Egyetemi doktori (PhD) értekezés tézisei

DYNAMIC RESOURCE ALLOCATION USING A NEW HYBRID META-HEURISTIC ALGORITHM IN CLOUD

Seyed Majid Mousavi

Témavezető: Dr. Fazekas Gabor

DEBRECEN EGYETEM
Informatikai Tudományok Doktori Iskola
Debrecen, 2017.

Table of Contents

1. Introduction

Cloud computing is an emerging internet-based practice to provides computing as a utility service. The NIST[1] is defined Cloud Computing as[1]: "A model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction". This technology is a collection of thousands of computers interlinked together in a complex manner [2,3].

Cloud computing consists of five main specifications such as: On-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service. Moreover, cloud computing has different service models and four main deployment models.

1.1. Cloud service models

- *Software as a Service (SaaS)*. This service is a model for the distribution of applications over the Internet which is accessible for customers through a web browser or a program interface. Some of the well-known cloud software services are Google Apps, Salesforce, Office 365, Netflix, and etc.
- *Platform as a Service (PaaS)*. This service refers to a set of cloud computing services that provide a distributed platform to allow

[1] The National Institute of Standards and Technology (NIST) is a measurement standards laboratory, and a non-regulatory agency of the United States Department of Commerce. Its mission is to promote innovation and industrial competitiveness.

developers (Software developers, web developers and businesses) to build applications and services over the internet. Some of these services are AWS[1], Windows Azure, Google App Engine, and etc.

- *Infrastructure as a Service (IaaS).* This service is one of the main layers of cloud computing in order to provide virtualized computing resources over the Internet. Leading IaaS providers include AWS, Rackspace Open Cloud, and IBM SmartCloud Enterprise.

1.2. Cloud deployment models

- *Private cloud.* This model is a particular deployