing on partitioned graphs in the cloud", (ACM, 2012, edn.), pp. 3.

[4] Li, A., Yang, X., Kandula, S., and Zhang, M.: "CloudCmp: comparing public cloud providers", in Editor (Ed.)^(Eds.): "Book CloudCmp: comparing public cloud providers" (ACM, edn.), pp. 1-14, 2010.

[5] Wang, G., and Ng, T.E.: "The impact of virtualization on network performance of amazon ec2 data center", in Editor (Ed.)^(Eds.): "Book The impact of virtualization on network performance of amazon ec2 data center" (IEEE, edn.), pp. 1-9, 2010.

[6] Abrishami, S., Naghibzadeh, M., and Epema, D.H.: "Deadline-constrained workflow scheduling algorithms for Infrastructure as a Service Clouds", Future Generation Computer Systems, 29, (1), pp. 158-169, 2013.

[7] Wu, Z., Ni, Z., Gu, L., and Liu, X.: "A revised discrete particle swarm optimization for cloud workflow scheduling", in Editor (Ed.)^(Eds.): "Book A revised discrete particle swarm optimization for cloud workflow scheduling" (IEEE, edn.), pp. 184-188, 2010.

[8] Pandey, S., Wu, L., Guru, S.M., and Buyya, R.: "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments", in Editor (Ed.)^(Eds.): "Book A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments" (IEEE, edn.), pp. 400-407, 2010.

[9] Yu, J., and Buyya, R.: "Scheduling scientific workflow applications with deadline and budget constraints using genetic algorithms", *Scientific Programming*, 14, (3-4), pp. 217-230, 2006.

[10] Genez, T.A., Bittencourt, L.F., and Madeira, E.R.: "Workflow scheduling for SaaS/PaaS cloud providers considering two SLA levels", in Editor (Ed.)^(Eds.): "Book Workflow scheduling for SaaS/PaaS cloud providers considering two SLA levels" (IEEE, edn.), pp. 906-912, 2012.

[11] Bittencourt, L.F., and Madeira, E.R.M.: "HCOC: a cost optimization algorithm for workflow scheduling in hybrid clouds", *Journal of Internet Services and Applications*, 2, (3), pp. 207-227, 2011.

[12] Sakellariou, R., Zhao, H., Tsiakkouri, E., and Dikaiakos, M.D.: "Scheduling workflows with budget constraints","Integrated research in GRID computing", pp. 189-202, Springer, 2007.

[13] Topcuoglu, H., Hariri, S., and Wu, M.-y.,"Performance-effective and low-complexity task scheduling for heterogeneous computing", *Parallel and Distributed Systems,* IEEE Transactions on, 13, (3), pp. 260-274, 2002.

[14] Van den Bossche, R., Vanmechelen, K., and Broeckhove, J.,"Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads", in Editor (Ed.)^(Eds.): "Book Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads" (IEEE, edn.), pp. 228-235, 2010.

[15] Houidi, I., Mechtri, M., Louati, W., and Zeghlache, D.,"Cloud service delivery across multiple cloud platforms", in Editor (Ed.)^(Eds.),"Book Cloud service delivery across multiple cloud platforms" (IEEE, edn.), pp. 741-742, 2011.

[16] Li, W., Tordsson, J., and Elmroth, E.,"Modeling for dynamic cloud scheduling via migration of virtual machines", in Editor (Ed.)^(Eds.),"Book Modeling for dynamic cloud scheduling via migration of virtual machines" (IEEE, edn.), pp. 163-171, 2011.

[17] Fard, H.M., Prodan, R., and Fahringer, T.,"A truthful dynamic workflow scheduling mechanism for commercial multicloud environments", *Parallel and Distributed Systems,* IEEE Transactions on, 24, (6), pp. 1203-1212, 2013.

[18] Fard, H.M., Prodan, R., Barrionuevo, J.J.D., and Fahringer, T.,"A multi-objective approach for workflow scheduling in heterogeneous environments", in Editor (Ed.)^(Eds.),"Book A multi-objective approach for workflow scheduling in heterogeneous environments" (IEEE Computer Society, edn.), pp. 300-309, 2012.

[19] Duan, R., Prodan, R., and Li, X.,"Multi-objective game theoretic schedulingof bag-of-tasks workflows on hybrid clouds", *Cloud Computing,* IEEE Transactions on, 2, (1), pp. 29-42, 2014.

[20] Montes, J.D., Zou, M., Singh, R., Tao, S., and Parashar, M.,"Data-driven workflows in multi-cloud marketplaces", in Editor (Ed.)^(Eds.),"Book Data-driven workflows in multi-cloud marketplaces" (IEEE, edn.), pp. 168-175, 2014.

Journal of Computer and Knowledge Engineering, Vol. 1, No. 1
DOI: 10.22067/cke.v1i2.58792

Safety Verification of Rate-Monotonic Least-Splitting Real-Time Scheduler on Multiprocessor System

Amin Rezaeian, Abolfazl Ghavidel, Yasser Sedaghat*

Abstract. In real-time task scheduling on multiprocessor systems, partitioning approach has received the attention of many researchers because of its higher least upper bound utilization of safe systems. Semi-partitioning allows some tasks to be split into subtasks and each subtask to be assigned to a different processor. Though task splitting improves the performance of systems, by counting each subtask as a separate task, it increases the effective number of tasks to be scheduled, which in turn, raises the execution overhead. This research is on semi-partitioning of tasks and assigning each partition to a separate processor to be scheduled by the well-known scheduler Rate-Monotonic (RM). Using our algorithm, we do not need to define release time for subtasks of a task to assure their non-concurrent execution and the number of effective tasks, in turn, is reduced. It is theoretically proven that with the proposed semi-partitioning and RM scheduling algorithm, all processors may safely run their tasks according to their deadlines. Further, experimental results on 3000 randomly generated task-sets indicates that not only is utilization factor boosted, but the number of broken tasks also is decreased.

keywords: Rate-Monotonic Least Splitting, Semi-Partitioning, Hard real-time, Multiprocessor Scheduling.

1. Introduction

Recently, the importance of utilizing embedded multicore and multiprocessor system-on-a-chip (MPSoC) has begun to appear as a sensible solution for both power efficiency and high-performance computing in diverse application areas including telecommunications, multimedia systems, computer games, space systems, and process control to name but a few. The importance of the issue becomes more noteworthy when we know that traditional computer-based control systems as well as high technology systems, require high-performance computers for their computations which are usually possible with multiprocessor or multicore systems. A multiprocessor system is composed of several processing elements, called processors, in which all processors can do their processing in parallel. If all processors have the same architecture with one processor repeated design in the processor, the architecture is called homogenous. They all share the same main memory, but each can have their own private cache memory. With this structure, a sequential computation can be shared among many processors if not more than one processor is executing the computation, simultaneously [1]. While manufacturers tend to use multiprocessors in new devices, the development of software facilities which use all available power of multiprocessors [2] is required. In this context, scheduling algorithms play a prominent role in safeness verification of hard real-time systems, i.e., making sure that every request is executed before its deadline. The problem becomes more challenging when there is more than one processor involved so that multiprocessor/multicore systems adds a new dimension to the analysis: how to assign tasks or their requests for different processors. Therefore, there are, overall, two key issues which are still open in multiprocessor scheduling as follows:

1. Task assignment to processors: Finding a target processor to run every selected task;
2. Identifying tasks priority: Making an appropriate order of priority to run tasks.

In this article, the problem of scheduling periodic hard real-time task sets with implicit deadlines on multiprocessors is investigated. What we mean by the implicit deadline is that the deadline of a request is the exact time when the next request arrives from the same task. Many other researchers have studied the same problem but new ideas have kept this practical area dynamic, attractive, and improving.

From the perspective of task migration, task assignment issue on a multiprocessor, generally, fall into two main categories:

1. Global: These algorithms are employed where tasks are allowed to migrate from one processor/core to another.
2. Partitioned: These algorithms are employed where tasks are not allowed to migrate from one processor/core to another.

Moreover, there is another important category which is the hybrid of the above methods called semi-partitioned.

In global scheduling, there is only one queue (or pool) of real-time requests and each processor takes its next execution request from this queue. Regardless of its advantages, in complex and large systems, the overheads of employing a single global queue would become too excessive. In partitioned approach, on the other hand, the set of tasks are divided and each partition is assigned to a separate processor. Finally, in semi-partitioned, some tasks are solely assigned to one processor and some tasks are shared among two or more processors, with the restriction that not more than one processor can work on a request for the shared task, simultaneously. Although the scheduler may be different for each of the three categories of, in almost all cases, the scheduler of all processors is considered to be the same.

Manuscript received September 11, 2016; revised October 17, 2016; accepted November 29, 2016.
Amin Rezaeian, Abolfazl Ghavidel and Yasser Sedaghat,
Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran.
The corresponding author's e-mail is: y_sedaghat@um.ac.ir

While global scheduling techniques, depending on the hardware architecture, potentially could have very high overheads owing to the fact that the task migration from one processor to another usually yields communication and cache missing cost; however, in fully partitioned techniques this does not occur. Nonetheless, non-migration techniques often do not use all available processor capacity. Moreover, it is a widely occurring situation where the total unused capacity may be more than task utilization but no single processor has enough capacity to schedule the task. To solve those problems, a hybrid technique (called semi-partitioned) is used which combines elements of global and partitioned techniques.

It is usually the case that semi-partitioned scheduling leads to a higher overall utilization of the whole system as compared to either of partitioned and global scheduling, for both fixed-priority and dynamic priority. Further, partitioning is a time-consuming task which is computationally equivalent to bin-packing problem that is known to be an NP-hard problem [3]. On the positive side, partitioning is done off-line. Therefore, for a small number of tasks, the time utilized for partitioning is tolerable but, for a large number of tasks efficient heuristics are used. Although an optimal semi-partitioned method has not yet been developed, many heuristic algorithms are presented by researchers. A semi-partitioned approach binds a disjoint set of whole tasks to each processor and allows the remaining tasks to be executed on multiple processors while all shared qualities are defined. In one of the researches on semi-partitioned methods in which Rate-Monotonic (RM) scheduler is used in each processor, worst-case utilization is reported to be $\ln(2) \approx 0.693$ [4].

In this article, a novel semi-partitioned scheduling algorithm called Rate-Monotonic Least Splitting (RMLS) is proposed for multiprocessors. The scheduler of each processor is RM with provisions to avoid simultaneous execution of a shared task by more than one processor. Using this algorithm, we see that the number of split tasks at most is equal to the number of used processors minus one. However, the actual number of split tasks might even be lower. Besides, no task is split into more than two subtasks. Splitting into fewer numbers of tasks has two benefits:

1. Effective number of tasks in the Liu and Leyland's bound is reduced: $\Theta(n) = n(2^{\frac{1}{n}} - 1)$
2. It increases overall system utilization.

The remainder of this article is divided into seven sections: We firstly present our system model and notations in Section 2. In Section 3, related works are briefly reviewed. Section 4, describes the proposed RMLS semi-partitioned scheduling. Section 5, is the theoretical foundations and safeness proof of the algorithm; in Section 6, the algorithm is simulated and results are documented, and finally a summary and future work are presented in Section 7.

2. System Model and Notations

We consider a system with *m* symmetric processors and the main aim of this article is to schedule a bag-of-tasks containing periodic hard real-time tasks with an implicit deadline in such a system. A very important assumption is that all tasks are synchronous. That is, the deadline of any arbitrary task request is the exact time when the next request of that task arrives. We have also further assumed that all tasks are preemptive. From now on, in order to avoid potential difficulty in naming, we simplify call it *task* in this article.

The following notations are used throughout the article:

1. n: total number of tasks
2. n_1: total number of tasks and subtasks
3. m: total number of available or processors
4. m_1: total number of used processors
5. τ_i: i^{th} task
6. T_i: minimum time between any two consecutive requests of task τ_i, i.e. minimum request interval of τ_i
7. C_i: worst case computation time needed by every request of task τ_i. It is clear that $C_i \le T_i$
8. $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$: set of all tasks (bag-of-task)
9. $\rho = \{P_1, P_2, \ldots, P_m\}$: set of all processors
10. u_i: the utilization of task τ_i which is equal to $u_i = \frac{C_i}{T_i}$
11. $U(\Gamma)$: total utilization of task-set Γ and subtasks

To evaluate the performance of scheduling heuristics, there are many theoretical factors such as utilization bound, speedup factor and many others. In addition to those factors, there are many articles which have used empirical methods to show and compare the relative performance of different algorithms. The most widely accepted are the number of randomly generated task-sets. Many different algorithms are now available to generate random task-sets and evaluate performance, but they are almost similar to one other in using parameters such as the number of total processors, task-set utilization, the number of all tasks in the task set, the distribution of task deadlines, the range of task periods, and the distribution of every task utilization. More details about the used performance evaluation algorithm in the present article and the number of randomly generated task-sets are provided in Section 6.

3. Related Works

Many studies have been conducted in the domain of homogeneous multiprocessor and multicore systems since the 1960s. Among those, semi-partitioned approaches, which try to merge and employ the best attributes of global and fully partitioned approaches, have a prominent role because of solving the task allocation problem which is analogous to the bin-packing problem. Anderson and Tovar [5], in one of the first studies in this respect, proposed an approach for scheduling hard real-time periodic tasks with implicit deadlines called EKG. In this approach, the parameter *k* has been used to control the splitting of tasks into the light and heavy sets. This suggestion is to set k=2 which yields the utilization bound of 66% and, on average, at most four preemptions per every task over the hyper period; yet, k=m gives the utilization bound of 100% and 2k preemption for each task. In this context, many researchers have proposed the semi-partitioned problem with Earliest Deadline First (EDF) scheduling [6], [7]. The best known worst-case utilization bound known using semi-partitioned EDF scheduling on multicores is 65% for

Earliest Deadline Deferrable Portion (EDDP) algorithm [8]. EDDP distinguishes between heavy tasks (those whose utilization are greater than 65%) and light tasks (other tasks with utilization < 65%) in such a way that the algorithm firstly assigns every heavy task to its own processor and then light tasks are placed on the remaining processors. They showed that 65% is a safe utilization bound for EDDP provided tasks which are periodic with implicit deadlines. Later, in 2009, they proposed EDF with Window-constraint Migration (EDF-WM) which has less context switching overhead [9].

On the other hand, relatively fewer algorithms are proposed for fixed-priority algorithms [10]. Rate Monotonic Deferrable Portion (RMDP) and Deadline Monotonic with Priority Migration (DM-PM) fixed-priority algorithms are proposed by Kato et al. [11, 12]. The worst-case utilization bound of those algorithms is shown to be 50%. RMDP consists of two phases: task assigning and task scheduling. The first phase is very simple: Sort all tasks in ascending order of T_i. Then, assign tasks to processors and if assigning a task causes the processor utilization bound to exceed, split the task into two subtasks. Sub-task 1 is placed on the current processor and sub-task 2 is assigned to the next processor. Task scheduling phase utilizes RM scheduler algorithm in every processor considering the fact that the second portion of a ready task in a processor cannot start to run until its first portion finishes execution.

PDMS_HPTS_DS is proposed by Lakshmanan et al. [2] which achieves the utilization bound of at least 60% providing tasks have an implicit deadline. It can, nevertheless, increase up to 65% if tasks are assigned to processors in order of decreasing utilization. Moreover, this bound can be extended to 69.3% for light tasks, i.e., tasks with utilizations less than 0.41. Guan et al. have proposed two algorithms which they called SPA1 and SPA2 [4], [10]. SPA2 has a pre-assignment phase in which special heavy tasks are first assigned to a separate processor. The advantages of this method are that the number of split tasks is m-1 and SPA2 reaches the worst-case utilization bound of 0.693. This is equal to Liu and Leyland's bound [13] for single processor systems The disadvantage includes, the worst-case bound in SPA2 is calculated using n which is the cardinality of the whole task-set, and every processor's utilization must be less than or equal to that. For example, although Liu and Leyland's least upper bound utilization for a two-task processor is approximately 0.83, but with SPA2 its utilization should not exceed 0.693. With this explanation, the claim that SPA2 has reached Liu and Leyland's utilization bound, does not seem to be entirely correct.

For supplementary information on hard real-time task scheduling algorithms and related issues, the reader is invited to refer to [14].

4. Rate Monotonic Least Splitting

The basic idea of the semi-partitioned method, which is being presented here, has been published in an in-progress research workshop [15]. In that paper, the fundamental theorem which shows the safeness of system was not proven. In addition, none of the other theoretical results provided by this article have appeared in that paper. Now, a brief introduction of the method is given here and new findings and performance evaluations follow. The method is called Rate-Monotonic Least Splitting (RMLS) because it is a semi-partitioned method in which at most m_1-*1* tasks are split and at the same time, to the best of our knowledge, the method of partitioning and task splitting presented here is very simple, comprehensible and it is shown to be efficient.

Our experiments show that the achieved processor utilization is approximately 9.6% higher than the best-known results for general real-time systems, i.e., no restrictions on utilization of individual tasks, up to now.

Having taken all the above explanations into account, in the following sub-section, we first propose our novel idea to demonstrate how tasks are placed on processors. Then, in the next sub-sections, we will have a fairly good discussion on the computational complexity of the RMLS algorithm.

4.1. Task Assignment

The proposed assignment algorithm is precisely outlined in Fig. 1 and we elaborate on it in the two super steps as follows:

Step 1: Selection of single tasks and pairs of tasks to assign each one to a separate processor (lines 4 to 25).

Step 2: Assignment of remaining tasks to remaining processors (lines 28 to 47).

In the first step, tasks are sorted in descending order of their utilizations and the result is saved as a sorted task-set. A greedy approach is followed to find single tasks or pairs of tasks which can be assigned to separate processors to which other tasks will not be assigned to. To such processors, no subtasks will be assigned. In this phase, two pointers, i and j, are set to the beginning and the end of the set, respectively. If θ (3) $\leq u_i + u_j \leq 1$ then these two tasks make a pair (lines 7 to 11) which is assigned to a separate processor; otherwise, one of these tasks is removed from the set (lines 18 to 22) and the process continues until the two pointers pass each other. The removed task will be scheduled in the second step. Step 1, continues by recognizing heavy tasks, i.e., a task τ_l with $u_l \geq \theta$ (2), and such task is assigned to a separate processor (lines 13 to 16.) No subtasks will be assigned to these processors.

To get the highest possible utilization, the scheduler of each processor with two tasks is a modified version of RM named Delayed Rate Monotonic (DRM) [16]. Suppose the two tasks $\tau_i = (C_i, T_i)$ and $\tau_j = (C_j, T_j)$, are solely assigned to processor p_k which has no other tasks. The delay time of each request of task τ_i, whose priority is higher than τ_j with respect to RM, is taken to be equal to T_i-C_i while there is no delay for requests of task τ_j. Here, you can consider that the delay is similar to (but not exactly the same as) ready time.

If the ready time of a request by τ_i is T_i-C_i and the request arrives at time t, then it would not be possible to start executing this request until time t+T_i-C_i. On the other hand, the delay of task τ_i is terminated at any time there is no pending request from task τ_j, and it will not re-enter delay state even if a new request arrives from τ_j. In addition, the delay of any request from this task can also end when T_i-C_i

time units have elapsed from the time that request is generated.

This modification will guarantee that if the total utilization of the two tasks τ_i and τ_j, assigned to a separate processor is less than or equal to one, then the processor will always run safely [17].

```
     Data: Task-set Γ //Includes Ti , Ci for each task i
     Result: Packing ρ
1    ρ←{}
2    k←0
3    Γ′←Γ
4    Sort Γ′ in descending order of tasks' utilization
5    i←1, j←length(Γ)
6    while i<j do
7      if $\Theta(3) \le u_{\tau'_i} + u_{\tau'_j} \le 1$ then
8        k←k+1
9        Add Pk to ρ
10       Move $\tau'_i$ and $\tau'_j$ from Γ to Pk
11       i←i+1 ; j←j-1
12     else
13       if $\Theta(2) \le u_{\tau'_i}$ then
14         k←k+1
15         Add Pk to ρ
16         Move $\tau'_i$ from Γ to Pk
17       else
18         if $\Theta(3) \le u_{\tau'_i} + u_{\tau'_j}$ then
19           i←i+1
20         else
21           j←j-1
22         end
23       end
24     end
25   end
26   k←k+1
27   Add Pk to ρ
28   while Γ≠∅ do
29     $\tau_i$←the task with the highest priority in Γ
30     if $U(p_k) = \Theta(|p_k|)$ then
31       k←k+1
32       Add Pk to ρ
33     end
34     if $U(p_k) + u_i \le \Theta(|p_k| + 1)$ then
35       Move $\tau_i$ from Γ to Pk
36     else
37       if $U(p_k) < \Theta(|p_k| + 1)$ then
38         Select $\tau_j$ from Γ where
             $u_{\tau_j} + U(p_k) < \Theta(|p_k| + 2)$
39         Move $\tau_j$ from Γ to Pk
40         Split $\tau_i$ into $\tau_{i1}$ and $\tau_{i2}$ such that
41           $u_{\tau_{i1}} = \Theta(|p_k| + 1) - U(p_k)$
42         Replace $\tau_i$ in Γ by $\tau_{i2}$ with $u_{\tau_{i2}} = \frac{C_{j2}}{T_j - C_{j1}}$
43         Move $\tau_{i1}$ to Pk
44         k←k+1
45         Add Pk to ρ
46       end
47     end
48   end
49   m1←k        //number of used processors
```

Fig. 1. The packing algorithm

The scheduler of all processors, except those which has two tasks, is the traditional RM without any delay or ready time for requests.

Step 1, serves two sole purposes: (1) It increases the number of processors with higher utilization than those processors which are assigned tasks in Step 2, and (2) It increases the number of processors with no split task and hence, it decreases the total number of tasks which will be split in Step 2. Thus, by reducing the effective number of tasks (the total number of tasks and subtasks) the intuition is that there would be less number of tasks preemptions during run time.

In Step 2, all unassigned tasks will be sorted in decreasing order of RM priorities, i.e., the non-descending order of their request interval lengths. An empty processor is selected and then an unassigned task is selected from the sorted list to assign to the selected empty processor. This scenario will continue to repeat itself until the current task, say task τ_i, will overload the processor (lines 29 to 35). Then a search amongst the remaining unassigned tasks must be done to find a task with maximum utilization which can be assigned to this processor without overloading it. If one is found, it will be assigned to the processor. If this processor is not filled, task τ_i is then split into two subtasks so that the first subtask is assigned to the current processor and makes it full with respect to Liu and Layland's bound for the respective number of tasks and subtasks in this processor (lines 37 to 44).

To clarify, suppose that the current processor is p_k and task τ_i is the task which is split into two subtasks τ_{i1} and τ_{i2} with execution times C_{i1} and C_{i2}, respectively. The utilization of τ_{i1} is $u_{i1} = \frac{C_{i1}}{T_i}$ for processor p_k. A new processor, p_{k+1}, is picked up and the second part of task τ_i which was split, i.e., τ_{i2}, is assigned to this processor. Although the actual utilization of this subtask is $\frac{C_{i2}}{T_i}$, its effective utilization on processor p_{k+1} is taken to be:

$$u_{i2} = \frac{C_{i2}}{T_i - C_{i1}} \tag{1}$$

The effective utilization of this subtask is greater than its actual utilization, i.e. $\frac{C_{i2}}{T_i - C_{i1}} > \frac{C_{i2}}{T_i}$. Therefore, the difference of these two values is what we have to sacrifice because there may be some situations in which both processors that share task τ_i want to execute this task but the only processor that can run it at this time, is the processor whose index is the lower. In Lemma 3, we will prove that, in the worst case, the second part of a request from task τ_i will have a time length of T_i-C_{i1}, not T_i, to be executed. As mentioned earlier, this is due to the interference between the two processors that share task τ_i.

For example, suppose tasks τ_1= (1.1, 4), τ_2= (3, 17), and τ_3= (3.2, 18) are completely assigned to processor p_1 and task τ_4 = (6.55, 20) is broken into two subtasks $\tau_{4\text{-}1}$ = (2.55, 20) and τ_{4_2} = (4, 20) which are assigned to processors p_1 and p_2, respectively. Besides, tasks τ_5 = (5, 25) and τ_6 = (6, 30) are completely assigned to processor p_2 and from task τ_7 = (7, 42) the subtask τ_{7_1} = (5.65, 42) is assigned to processor p_2. The rest of task τ_7, i.e., τ_{7_2} = (1.35, 42) and task τ_8 = (47.4, 60) are assigned to processor p_3. The total utilization of these three processors are computed as follow:

$$U_1 = \frac{1.1}{4} + \frac{3}{17} + \frac{3.2}{18} + \frac{2.55}{20} = 0.7567.$$

$$U_2 = \frac{4}{20 - 2.55} + \frac{5}{25} + \frac{6}{30} + \frac{5.65}{42} = 0.7637.$$

$$U_3 = \frac{1.35}{42 - 5.65} + \frac{47.4}{60} = 0.8271$$

In RMLS algorithm outlined in Fig. 1, the process of assigning tasks to the processors continues until all tasks are assigned. If processors are exhausted but some unassigned tasks still remained, the assignment is unsuccessful; otherwise, it is successful.

Suppose the assignment is successful, RMLS splits at the most m_l-1 tasks, where m_l is the actual number of used processors. There is no release time or delay time for the tasks that are assigned in Step 2 and thus, the scheduler is the traditional RM with a minor amendment. Obviously, it is clear that only one processor can execute a sequential task at any given time. In order to make sure this vital condition is observed, whenever there is a conflict, the processor with the lower index must have the precedence in executing the shared request. That is, under RMLS, the execution of that portion of the split task which is assigned to the lower indexed processor is not affected by the execution of that portion of the split task which is assigned to the higher index processor. However, the execution of that portion of the split task which is assigned to the higher indexed processor may be delayed because the lower indexed processor is running the split task. In other words, the lower indexed processor can run its own portion of the split task whenever it desires to, but the higher indexed processor can only run its own portion if the lowered indexed processor is not running its portion of the split task.

Using Equation (1) in computing the total utilization of processor p_{k+1} will reduce the actual sum of the task utilization on processor p_{k+1} to less than Liu & Layland's bound. However, this is an unavoidable cost that we have to pay for all processors to run safely.

On the positive side, by using RMLS, there is no need to restrict the sum of utilizations of all processors to be less than or equal to $\theta(n_l)$ (where n_l is the total number of tasks and subtasks of the whole system after partitioning and complete assignment). That is, the total utilization could be more than 69.3% and the system is still in a safe state for any arbitrary number of processors (e.g. for a large number of processors).

4.2. Computational Complexity

To calculate computational complexity, we divide the scheduler into three steps. The task-set is sorted in the first step; thus, for a task-set of size n, the computational complexity of the first step would be O(nlog n). In the second step, a search for large tasks and pairs of tasks is done. This step is preceded by a loop. This loop continues until i and j variables, which respectively started from the beginning and the end of the tasks list, become equal. Since at least one of those variables is changed during each iteration, this loop iterates at most n times. Thus time complexity of the second step is to O(n). The third step assigns the remaining tasks (at most n tasks) to processors. This step contains a loop, which iterates one time for each task (either it is split or not). However, before every task splitting, a search must be conducted among unscheduled tasks. This search might find a small task to put in the current processor. As this procedure is done for every processor, the total time complexity of the third step would be O(mn), in which m is the number of processors used.

The total time complexity of the scheduling algorithm can be calculated by sum of complexities of steps 1 to 3, which is
O(max(n log n, mn)).

5. Safeness Verification of RMLS

The great advantage of RMLS is that Liu & Layland's bound is only computed based on the total number of tasks and subtasks assigned to every processor separately and that is why well-known similar previous researches could not reach such a high degree of freedom. As a fine example, Guan et al. [4] proposed a semi-partitioned algorithm with this additional constraint that the total utilization of all tasks coupled with subtasks must not exceed Liu & Layland's bound, whereas we successfully removed this restriction in our algorithm RMLS. We first present three lemmas and then prove the claimed statement.

In the rest of this article, it is assumed that two processors p_k and p_{k+1} share a task $\tau_i = (T_i, C_i)$ and for each request of the common task C_{i1} is executed by p_k and C_{i2} is executed by p_{k+1} so that $C_i=C_{i1}+C_{i2}$. In addition, effective utilization of the second part of a shared task is used as its utilization in the corresponding processor.

Lemma 1.

If Liu & Layland's bound, is satisfied by all processors, the second part of a request from a shared task, τ_i, between two processors, p_k and p_{k+1}, never overruns.

Proof.

The preference of executing a request from the shared task τ_i between processors p_k and p_{k+1} is given to p_k. Furthermore, the second part of a request from task τ_i has the highest priority within all tasks in p_{k+1}. Therefore, as soon as a request from task τ_i is generated, its execution starts by either p_k or p_{k+1} and continues executing (migrating between the processors, if necessary) until the second part of the task is completed. Therefore, in the worst case scenario, the execution of the second part of the task will be completed after a time length of C_i is passed from its request ($C_i \leq T_i$).

Definition 1.

A conflict-idle period is a time interval in which both processors, p_k and p_{k+1}, that share the shared task τ_i, want to run a request from the task, but because p_k is given a higher precedence, it will proceed with its execution; and at the same time, there is no other pending requests for processor p_{k+1} within this period and it will be idle. Note that, not all conflict periods of processors p_k and p_{k+1} are necessarily conflict-idle because if there are other requests for p_{k+1} then it will proceed with their execution and hence, it will therefore not be idle.

Consider a situation in which the task τ_i is split into subtasks τ_{i1} and τ_{i2}, and they are assigned to processors p_k and p_{k+1}, respectively. Subtask τ_{i1} is the task with the lowest priority (or in some cases the second lowest priority) within processor p_k while task τ_{i2} is always the task with the highest priority among all tasks and subtasks assigned to processor p_{k+1}. This decreases the chance of encountering a situation in which both processors that want to simultaneously run the task τ_i; however, it is not zero. Therefore, conflict periods are very rare and as a result, seldom will conflict-idle periods to take place.

Lemma 2.

Suppose two processors p_k and p_{k+1} share a task τ_i and run n_k and n_{k+1} tasks while their total utilization is not greater than $\Theta(n_k)$ and $\Theta(n_{k+1})$, respectively. If there will not be any conflict-idle period with respect to τ_i, then both processors will always run overrun-free.

Proof.

Since processor p_k has a higher precedence to run τ_i than p_{k+1}, this processor will always run overrun-free. On the other hand, the only effect that p_k can have on the execution of tasks of processor p_{k+1} is that it may postpone the execution of the second part of a request from the shared task. This may harm the overrun-freeness of the shared task in p_{k+1} but it can be beneficial to the other tasks of this processor. However, in Lemma 1, it was proven that the second part of a request from a shared task never overruns. Therefore, this processor runs overrun-free as well.

Lemmas 1 and 2, will hold even if actual utilization of subtask τ_{i2}, i.e., $\frac{C_{i2}}{r_i}$, is used for computing the utilization of p_{k+1}. It is for compensation of possible conflict-idle periods that, in general, effective utilization of the shared task on processor p_{k+1} is computed as $\frac{C_{i2}}{T_i - C_{i1}}$.

Definition 2.

The remaining utilization of a request (not a task or subtask) at a given time is defined to be its remaining execution time divided by its remaining time to reach the deadline. At the exact time when a request is generated its remaining utilization is equal to its actual utilization. However, as time passes, its remaining utilization may fluctuate depending on how much time has been passed from its request time and how much it has been executed until the time that the remaining utilization is computed.

For example, suppose task $\tau = (10, 4)$ has generated a request at time 20 and the current time is 26 and up to now, this request has received 1.5 unit of CPU time. The remaining utilization of the request at time 26 is $(4 - 1.5)/(30 - 26) = 0.625$.

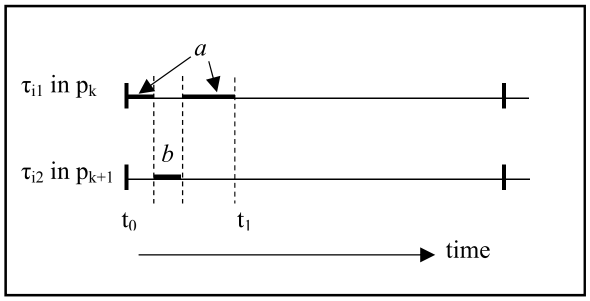


Fig. 2. A Sample execution of parts of a split task

Lemma 3.

Suppose two processors p_k and p_{k+1} share a task τ_i. The remaining utilization of a request from τ_i for processor p_{k+1} is maximal at the exact time when the execution of processor p_k's share of this request is completed when p_k starts this request immediately after it is generated and continues running it until its share is completed.

Proof.

Suppose as soon as a request from τ_i is generated at a time t_0, processor p_k starts to execute it until its share is finished at time t_0+C_{i1}. At this time, effective utilization of the subtask τ_{i2} on p_{k+1} is equal to $\frac{C_{i2}}{T_i - C_{i1}}$. We show that this is, in fact, the maximal effective utilization of τ_{i2}, which means subtask τ_{i2}'s effective utilization never becomes greater than this value. It is worth mentioning to recall that requests of task τ_i have the highest priority in processor p_{k+1}. This implies that any request for this task will be immediately picked up for execution by p_{k+1} if p_k is not executing it.

On the other hand, if the execution of the second part of a request of task τ_i is completed by processor p_{k+1} then its remaining utilization becomes zero and remains zero until a new request is generated from the same task.

With these points in mind, consider a situation where at any time t_1 ($t_0 \le t_1 \le t_0+C_i$), processor p_k has executed this request for the duration of length an ($a \le C_{i1}$), and processor p_{k+1} has executed the same request for duration b ($b < C_{i2}$ and $a+b = t_1-t_0$). This example is illustrated graphically in Fig. 2.

At time t_1 effective utilization of τ_{i2} is:

$$\frac{C_{i2} - b}{T_i - (a + b)}$$

Since $a \le C_{i1}$,

$$\frac{C_{i2}-b}{T_i-(a+b)} \le \frac{C_{i2}-b}{T_i-(C_{i1}+b)} = \frac{C_{i2}-b}{T_i-C_{i1}-b}$$

To show that the maximal effective utilization of τ_{i2} is $\frac{C_{i2}}{T_i-C_{i1}}$. it has to be shown that:

$$\frac{C_{i2}-b}{T_i-C_{i1}-b} \le \frac{C_{i2}}{T_i-C_{i1}}$$

That is,

$$(C_{i2} - b)(T_i - C_{i1}) \le C_{i2}(T_i - C_{i1} - b)$$

Or,

$$-bT_i + bC_{i1} \le -bC_{i2}$$

Or,

$$b(C_{i1} + C_{i2}) \le bT_i$$

Which is always true because b is positive and$C_{i1} + C_{i2} \le T_i$.

We have now provided a solid foundation of what must be considered to prove the safeness of multiprocessor systems to satisfy the requirements verbalized by RMLS scheduling algorithm.

Theorem 1.

If effective utilization of each of two processors p_k and p_{k+1} which share a task τ_i, is not greater than Liu and

Layland's bound, then both processors will always safely run their corresponding tasks and subtasks.

Proof.

This theorem is similar to Lemma 2 in which it is assumed that there will be no conflict-idle period. However, here, this restriction is removed. In Lemma 2, it is mentioned that processor p_{k+1} does not have any influence on the execution of tasks and subtasks assigned to processor p_k. Since Liu and Layland's bound is satisfied for p_k it will always safely run its assigned tasks and subtask. In the packing algorithm (Fig. 1), the utilization of the shared task on processor p_{k+1} is computed as $\frac{C_{i2}}{T_i - C_{i1}}$ which, based on Lemma 3 and is the maximum utilization that τ_{i2} can always impose on the processor p_{k+1}.

On the other hand, the utilization of processor is taken to be less than or equal Liu and Layland's bound. Therefore, this processor will always safely run its assigned tasks and subtask, as well.

In each processor p_k (k=1,2,3,.., m_l), at most, there is only one task which is shared with the processor p_{k-1} (if k>2) and one task which is shared with processor p_{k+1} (if $k<m_l$). Using Theorem 1 twice, once for p_{k-1} and p_k and once for p_k and p_{k+1}, we can conclude that all processors would be safe with RMLS.

6. Simulations

Despite the fact that theoretical results such as speedup factors, play a prominent role to verify the schedulability performance, over the last years, many different empirical studies have also aimed to investigate the relative schedulability test performance amongst many different scheduling algorithms in the domain of real-time systems. Empirical studies also provide more general schedulability tests by focusing more on individual tasks parameters which yield to take into account non-specific task-sets. Task parameters such as the number of processors and tasks, the total utilization of the task-set, task deadline distribution and period and many others, would be very important especially for those techniques that will be employed in safety-critical industrial environments.

In the current section, the proposed method is compared with SPA2 [10]. We used UUnifast algorithm, a de facto standard proposed by Bini and Buttazo [18], to produce random task-sets without any bias. It is given that the n tasks' utilization (as random variables) are uniformly distributed between 0 and 1, and the sum of them is a specific value which is the total utilization of the system. The UUnifast algorithm uses the cumulative distribution function and generates task-sets with uniform distribution in O(n) time order. For further reading please refer to [18], [19].

6.1. Comparisons

To compare the results of packing, we used the method used by Burns et al [7]. In this method, task-sets are divided into some categories. For each category of task-sets, the result mean of that category is selected for comparison purposes. Task-sets in each category have the same overall utilizations and the same number of tasks. For example, there are 200 task-sets in the first category, with their overall utilization is equal to 4, and for this category, there are 16 tasks in each task-set.

Experiments were performed on 3000 randomly generated task-sets with a different number of tasks and different overall utilizations. Overall utilizations used are 4, 8 and 16. The number of tasks tested with each utilization is shown in Figures 3 to 5. For example, we compared the three methods SPA2, RMLS, and RMLS+DRM with task-sets and with the overall utilization of 8 so that task-sets contain 16, 20, 28, 44, and 76 tasks.

We allow RMLS algorithm to assign processors as needed and, for the SPA2 algorithm, we initially start with a high number of processors with which we are assured of the safety of the system. Then, we gradually reduce the number of processors one at a time until reaching the lowest number of processors in which the system is still in a safe mode. When the minimum number of processors needed for each method is found, the average utilization of all processors is calculated by dividing the overall utilization of each task set by the number of processors simply.

The primitive version of RMLS (represented by PRMLS in Figures 3 to 5) uses RM scheduler in all processors. Thus, the utilization of systems containing two processors must not exceed the higher bound of θ (2). Moreover, in that version, single tasks whose utilizations are greater than or equal to 0.83, were not separated to schedule each one on a single processor. The partitioning algorithm of PRMLS is a simplified version of the algorithm outlined in Fig. 1, in which all activities concerning Step 1 is removed. We would now like to compare primitive RMLS with SPA2 and RMDP.

The median utilization of SPA2 is either the same or lower than PRMLS. However, for equal medians, error bars indicate that the efficiency of PRMLS partitioning is better than SPA2. The average utilizations achieved for whole task-sets are 0.680 and 0.735 for SPA2 and RMLS respectively. This shows that the overall performance of Primitive RMLS (PRMLS) is more than 8% higher than that of SPA2.

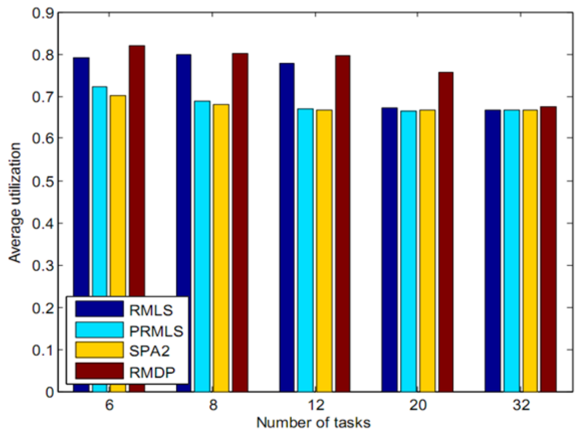


Fig. 3. Median of performance, by each method, for U=4

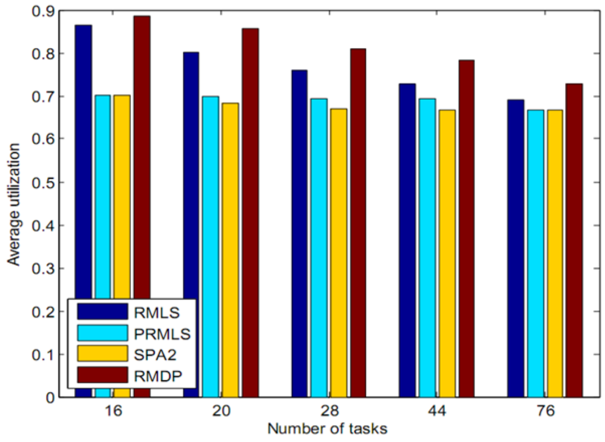


Fig. 4. Median of performance, by each method, for U=8

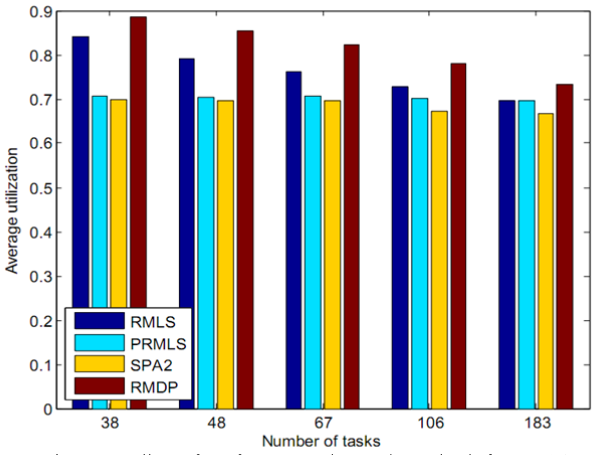


Fig. 5. Median of performance, by each method, for U=16

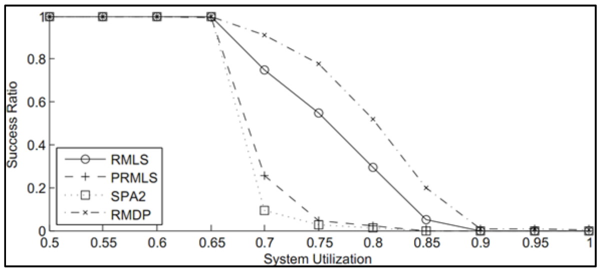


Fig. 6. Success rate for each method, for 3000 task sets

6.2. Discussion

Our experiments reveal that using DRM for two-task processors, greatly improvement the overall utilization of processors. The average utilization for RMLS on the randomly generated 3000 task-sets was 0.776. Medians, 25 and 75 percentiles are shown in Figures 3 to 5. An overall improvement of more than 14% as compared to SPA2 is a remarkable achievement for RMLS.

By comparing RMLS and PRMLS, one gets the impression that a little change can have a great performance improvement (i.e. close to 6%). Experiments, however, show better performance for RMDP method, which is caused by its scheduler. The scheduler of the RMDP method differs from rate-monotonic and, in actual fact, it is really more complex and involved; thus, its better performance is expected.

Although one can infer from Figures 3 to 5, that SPA2 usually should use a higher number of processors to safely schedule the same set of tasks as compared to both of PRMLS and RMLS, some complement charts are provided for visual comparisons (see Figures 7 to 9).

In addition, another experiment was conducted to test the schedulability of RMLS, PRMLS, SPA2, and RMDP on total 3000 randomly generated task-sets (see Fig. 6). In this experiment, we set different system utilizations and measure the ratio of task-sets that are schedulable. Taking a quick look at Figure 6, we clearly see that for all task-set with total utilization $U(\Gamma) <= 0.66$, all four algorithms can schedule every task-set with success ratio 1; however, for task-sets with total utilization greater than 0.66, SPA2 and PRMLS show their downside so that system total utilization of below 0.7, for instance, only 9% of randomly generated task-sets are schedulable using SPA2 whereas RMLS schedules about 79% of task-sets safely.

As was mentioned before, the high ability of RMDP to schedule task-sets is owing to its complicated algorithm. Additionally, please note that RMDP scheduler policy does not rate-monotonic.

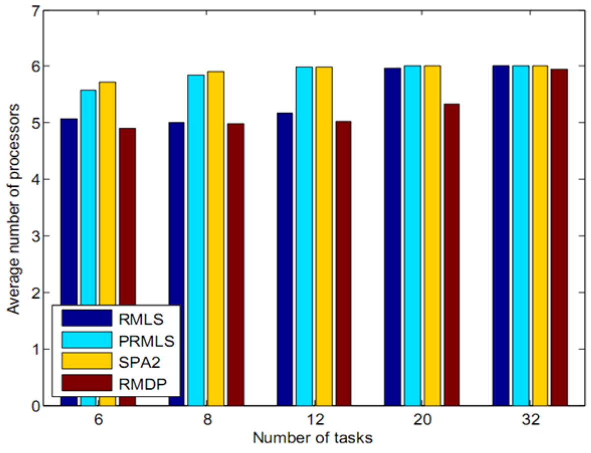


Fig. 7. Number of used processors for each method, for U=4

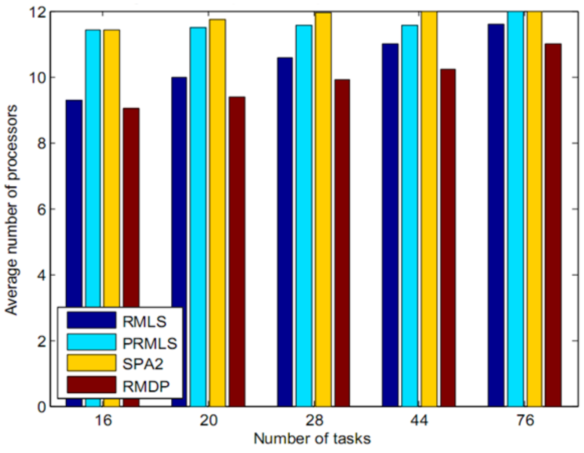


Fig. 8. Number of processors used for each method, for U=8

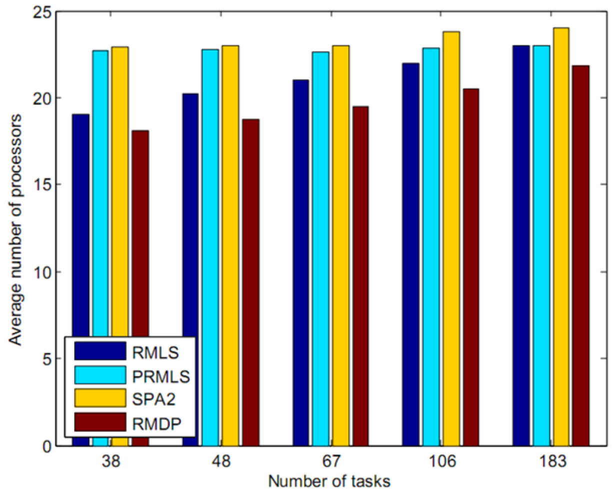


Fig. 9. Number of processors used for each method, for U=16

7. Summary and Future Work

Significant advances which have been made in many industrial areas are good evidence to support this claim that the importance of utilizing embedded multiprocessor system-on-a-chip (MPSoC) is an undeniable fact. In this context, regardless of many problems that must be considered, task management is a key issue which we focused on in the current article. Out of different approaches for hard real-time task scheduling, semi-partitioning of periodic tasks on multiprocessors was studied here in which the scheduler of each processor is rate monotonic, with the exception that the scheduler of processors with exactly two whole tasks is delayed rate monotonic (DRM) [16]. It was proven that when a task is split between two processors, if the utilization of the second part of the task is considered a little higher than its actual utilization, and RM scheduling policy is used on both processors while Liu and Layland's bound is satisfied, both processors run safely and all tasks meet their deadlines.

With this method, there is no need to define a release time for the second subtask. The Rate-Monotonic Least Splitting (RMLS) algorithm was developed and its performance was compared with the SPA2 algorithm as well as RMDP. It was concluded that the performance of PRMLS (Primitive RMLS) is more than 8% higher than SPA2 and the performance of RMLS is more than 14% higher than that of SPA2. This means that both PRMLS and RMLS usually need a fewer number of processors to safely schedule the same set of real-time tasks than SPA2, using a semi-rate-monotonic scheduler.

Although many types of research in the domain of semi-partitioned scheduling are being conducted, the use of new methods seems to be very important to improve real-time scheduling performance on multiprocessors. For example, authors in [20] did employ the equation of the line to dynamically assign priority to the tasks (called LTS) which appear to be an interestingly novel method in global multiprocessor scheduling. They claimed that their algorithm schedules all periodic task sets with total utilization up to 100% safely. One can be to modify (and improve) the LTS algorithm so that it is possible to use in semi-partitioned multiprocessor scheduling (e.g. the policy of each processor scheduler).

8. Acknowledgment

The authors wish to sincerely acknowledge the advice and support that they have received from Professor Mahmoud Naghibzadeh, the director of Knowledge Engineering Research Group (KERG) at Ferdowsi University of Mashhad. He generously provided insight and expertise that greatly assisted the research, even though he did not completely agree with all of the conclusions and interpretations of the current article.

References

[1] C.L. Liu, "Scheduling algorithms for multiprocessors in a hard real-time environment". *JPL Space Programs Summary,* vol. 37-60, 1969, pp. 28–31.

[2] K. Lakshmanan, R. Rajkumar and J. Lehoczky, "Partitioned fixed-priority preemptive scheduling for multi-core processors", *in Real-Time Systems, 2009. ECRTS'09. 21st Euromicro Conference on. IEEE*, pp. 239–248, 2009.

[3] M.R. Gary and D.S. Johnson: "Computers and Intractability; A Guide to the Theory of NP-Completeness" (W. H. Freeman & Co.), 1979.

[4] N. Guan, M. Stigge, W. Yi and G. Yu, "Fixed-priority multiprocessor scheduling with liu and layland's utilization bound", *in Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2010 16th IEEE. IEEE, pp. 165–174, 2010.

[5] B. Andersson and E. Tovar, "Multiprocessor scheduling with few preemptions", in Embedded and Real-Time Computing Systems and Applications, 2006. Proceedings. 12th IEEE International Conference on. IEEE, pp. 322–334, 2006.

[6] J. Anderson, V. Bud and U. Devi, "An edf-based scheduling algorithm for multiprocessor soft real-time systems", *in Real-Time Systems.(ECRTS 2005). Proceedings. 17th Euromicro Conference on, 2005,* pp. 199–208, 2005.

[7] Burns, R. I. Davis, P. Wang and F. Zhang, "Partitioned edf scheduling for multiprocessors using a c=d task splitting scheme", *Real-Time Systems,* vol. 48, no. 1, pp. 3–33, 2012.

[8] S. Kato and N. Yamasaki, "Portioned edf-based scheduling on multiprocessors", *in Proceedings of the 8th ACM international conference on Embedded software. ACM,* 2008, pp. 139–148.

[9] S. Kato, N. Yamasaki and Y. Ishikawa, "Semi-partitioned scheduling of sporadic task systems on multiprocessors", *in Real-Time Systems, 2009. ECRTS'09. 21st Euromicro Conference on.* IEEE, pp. 249–258, 2009.

[10] N. Guan and W. Yi, "Fixed-priority multiprocessor scheduling: Critical instant, response time and utilization bound", *in Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International. IEEE*, pp. 2470–2473, 2012.

[11] S. Kato and N. Yamasaki, "Portioned static-priority scheduling on multiprocessors", *in Parallel and*

Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on. IEEE, pp. 1–12, 2008.

[12] S. Kato and N. Yamasaki, "Semi-partitioned fixed-priority scheduling on multiprocessors", *in Real-Time and Embedded Technology and Applications Symposium,* 2009. RTAS 2009. 15th IEEE. IEEE, pp. 23–32, 2009.

[13] L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment", *Journal of the ACM (JACM)*, vol. 20, no. 1, 1973, pp. 46–61.

[14] R. I. Davis and A. Burns, "A survey of hard real-time scheduling for multiprocessor systems", *ACM Computing Surveys (CSUR)*, vol. 43, no. 4, p. 35, 2011.

[15] M. Naghibzadeh, P. Neamatollahi, R. Ramezani, A. Rezaeian and T. Dehghani, "Efficient semi-partitioning and rate-monotonic scheduling hard real-time tasks on multi-core systems", *in Industrial Embedded Systems (SIES), 2018 8th IEEE International Symposium on. IEEE,* 2013, pp. 85–88.

[16] M. Naghibzadeh, "A modified version of rate-monotonic scheduling algorithm and its' efficiency assessment", *in Object-Oriented Real-Time Dependable Systems, 2002. (WORDS 2002). Proceedings of the Seventh International Workshop on,* 2002, pp. 289-294.

[17] M. Naghibzadeh and K.H. Kim "The yielding-first rate-monotonic scheduling approach and its efficiency assessment", *International Journal of Computer System Science & Engineering*, pp. 173-180*,* 2003.

[18] E. Bini and G. C. Buttazzo, "Measuring the performance of schedulability tests", *Real-Time Systems,* vol. 30, no. 1-2, pp. 129–154, 2005.

[19] R. I. Davis and A. Burns, "Improved priority assignment for global fixed priority pre-emptive scheduling in multiprocessor real-time systems", *Real-Time Systems,* vol. 47, pp. 1-40, 2011.

[20] Ghavidel, M. Hajibegloo, A. Savadi and Y. Sedaghat, "LTS: Linear task scheduling on multiprocessor through equation of the line", *in Computer Architecture and Digital Systems (CADS),* 2015 18th CSI International Symposium on, pp. 1-6, 2015.

Journal of Computer and Knowledge Engineering, Vol. 1, No.1.
DOI: 10.22067/cke.v1i2.56005

β-sheet Topology Prediction Using Probability-based Integer Programming

Mahdie Eghdami*, Toktam Dehghani, Mahmoud Naghibzadeh

Abstract. β-sheet topology prediction is a major unresolved problem in modern computational biology. It is a challenging intermediate step toward the protein tertiary structure prediction. Different methods have been provided to deal with the problem of determining the β-sheet topology. Here, ab-initio probability-based methods called "*BetaProbe1*" and "*BetaProbe2*" are utilized to specify the β-sheet topology. In these methods, the stability and the frequency of β-strand pairwise interaction and β-sheet conformation are spotted. To predict more frequent interactions between β-strand pairs, besides pairwise alignment probability, the probability of occurring β-strand pairwise interaction is considered to compute the score of the interactions. Furthermore, to determine the β-strand pairwise alignment probability more accurately, a dynamic programming approach is utilized. In addition, the integer programming optimization is combined with the probabilities of β-strand pairwise interactions to determine the β-sheet topology. Moreover, the β-sheet conformation probability is considered to give better chances to more observed conformations for selection. Experimental results show that *BetaProbe1* and *BetaProbe2* significantly outperform the most recent integer programming-based method with respect to β-sheet topology prediction.

Keywords: β-sheet topology prediction; integer programming; dynamic programming; pairwise alignment;

1.Introduction

Proteins perform critical functions within the living organisms. Biologists believe that the functionality of proteins is determined by their tertiary structures. Therefore, it is important to specify the protein structure. Further, the conventional empirical methods to determine the structure of protein, namely, X-ray crystallography and Nuclear Magnetic Resonance (NMR) spectroscopy are very costly, time-consuming, and sometimes impossible. In addition, now, from the 30 million proteins with known primary structures in the protein databases [1], only the tertiary structures of 30 thousand of them have been determined by experimental methods [2]. Therefore, there is a huge gap between the number of known primary structures and the number of determined tertiary structures. Hence, insufficiency of empirical methods leads to utilizing computational methods in protein structure prediction problem.

Manuscript received May 16, 2016; accepted September 24, 2016.
Department of Computer Engineering, Engineering Faculty, Ferdowsi University of Mashhad, Mashhad, Iran.
*The corresponding author's e-mail is: eghdami.mahdie@mail.um.ac.ir

One of the most frequent elements in the protein structure is β-sheet which consists of separate sections known as β-strands. β-strands are typically six to eight amino acids long [3] that interact with amino acids of other β-strands and make paired β-strands (partners). The interaction between two β-strands can occur in two different forms (parallel or anti-parallel) depending on their orientation given by the position of the β-strands' N- and C-termini [4]. Each amino acid in a β-strand can make at most two hydrogen bonds with other ones in the paired stand. The interactions between the amino acid residues of the paired β-strands are known as a β-contact map.

β-sheets can be open or closed. Open β-sheets have two edge strands and they are the most common types of β-sheets. Fig. 1 shows an example of an open β-sheet type, where four β-strands interact. On the other hand, in the closed ones a circle is formed by a hydrogen bond between the first strand and the last one.

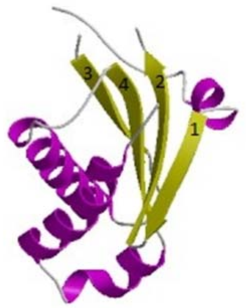


Fig. 1. Open β-sheet of a protein with PDB (Protein Data Bank) id 1NZ0D. β-strands that form the β-sheet are numbered in sequential order.

β-sheet topology prediction is regarded as one of the most important unresolved problems toward the tertiary structure prediction of proteins [5]. Correct prediction of β-sheet topology remains challenging because of hydrogen bond formations between linearly distant β-sheet residues [4]. Furthermore, the global covariations and constraints characteristic of β-sheet structures have not been well exploited [4].The β-sheet topology prediction provides valuable information for predicting protein three-dimensional structure [6], [7], designing new proteins and new drugs [8], [9] and determining folding pathways [10], [11].

The main goal of predicting β-sheet topology from the protein's amino acids is to determine the organization of β-strands in the β-sheets. This includes identifying β-strand members of each β-sheet and describing β-sheets by specifying paired β-strands and their interaction types. Further, β-contact maps are determined in β-sheet structure prediction. Different methods have been proposed to address the problem of predicting β-sheet topology which will be described in the next section.

In this article, we present *BetaProbe1* [12] and *BetaProbe2*, ab-initio probability based methods for β-sheet topology prediction. The main advantage of the proposed methods as compared to the previous researches is that we make use of the fact that more frequent and more stable conformations should have greater chances of being selected. For this purpose, the score of an interaction between each two β-strands is computed considering both pairwise alignment probability and pairwise interaction probability. Moreover, in order to make more accurate alignments, the β-strand optimum pairwise alignment is found using a dynamic programming approach. Furthermore, combining integer optimization with the β-strand pairwise interaction probability improves the accuracy of the predicted interactions. In addition, using β-sheet conformation probability in the last step of *BetaProbe1* leads to predicting more frequent and more stable conformations.

In the rest of this paper, first, related studies are reviewed in Section 2. Then, the details of the proposed methods will be described in Section 3. Finally, the performances of the proposed methods are compared with the most recent integer programming-based β-sheet prediction method in Section 4.

2. Related Work

Most β-sheet topology prediction methods utilize contact maps and strands alignment. Any improvement in the accuracy of these fields leads to a higher accuracy in determining the architecture of β-sheets. In this section first the related works in these fields are introduced. Then, some β-sheet prediction methods are explained.

Specifying the protein contact map is the first step in determining its final structure. Mainly, a contact map is expressed by a two-dimensional matrix. For two amino acids r_i and r_j, if the value of the i-th row and the j-th column ($0 \leq$contact Map (i, j) ≤ 1) is closer to one then they are more likely to interact with each other in the final structure. In other words, the likelihood of their relationship in the final structure of proteins is higher. NNcon [13], DNcon [14], SVMcon [15] and Distill [16] can be mentioned as contact map prediction methods. CMAPpro [17], PSICOV[18] and PhyCMAP [19] are the most recent methods which include contact map prediction.

So far, methods with high accuracy and acceptable execution time have been suggested for the sequence alignment problem. Further, pairwise sequence alignment is the most common technique used in β-sheet prediction methods. The most usual approach to determine the best alignment between two strands is dynamic programming.

Many efforts have been made to address the problem of predicting β-sheet topology. These works can be divided into two major categories: homology-based methods and ab-initio methods. The homology-based methods such as SMURF [20], SMURFLite [21], and MRFy [22] use homological information of proteins for recognizing their topologies. On the other hand, ab-initio methods only consider amino acids' pairing potentials and statistical information. In this article, we concentrate on the ab-initio β-sheet topology prediction methods. They utilize different approaches such as statistical potentials[23], information theory[24], Bayesian models and exploration of entire search space[25], linear programming [5], [26], [27], hidden Markov models [28], and graph matching algorithms [4]. These approaches can be divided into two major categories[29]: in one category, all possible β-topologies are enumerated, and a score for each complete β-topology is computed. Then, the β-topology with the highest score is selected as the best one [7], [25]. In the other category, in order to predict the β-sheet topology of a protein, pseudo-energy is assigned to each pair of β-strands. Then the problem of determining the best β-topology is reduced to maximizing the strand-to-strand contact potentials of the protein [5], [4], [26], [27], [28], [30].

BetaPro [4] was the first method to take into consideration the global nature of β-sheet topologies. In this method, three stages are used to predict β-topologies. Jones [31] takes advantage of linear programming to predict the secondary structure of the protein and β-sheet topologies. In [27], BetaPro was combined with linear programming to predict β-sheet topologies. Also, Rajgaria et al. [30] presented a method to determine the tertiary structure of proteins. In this method, strand pairing scores and contact maps are computed using linear programming. BetaZa[25] is a Bayesian approach which was introduced for proteins up to six β-strands. The conformational features were modeled in a probabilistic framework. The model is a combination of prior knowledge about β-strand arrangements with pairing potentials between the strands amino acid. Also, to select the optimum β-sheet architecture, using some heuristics, the search space was reduced. A dynamic programming was used to determine the β-strands optimum pairwise alignment. In the proposed dynamic programing, any number of gaps were allowed. As a result of exploration approach of the entire search space, BetaZa has a high time complexity. BeST [5] and BCov [26] predict the β-sheet topology using integer programming. BCov determines the β-sheet topology in three steps: first, it computes the residue contact propensity using PSICOV[18]; then, it computes the score of each possible β-strand pairing. Finally, an integer programming optimization is used to determine the β-sheet topology by finding the best solution according to the constraints and the pairing scores. In BCov two β-strands are paired only according to their alignment scores and the stability of conformations are not considered. Ruczinski et al. [7] showed that the arrangement of β-strands into β-sheets is not random. Based on the observations, there is a distinct pattern for β-strands arrangements. Some of the arrangements are unstable. Thus, they are never seen in nature. On the other hand, some particular orientations are more favorable than others. In addition, models for computing the probability of open β-topologies for proteins were derived. The discriminative power of these models is reduced significantly because the number of possible β-strand organizations increase exponentially and there is not sufficient training data to reliably represent such conformations. Therefore, these models are limited to proteins that contain at most ten β-strands. In this research, we try to improve BCov by considering the stability and frequency of β-strand pairing and β-sheet conformation.

3. Proposed Method

In this article, two efforts are made to resolve the problem of predicting β-sheet topology: *BetaProbe1* and *BetaProbe2*.These efforts can predict both β-sheet topology and β-contact map. As previously mentioned, in BCov[26] two β-strands are paired based on only their alignment score; but, Ruczinski et al. [7] showed that the organization of β-strands into β-sheets is not random and there is a distinct pattern. Therefore, to improve BCov, we attempt to give greater chances to more stable and more frequent conformations during the selection. In this section, first, a general description of each attempt is presented. Then, the steps of the proposed methods are described in detail.

3-1. First Effort: BetaProbe1

BetaProbe1 consists of three major steps: (i) in order to achieve more accurate alignments, a dynamic programming approach is used to compute the β-strand pairwise alignment probability. In addition, pairwise interaction probability of each pair of β-strands is computed according to [32]. Then, both pairwise alignment probability and pairwise interaction probability are utilized to compute the score of each interaction (ii) to determine the maximum total strand-to-strand contact potentials of the protein an integer programming optimization is used. In this step, to enforce more stable and more observed paired β-strands to be selected, pairwise interaction scores obtained in the previous step are utilized (iii) the best β-sheet topology is achieved according to paired strands determined in the previous step. To predict more stable conformations, β-sheet topology probabilities are considered. The pseudo code of *BetaProbe1* is illustrated in Pseudo code1.

Computing β-strand Pairwise Interaction Score: Many methods have been proposed to find the best alignment between sequences [33], [34]. Here we concentrate on an alignment method which is especially proposed for β-strands. In *BetaProbe1* the alignment probability of each two β-strands is computed based on the proposed method in BetaZa[25]. In this method, the Needleman-Wunsch algorithm [33][34] is used to compute the optimum alignment between each pair of β-strands in the parallel and anti-parallel directions. Then, the probability of the optimum alignment is computed by dividing the score of the best alignment by the sum of all possible alignments. To improve the accuracy of the alignments, the amino acid pairing potentials are used which are computed especially based on the β-amino acids.

Pseudocode 1: Probability-based algorithm for β-sheet topology prediction (*BetaProbe1*)

```
❖ Input: protein's strands
❖ Output: an open β-sheet conformation with the highest probability

❖ Step 1: Determining β-strand Pairwise Interaction Score
    for each pair of strands $s_i$ and $s_j$ do
        compute their parallel and anti-parallel pairwise alignment probabilities
        compute their parallel and anti-parallel pairwise interaction probabilities
        scores=alignment probability × interaction probability
❖ Step 2: Predicting the Closed β-Sheet Topology
    Solve the integer programming problem
❖ Step 3: Determining the Best Open β-Sheet Topology
    for each closed β-topology do
        for each interaction between two β-strands do
            Omit the interaction temporarily
            Compute the probability of the new open β-sheet
        Select the open β-sheet with the highest conformation probability.
```

To store the pairwise alignment probability, a matrix called "PAP (Pairwise Alignment Probability)" with n rows and 2n columns is defined. In this matrix, n is the number of β-strands in the protein. Matrix PAP is defined as follows:

$$\mathrm{PAP}(i,j)=\begin{cases} S_{parallel}(s_i,s_j) & \text{if } i\leq n \text{ and } j\leq n \text{ and } j\neq i \\ S_{anti\text{-}parallel}(s_i,s_j) & \text{if } i\leq n,\ n+1\leq j\leq 2\times n \text{ and } j\neq n+i \\ 0 & \text{if } j=i \text{ or } j=n+i \end{cases} \quad (1)$$

In Equation (1), $S_{parallel}(s_i,s_j)$ represents the probability of optimum alignment between strands s_i, i=1,2,…,n, and s_j, j=1,2,…,n, where their interaction type is parallel. Also, $S_{anti\text{-}parallel}(s_i,s_j)$ represents the probability of optimum alignment between strands s_i, i=1,2,…,n, and s_j, j=1,2,…,n, where their interaction type is anti-parallel. The definition shows that the matrix PAP is divided into two sections with an equal number of columns. The left section is used to store the parallel alignment probabilities and the right section is used to store the anti-parallel ones. The Score matrix for the protein in Fig. 1 is shown in Fig. 2-(a). It is important to note that the alignment probability depends on the spatial ordering of strands [25]. Therefore, the score of the optimum alignment between non-bridge strands can be different. This is expressed in (2) and (3):

$$S_{parallel}(s_i,s_j)\neq S_{parallel}(s_j,s_i) \quad (2)$$

$$S_{antiparallel}(s_i,s_j) \neq S_{antiparallel}(s_j,s_i) \tag{3}$$

According to [32], some β-strand pairs are more stable and they are more frequently observed in nature, as compared to others. Based on this observation, matrix "PIP (Pairwise Interaction Probability)" is defined to store the pairwise interaction probabilities of β-strands. The models derived by [32] were used to compute these probabilities. Matrix PIP contains n rows and 2×n columns as defined in (4):

$$\text{PIP(i,j)}= \begin{cases} P_{parallel}(s_i,s_j) & if\ i\leq n\ and\ j\leq n\ and\ j\neq i \\ P_{antiparallel}(s_i,s_j) & if\ i\leq n,\ n+1\leq j\leq 2\times n\ and\ j\neq i+n \\ 0 & if\ j=i\ or\ j=i+n \end{cases} \tag{4}$$

In (4), $P_{parallel}(s_i,s_j)$ represents the probability of strands s_i and s_j to make a parallel interaction in the final structure based on the protein characteristics such as the helical status and the number of residues between each two beta strands. Similarly, $P_{antiparallel}(s_i,s_j)$ is the probability of strands s_i and s_j to make an antiparallel interaction. The spatial ordering of strands has no effect on the β-strand pairwise interaction probability. This is expressed in (5) and (6):

$$P_{parallel}(s_i,s_j) = P_{parallel}(s_j,s_i) \tag{5}$$

$$P_{antiparallel}(s_i,s_j) = P_{antiparallel}(s_j,s_i) \tag{6}$$

In Fig. 2-(b), the matrix PIP is computed for the protein 1NZ0D. Then the scores of interactions between each pair of β-strands are determined. In the computation of each score, both pairwise interaction probability and pairwise alignment probability is considered. To store the scores of the interactions, a bi-dimensional n×2n matrix called "Score" is introduced, where n is the number of β-strands in the protein. The matrix definition is declared in (7).

$$Score(i,j)=PAP(i,j)\times PIP(i,j)\ \ 1\leq i\leq n\ ,\ 1\leq j\leq 2\times n \tag{7}$$

In the matrix Score definition, the first n columns represent the scores of parallel interactions. Similarly, the last n columns show anti-parallel ones. It is important to note that the score of an interaction between two strands depends on their spatial ordering. The Score matrix is illustrated in Fig.2-(c) for the protein 1NZ0D.

Prediction of the Closed β-Sheet Topology: Unlike BCov, in the integer optimization problem the pairwise interaction probabilities are considered in order to predict more stable paired β-strands. In addition, the integer programming model of *BetaProbe1* is defined differently from the BCov's. As a result, the closed β-sheet topology is obtained by solving the integer problem in (8).

$$\begin{aligned} maximize: \quad & \sum_{i=1}^{n}\sum_{j=1}^{2\times n} Score(i,j)\ X(i,j) \\ subject\ to: \quad & c1: X(i,j)\in\{0,1\}\ \forall 1\leq i\leq n\ ,\ 1\leq j\leq 2\times n \\ & c2: X(i,j)+X(j,i)+X(i,j+n)+X(j,i+n)\in\{0,1\} \\ & \qquad \forall\ 1\leq i\leq n,\ 1\leq j\leq 2\times n \\ & c3: \textstyle\sum_{j=1}^{2\times n} X(i,j) \in\{0,1\}\ \forall 1\leq i\leq n \\ & c4: \textstyle\sum_{i=1}^{n}\big(X(i,j)+X(i,j+n)\big) \in\{0,1\}\ \ \forall 1\leq j\leq n \\ & c5: \textstyle\sum_{j=1}^{2\times n} X(i,j) + \sum_{j=1}^{n} (X(j,i)+X(j,i+n)) \in\{1,2\} \\ & \qquad \forall 1\leq i\leq n \\ & c6: X(i,i)=X(i,i+n)=0\ \ \forall\ 1\leq i\leq n \end{aligned} \tag{8}$$

(a)

$$\text{PAP=}\begin{bmatrix} 0 & 0 & 0.138 & 0.019 & 0 & 1 & 0.047 & 0.007 \\ 0 & 0 & 0.023 & 0 & 1 & 0 & 0.019 & 0.089 \\ 0.15 & 0.025 & 0 & 0.238 & 0.05 & 0.015 & 0 & 0 \\ 0.027 & 0 & 0.437 & 0 & 0.015 & 0.055 & 0 & 0 \end{bmatrix}$$

$\longleftrightarrow$ parallel alignment score $\longleftrightarrow$ anti-parallel alignment score

(b)

$$\text{PIP =}\begin{bmatrix} 0 & 0.01 & 0.27 & 0.27 & 0 & 0.99 & 0.73 & 0.73 \\ 0.01 & 0 & 0.01 & 0.27 & 0.99 & 0 & 0.99 & 0.73 \\ 0.27 & 0.01 & 0 & 0.27 & 0.73 & 0.99 & 0 & 0.73 \\ 0.27 & 0.27 & 0.27 & 0 & 0.73 & 0.73 & 0.73 & 0 \end{bmatrix}$$

$\longleftrightarrow$ parallel interaction $\longleftrightarrow$ anti-parallel interaction

(c)

$$\text{Score=}\begin{bmatrix} 0 & 0 & 0.138 & 0.019 & 0 & 1 & 0.047 & 0.007 \\ 0 & 0 & 0.023 & 0 & 1 & 0 & 0.019 & 0.089 \\ 0.15 & 0.025 & 0 & 0.238 & 0.05 & 0.015 & 0 & 0 \\ 0.027 & 0 & 0.437 & 0 & 0.015 & 0.055 & 0 & 0 \end{bmatrix}$$

$\longleftrightarrow$ parallel interaction score $\longleftrightarrow$ anti-parallel interaction

Fig 2. (a) The matrix PAP for a protein with PDB ID 1NZ0D computed by the dynamic programming algorithm [25]. (b) The matrix PIP for protein 1NZ0D computed by using the pairwise interaction probabilities in [32]. (c) The matrix Score for protein 1NZ0D computed by considering both pairwise alignment probability and pairwise interaction probability in this paper.

X is a n×2n binary matrix (constraint c1) in which non-zero entries show an interaction between two related strands. c2 constraint shows whether the interaction between two strands is parallel or antiparallel. c3 and c4 constraints ensure that all strands have at most one strand partner on either side. Furthermore, each strand can pair with at least one and at most two other β-strands (constraint c5).

In Fig. 3, the matrix X and the predicted closed β-sheet topology for protein 1NZ0D are shown.

(a)

$$X = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\longleftrightarrow$ parallel $\longleftrightarrow$ anti-parallel

(b)

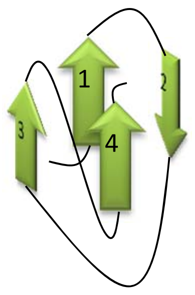


Fig. 3. (a) The matrix X obtained by solving the integer program. (b) The predicted closed β-sheet topology for the protein in Fig. 1.

Determining the Best Open β-Sheet Topology: In the previous step, paired β-strands and their interaction types are determined by the integer program solution. The predicted interactions make closed β-sheets, in other words, each strand has two partners. To extend the proposed method for the open β-sheets, the β-sheet topology probabilities determined by [32] are used. In this step, the probability of each possible open sheet is computed. Then the most probable one is selected as the best β-sheet topology. To enumerate all possible open β-sheets, one of the interactions of the closed one is omitted at a time. The process of determining the best open β-sheet topology is illustrated in Fig.4. In addition, Fig. 5 shows all possible open β-sheets for the closed one in the Fig. 3-(b).

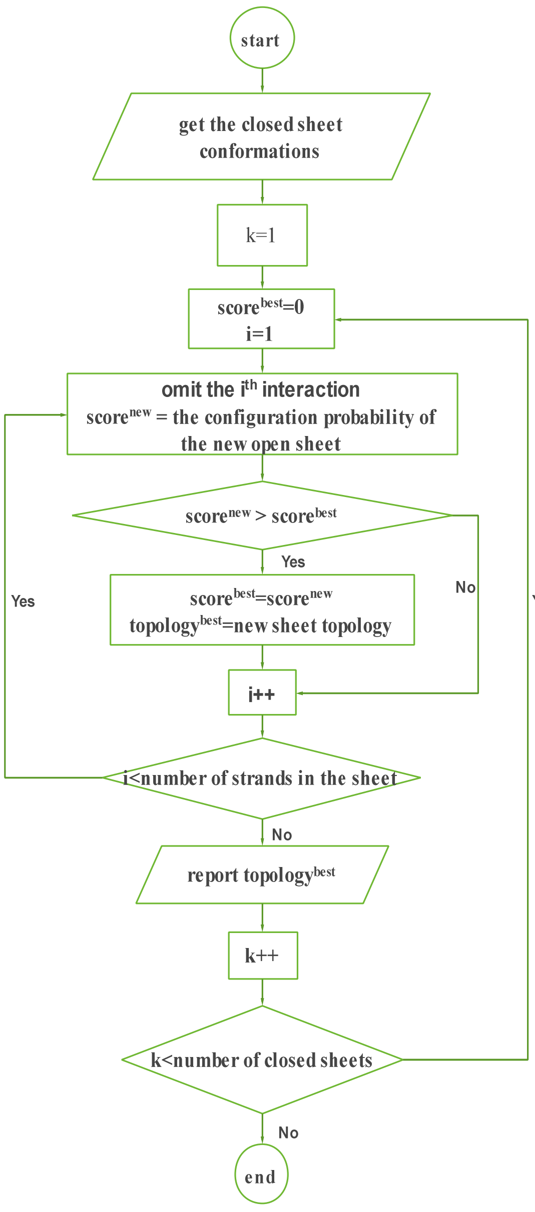

start
get the closed sheet conformations
k=1
score^best=0
i=1
omit the i^th interaction
score^new = the configuration probability of the new open sheet
score^new > score^best
Yes
No
score^best=score^new
topology^best=new sheet topology
i++
i<number of strands in the sheet
Yes
No
report topology^best
k++
k<number of closed sheets
Yes
No
end

Fig. 4. The process of determining the most probable open β-sheet

3-2. Second Effort: BetaProbe2

BetaProbe2 consists of two major steps: (i) similar to *BetaProbe1*, the score of each interaction is computed by considering both pairwise alignment probability and pairwise interaction probability. To obtain more accurate alignments, a dynamic programming approach is used to compute the alignment probability of each pair of β-strands (ii) to unravel the problem of determining the β-sheet topology, an integer programming optimization is introduced. Unlike *BetaProbe1*, the integer problem is defined to maximize the product of the interaction scores. In this step, both pairwise interaction probabilities and pairwise alignment probabilities are utilized to give a greater chance to more stable and more observed paired β-strands for selection. Unlike *BetaProbe1*, the β-sheet topology achieved by the integer program solution is not closed. The pseudo code of *BetaProbe2* is illustrated in Pseudocode 2.

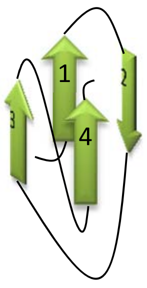
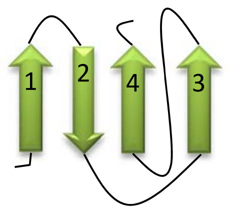
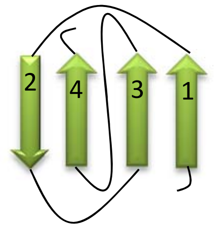
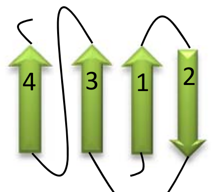
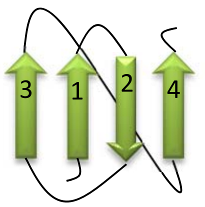
Closed β-sheet topology	Open β-sheet topology
	
	
	
	

Fig. 5. All possible open β-sheets from a closed one. The gray cell shows the best open β-sheet topology for protein 1NZ0D.

Pseudocode 2: Probability-based algorithm for β-sheet topology prediction (*BetaProbe2)*

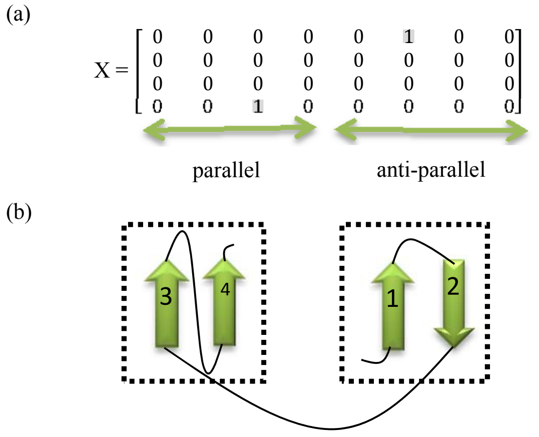
```
❖ Input: protein's strands
❖ Output: an open β-sheet conformation with the highest probability

❖ Step 1: determining B-strand pairwise Interaction Score
      for each pair of strands $s_i$ and $s_j$ do
          compute their parallel and anti-parallel pairwise alignment probabilities
          compute their parallel and antiparallel pairwise interaction probabilities
          scores=alignment probability×interaction probability
❖ Step 2: Prediction of the β-Sheet Topology
      for each pair of strands $s_i$ and $s_j$ do
          $scores^{new}$=log(scores)
      Solve the integer programming problem
```

Computing β-strand Pairwise Interaction Score: Similar to *BetaProbe1*, first the elements of matrices PIP and PAP are computed as in *BetaProbe1*. Then, the score of interaction between each pair of β-strands is determined.

Determining the β-sheet Topology: The problem of specifying the best β-sheet topology is reduced to an integer optimization. By assuming that the event of existing an interaction between two strands is independent of other β-strand interactions, the probability of the occurrence of several interactions is computed by the product of their probabilities. Therefore, an integer optimization is used to maximize the product of β-strand pairwise interaction scores, because each pairwise interaction score shows the probability of occurrence of an interaction between two strands according to the pairwise alignment probability and the pairwise interaction probability. Since pairwise interaction scores have positive values and the logarithm function is ascending, it is possible to maximize the sum of the logarithms of the pairwise interaction scores instead of maximizing their product. Then, the problem of determining the β-sheet topology becomes an integer linear problem represented in (9). Note that the constraints of the problem are the same as (8). In Fig. 6, the matrix X and the final β-sheet topology for protein 1NZ0D are presented.

$$
\begin{aligned}
&\textit{maximize:} && \sum_{i=1}^{n}\sum_{j=1}^{2\times n} \log(Score(i,j))\, X(i,j) \\
&\textit{subject to:} && c1: X(i,j)\in\{0,1\}\ \forall\, 1\le i\le n,\ 1\le j\le 2\times n \\
& && c2: X(i,j)+X(j,i)+X(i,j+n)+X(j,i+n)\in\{0,1\} \\
& && \qquad \forall\, 1\le i\le n,\ 1\le j\le 2\times n \\
& && c3: \textstyle\sum_{j=1}^{2\times n} X(i,j)\in\{0,1\}\ \forall 1\le i\le n \\
& && c4: \textstyle\sum_{i=1}^{n}\big(X(i,j)+X(i,j+n)\big)\in\{0,1\}\ \forall 1\le j\le n \\
& && c5: \textstyle\sum_{j=1}^{2\times n} X(i,j)+\sum_{j=1}^{n}(X(j,i)+X(j,i+n))\in\{1,2\} \\
& && \qquad \forall 1\le i\le n \\
& && c6: X(i,i)=X(i,i+n)=0\ \forall\, 1\le i\le n
\end{aligned}
\tag{9}
$$



$$
X=\begin{bmatrix}0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\end{bmatrix}
$$

Fig 6. (a) matrix *X* represent the result of solving integer programming problem. (b) the final predicted β-sheet topology

4. Results

In this section, first, the evaluation metrics and the data set are described. Then, the results of evaluating *BetaProbe1* and *BetaProbe2* are presented.

Evaluation metrics: To evaluate the performance of the proposed methods, well-known metrics in (10), (11) and (12) are used. These metrics have been used to evaluate state-of-the-art methods [5], [26], [25]:

$$Precision=\frac{TP}{Tp+FP}\times 100 \tag{10}$$

$$Recall=\frac{TP}{TP+FN}\times 100 \tag{11}$$

$$F1\text{-}score=\frac{2\times Precision\times Recall}{Precision+Recall} \tag{12}$$

Note that TP, FP, and FN represent true positives, false positives, and false negatives values, respectively.

Dataset: We used the BetaSheet916 set for the evaluation. This dataset is extracted from the PDB by [4]. It includes 916 proteins. To perform cross-validation, it is split into 10-folds randomly and evenly. DSSP program [35] is used for assigning the secondary structure. In this article β-residues includes: (1) the extended β-strands (shown by E in the DSSP) and (2) the isolated β-bridges (shown B in the DSSP output).

Cross validation: At each step in a cross-validation, one fold is considered as the test data and the remaining ones are the training set. Models are trained based on the training set. Predictions are determined in the test set. This process is repeated for all proteins in the original set. The accuracy measures are computed after the predictions are accomplished.

We carried out three simulations. In the first simulation, a 10-fold cross-validation experiment was performed on the BetaSheet916 for proteins with less than or equal to four β-strands and less than three partners. Similarly, in the second simulation, the proposed method was evaluated on proteins with less than or equal to five β-strands with less than three partners. The third simulation was performed on proteins with less than or equal to six β-strands with less than three partners.

BetaProbe1 and *BetaProbe2* were compared with the state-of-the-art method, BCov, which is also based on integer programming. For this purpose, in the first step of BCov, the residue pairing probabilities calculated in BetaPro were used. Then the methods were evaluated on the same data set.

In Table 1, the performance of *BetaProbe1* at the strand level is compared with the performance of BCov. The recall, precision, and F1-score measures are shown in this table.

Wilcoxon test for related samples has been utilized to determine whether there is a significant difference in the precision, recall, and F1-score of the two methods. To perform the test, the data set was broken into ten subsidiaries as declared in the Dataset section. After that, the results of *BetaProbe1* and BCov were evaluated for these subsets. The test showed that with an average error of 5%, there is a significant difference between the recall of the two methods at the pairing direction level for proteins with up to six and up to five strands. This means that the recall improvement of *BetaProbe1* compared with that of BCov is significantly meaningful. From Table 1, it can be concluded that besides using β-sheet conformation probabilities, considering pairwise interaction probabilities in the computation of β-strands interaction score and combining it with the integer programming greatly improves the accuracy of pairing directions. In Chart 1, Chart 2, and Chart 3 the recall, precision, and F1-score of *BetaProbe1* at pairing direction level is illustrated and compared to BCov's.

Table 1. The performance of *BetaProbe1* at strand level on proteins with 6 or fewer β-strands on BetaSheet916.

Evaluation level	Method	Recall	Precision	F1-score
strand pairing	BCov≤6 [a]	79	84	82
	BetaProbe1≤6	73	69	71
	BCov≤5 [b]	81	86	83
	BetaProbe1≤5	76	73	74
	BCov≤4 [c]	82	85	83
	BetaProbe1≤4	75	72	74
Pairing direction	BCov≤6	64	68	66
	BetaProbe1≤6	70	67	68
	BCov≤5	64	69	66
	BetaProbe1≤5	73	70	72
	BCov≤4	72	75	73
	BetaProbe1≤4	74	70	72

a)The evaluation is done on proteins with up to 6 β-strands
b) The evaluation is done on proteins with up to 5 β-strands
c) The evaluation is done on proteins with up to 4 β-strands

In Table 2, the performance of *BetaProbe2* is compared to BCov at strand level. For the three subsets of proteins, the precision and the F1-score measures of the proposed method at pairing direction level is better than BCov. Further, the precision of *BetaProbe2* is better than BCov at strand pairing level. Wilcoxon test for related samples showed that with an average error of 5%, there is a significant difference between the precision of BCov and *BetaProbe2* at the pairing direction level for all subsets of proteins. It can be concluded that the precision improvement of *BetaProbe2* is significantly meaningful as compared with BCov's.

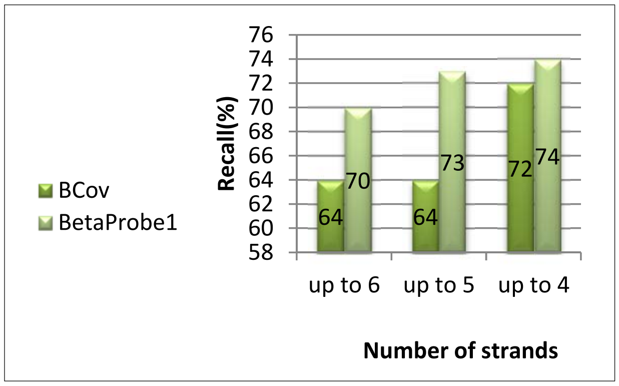


Chart 1: Recall comparison of *BetaProbe1* to BCov

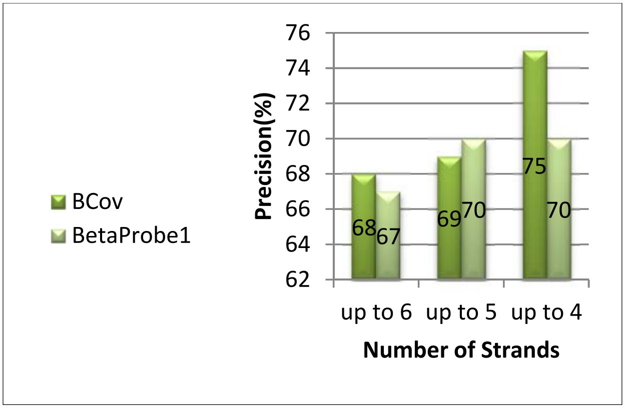


Chart 2: Precision comparison of *BetaProbe1* to BCov

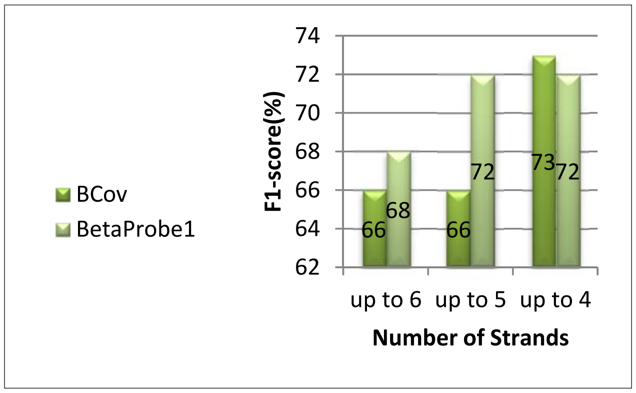


Chart 3: F1-score comparison of *BetaProbe1* to BCov

The reason for the improvement of proposed methods as compared with BCov is that adding pairwise interaction probabilities to the integer programming in the second step, enforces β-strand interactions which are more frequent in the nature to be selected with higher probabilities. In addition, to improve the pairwise alignments between β-strands, a dynamic programming approach is utilized in which gaps are allowed. Furthermore, using the amino acid pairing potentials provided by the BetaZa in the first step has improved the accuracy.

Table 2. The performance of *BetaProbe2* at strand level on proteins with 6 or fewer β-strands on Beta Sheet 916.

Evaluation level	Method	Recall	Precision	F1-score
strand pairing	BCov≤6	79	84	82
	BetaProbe2≤6	65	85	73
	BCov≤5	81	86	83
	BetaProbe2≤5	68	87	76
	BCov≤4	82	85	83
	BetaProbe2≤4	71	90	79
Pairing direction	BCov≤6	64	68	66
	BetaProbe2≤6	63	83	72
	BCov≤5	64	69	66
	BetaProbe2≤5	67	87	75
	BCov≤4	72	75	73
	BetaProbe2≤4	71	89	79

In Table3 and Table4 the results of *BetaProbe2* are compared to BetaZa at the residue level and strand level, respectively. The same alignment technique is used in both methods. BetaZa searches the entire search space to find the best β-sheet topology. Although the execution time of *BetaProbe2* is less than BetaZa, the precision of *BetaProbe2* is better at pairing direction level. In addition, the precision of *BetaProbe2* at strand pairing level and contact map level is comparable with BetaZa's. Comparing the recall, precision, and F1-score measures of *BetaProbe2* at pairing direction level with the other method, the results are represented in Chart4, Chart 5, and Chart 6, respectively.

In Table5 and Table6 the performance of *BetaProbe1* and *BetaProbe2* are represented at the residue level and strand level, respectively. As mentioned before, in *BetaProbe1*, the sum of the interaction scores is maximized in the integer programming step while in *BetaProbe2* the product of the interaction scores is maximized. It leads to predicting fewer interactions between β-strands in *BetaProbe2* because the scores of the interactions are in the range of zero and one. Subsequently, the predicted pairwise interactions are the most frequent ones. Therefore the precision of predicted interactions increases while the recall decreases.

Table 3. The performance of BetaProbe2 at strand level on proteins with 6 or fewer β-strands on Beta Sheet 916

Evaluation level	Method	Recall	Precision	F1-score
Contact map	BetaZa≤6	78	80	79
	BetaProbe2≤6	58	80	67
	BetaZa≤5	80	80	80
	BetaProbe2≤5	60	79	68
	BetaZa ≤4	82	82	82
	BetaProbe2≤4	61	80	69

Table 4. The performance of *BetaProbe2* at strand level on proteins with 6 or fewer β-strands on Beta Sheet 916.

Evaluation level	Method	Recall	Precision	F1-score
strand pairing	BetaZa≤6	83	84	84
	BetaProbe2≤6	65	85	73
	BetaZa≤5	87	88	87
	BetaProbe2≤5	68	87	76
	BetaZa ≤4	91	91	91
	BetaProbe2≤4	71	90	79
Pairing direction	BetaZa ≤6	80	81	81
	BetaProbe2≤6	63	83	72
	BetaZa ≤5	84	85	84
	BetaProbe2≤5	67	87	75
	BetaZa ≤4	88	88	88
	BetaProbe2≤4	71	89	79

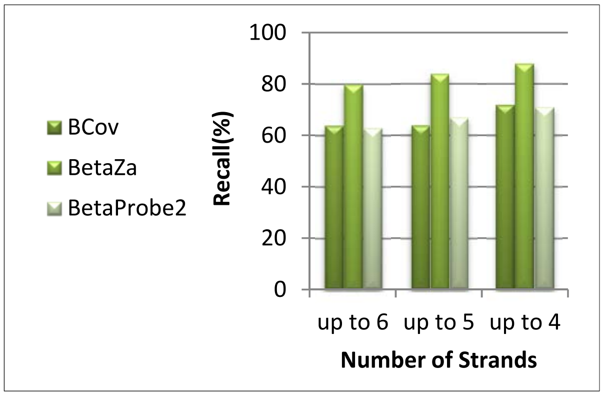


Chart 4: Recall comparison of *BetaProbe2* to other methods

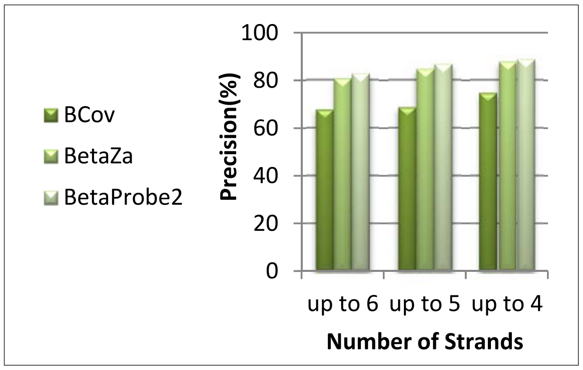


Chart 5: Precision comparison of *BetaProbe2* to other methods

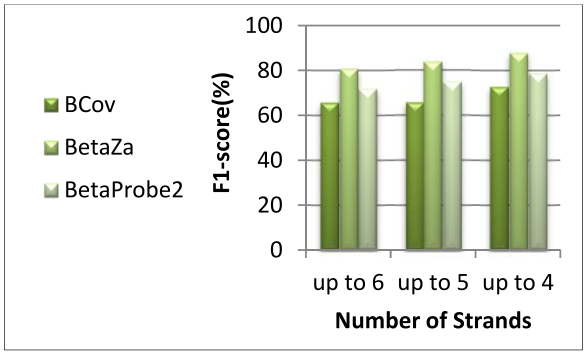


Chart 6: Precision comparison of *BetaProbe2* to other methods

5. Conclusion and Future Work

The issue of determining the topology of β-sheets is considered as a challenging problem. In this paper, *BetaProbe1* and *BetaProbe2*, two probability-based methods for the β-sheet topology prediction, are introduced. In these methods, first, the optimum pairwise alignment probabilities of β-strands are determined using the dynamic programming approach while any number of gaps are allowed. Then, the probability of the occurrence of an interaction is computed. After that, the score of on interaction is computed utilizing both pairwise alignment probability and pairwise interaction probability. Finally, we reduced the problem of finding the β-sheet topology to an integer optimization. 10-fold cross-validation experiments are performed to evaluate the proposed methods. The results show that these methods outperform the most recent integer programming-based method[26]. The major novelties in this research can be summarized as follow:

1. Considering both pairwise alignment probability and pairwise interaction probability to compute the score of an interaction between two β-strands;
2. Combining the probability of occurrence of an interaction with the integer programming;
3. Considering β-sheet conformation probability in the nature to predict more frequent β-topologies;
4. Considering the spatial ordering of β-strands in β-sheets in the integer programming;
5. The ability of the proposed methods to predict the β-sheet structure for proteins with multiple β-sheets;
6. The ability of the proposed methods to predict the β-sheet topology for proteins with closed β-sheets.

The performance of predictions can be improved even further. By combining residue pairing propensities with PSICOV [18] ones, the methods can become more accurate. Our methods can predict proteins with six or fewer β-strands with less than three partners. This can be extended to predict proteins with a higher number of β-strands and higher order partners by extending probabilities and adding new constraints to the integer programming step.

Table 5. The performance of *BetaProbe1* and *BetaProbe2* at residue level on proteins with 6 or fewer β-strands on Beta Sheet 916.

Evaluation level	Method	Recall	Precision	F1-score
Contact map	BetaProbe1≤6	63	66	64
	BetaProbe2≤6	58	80	67
	BetaProbe1≤5	64	66	65
	BetaProbe2≤5	60	79	68
	BetaProbe1≤4	64	67	65
	BetaProbe2≤4	61	80	69

Table 6. The performance of *BetaProbe1* and *BetaProbe 2* at strand level on proteins with 6 or fewer β-strands on Beta Sheet 916.

Evaluation level	Method	Recall	Precision	F1-score
strand pairing	BetaProbe1≤6	73	69	71
	BetaProbe2≤6	65	85	73
	BetaProbe1≤5	76	73	74
	BetaProbe2≤5	68	87	76
	BetaProbe1≤4	75	72	74
	BetaProbe2≤4	71	90	79
Pairing direction	BetaProbe1≤6	70	67	68
	BetaProbe2≤6	63	83	72
	BetaProbe1≤5	73	70	72
	BetaProbe2≤5	67	87	75
	BetaProbe1≤4	74	70	72
	BetaProbe2≤4	71	89	79

References

[1] J. Peng, “Statistical inference for template-based protein structure prediction,” Doctoral thesis, Toyota Technological Institute at Chicago, 2013.

[2] C. W. O’Donnell, “Ensemble modeling of beta-sheet proteins,” PhD thesis, Massachusetts Institute of Technology, 2011.

[3] M. J. Sternberg and J. M. Thornton, “On the conformation of proteins: an analysis of beta-

pleated sheets," *J. Mol. Biol.*, vol. 110, no. 2, pp. 285–296, 1977.

[4] J. Cheng and P. Baldi, "Three-stage prediction of protein β-sheets by neural networks, alignments and graph algorithms," *Bioinformatics*, vol. 21, no. suppl 1, pp. i75–i84, 2005.

[5] A. Subramani and C. A. Floudas, "Beta-Sheet Topology Prediction With High Precision and Recall for Beta and Mixed alpha/beta Proteins," *PLoS One*, vol. 7, no. 3, 2012.

[6] S. M. Zaremba and L. M. Gregoret, "Context-dependence of Amino Acid Residue Pairing in Antiparallel β-Sheets," *J. Mol. Biol.*, vol. 291, no. 2, pp. 463–479, 1999.

[7] I. Ruczinski, C. Kooperberg, R. Bonneau, and D. Baker, "Distributions of beta sheets in proteins with application to structure prediction," *Proteins Struct. Funct. Bioinforma.*, vol. 48, no. 1, pp. 85–97, 2002.

[8] T. Kortemme, "Design of a 20-Amino Acid, Three-Stranded -Sheet Protein," *Science*, vol. 281, no. 5374, pp. 253–256, Jul. 1998.

[9] B. Kuhlman, G. Dantas, G. C. Ireton, G. Varani, B. L. Stoddard, and D. Baker, "Design of a novel globular protein fold with atomic-level accuracy," *Science*, vol. 302, no. 5649, pp. 1364–1368, 2003.

[10] J. S. Merkel and L. Regan, "Modulating protein folding rates in vivo and in vitro by side-chain interactions between the parallel β strands of green fluorescent protein," *J. Biol. Chem.*, vol. 275, no. 38, pp. 29200–29206, 2000.

[11] Y. Mandel-Gutfreund, S. M. Zaremba, and L. M. Gregoret, "Contributions of residue pairing to β-sheet formation: conservation and covariation of amino acid residue pairs on antiparallel β-strands," *J. Mol. Biol.*, vol. 305, no. 5, pp. 1145–1159, 2001.

[12] M. Eghdami, T. Dehghani, and M. Naghibzadeh, "BetaProbe: A probability based method for predicting beta sheet topology using integer programming," in *Computer and Knowledge Engineering (ICCKE), 2015 5th International Conference on*, 2015, pp. 152–157.

[13] A. N. Tegge, Z. Wang, J. Eickholt, and J. Cheng, "NNcon: improved protein contact map prediction using 2D-recursive neural networks," *Nucleic Acids Res.*, vol. 37, no. suppl 2, pp. W515–W518, 2009.

[14] J. Eickholt and J. Cheng, "Predicting protein residue–residue contacts using deep networks and boosting," *Bioinformatics*, vol. 28, no. 23, pp. 3066–3072, 2012.

[15] J. Cheng and P. Baldi, "Improved residue contact prediction using support vector machines and a large feature set," *BMC Bioinformatics*, vol. 8, no. 1, pp. 113–121, 2007.

[16] D. Baú, A. J. M. Martin, C. Mooney, A. Vullo, I. Walsh, and G. Pollastri, "Distill: a suite of web servers for the prediction of one-, two-and three-dimensional structural features of proteins," *BMC Bioinformatics*, vol. 7, no. 1, p. 402, 2006.

[17] P. Di Lena, K. Nagata, and P. Baldi, "Deep architectures for protein contact map prediction," *Bioinformatics*, vol. 28, no. 19, pp. 2449–2457, 2012.

[18] D. T. Jones, D. W. A. Buchan, D. Cozzetto, and M. Pontil, "PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments," *Bioinformatics*, vol. 28, no. 2, pp. 184–190, 2012.

[19] Z. Wang and J. Xu, "Predicting protein contact map using evolutionary and physical constraints by integer programming," *Bioinformatics*, vol. 29, no. 13, pp. i266–i273, 2013.

[20] A. Kumar and L. Cowen, "Recognition of beta-structural motifs using hidden Markov models trained with simulated evolution," *Bioinformatics*, vol. 26, no. 12, pp. i287–i293, 2010.

[21] N. M. Daniels, R. Hosur, B. Berger, and L. J. Cowen, "SMURFLite: combining simplified Markov random fields with simulated evolution improves remote homology detection for beta-structural proteins into the twilight zone," *Bioinformatics*, vol. 28, no. 9, pp. 1216–1222, 2012.

[22] N. M. Daniels, A. Gallant, N. Ramsey, and L. J. Cowen, "MRFy: remote homology detection for beta-structural proteins using Markov random fields and stochastic search," *Comput. Biol. Bioinformatics, IEEE/ACM Trans.*, vol. 12, no. 1, pp. 4–16, 2015.

[23] T. J. P. Hubbard, "Use of beta-strand interaction pseudo-potentials in protein structure prediction and modelling," in *System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference on*, 1994, vol. 5, pp. 336–344.

[24] R. E. Steward and J. M. Thornton, "Prediction of strand pairing in antiparallel and parallel beta□ sheets using information theory," *Proteins Struct. Funct. Bioinforma.*, vol. 48, no. 2, pp. 178–191, 2002.

[25] Z. Aydin, Y. Altunbasak, and H. Erdogan, "Bayesian models and algorithms for protein β-sheet prediction," *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, vol. 8, no. 2, pp. 395–409, 2011.

[26] C. Savojardo, P. Fariselli, P. L. Martelli, and R. Casadio, "BCov: a method for predicting β-sheet topology using sparse inverse covariance estimation and integer programming," *Bioinformatics*, pp. 3151–3157, 2013.

[27] J. Jeong, P. Berman, and T. M. Przytycka, "Improving strand pairing prediction through exploring folding cooperativity," *IEEE/ACM Trans.*

Comput. Biol. Bioinforma., vol. 5, no. 4, pp. 484–491, 2008.

[28] M. Lippi and P. Frasconi, "Prediction of protein β-residue contacts by Markov logic networks with grounding-specific weights," *Bioinformatics*, vol. 25, no. 18, pp. 2326–2333, 2009.

[29] R. Fonseca, G. Helles, and P. Winter, "Ranking Beta Sheet Topologies with Applications to Protein Structure Prediction," *J. Math. Model. Algorithms*, vol. 10, no. 4, pp. 357–369, 2011.

[30] R. Rajgaria, Y. Wei, and C. A. Floudas, "Contact prediction for beta and alpha□beta proteins using integer linear optimization and its impact on the first principles 3D structure prediction method ASTRO-FOLD," *Proteins Struct. Funct. Bioinforma.*, vol. 78, no. 8, pp. 1825–1846, 2010.

[31] D. T. Jones, "Protein secondary structure prediction based on position-specific scoring matrices," *J. Mol. Biol.*, vol. 292, no. 2, pp. 195–202, 1999.

[32] I. Ruczinski, "Logic regression and statistical issues related to the protein folding problem," PhD thesis, University of Washington, 2000.

[33] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *J. Mol. Biol.*, vol. 48, no. 3, pp. 443–453, 1970.

[34] O. Gotoh, "An improved algorithm for matching biological sequences," *J. Mol. Biol.*, vol. 162, no. 3, pp. 705–708, 1982.

[35] W. Kabsch and C. Sander, "Dictionary of protein secondary structure: pattern recognition of hydrogen□bonded and geometrical features," *Biopolymers*, vol. 22, no. 12, pp. 2577–2637, 1983.

Focus and Scope

Papers on all aspects of Computer Science and Engineering, Machine Intelligence, and Knowledge Engineering are accepted for possible inclusion in the journal of **Computer and Knowledge Engineering** (CKE). It is for publishing the latest ideas and research that have not been published before and are not currently being considered for publication. CKE is oriented towards both theoretical findings and practical and industrial applications. All papers are reviewed in a peer-review process. CKE focuses on publishing in English only to make it usable by a large group of spectators. Although not restricted to the following list, some thematic areas are named in the following.

-Parallel and Distributed Processing
-Computer Networking
-Machine learning
-Software Engineering
-Internet of Thing (IoT)
-Pattern Recognition
-Image Processing
-Semantic Technology
-Image Processing
-Bioinformatics
-Computer Architecture
-Real-Time and Embedded Systems
-Computer and Network Security
Software-Defined Networking (SDN)

Information for Contributors

The editorial board of the journal of **Computer and Knowledge Engineering (CKE)** welcomes papers that report original research within the scope of the journal. The submitted papers should not be published before or currently be under the review process elsewhere. Extended versions of conference papers are exempted and will be reviewed, provided the copy right is honored.

The papers should be prepared according to the instructions provided below and submitted using the Web site of the journal.

The manuscript should have two columns of 8.5 cm width in each page and be prepared by the Microsoft Word software using the Times New Roman fonts. The following points should be considered in preparing the manuscript.

- The title of the paper should be concise, but expressive.
- The manuscript must contain an abstract of up to 200 words followed by up to 8 keywords.
- The main body of the paper should be divided into sections and subsections and also be numbered.
- The references should be numbered according to their appearance in the text, and the information about the references should be provided according to the IEEE standards.
- The figures and tables should be prepared in the final size, such that no enlargement or reduction is needed. The figures should have captions below and the tables should have headings above.
- The affiliations of the authors and the email address of the corresponding author should be provided at the footnote of the first page.
- To provide more information about the authors, it is recommended that a brief professional biography (not exceeding 150 words) and a photograph of each author be included at the end of the paper.

Table of Contents: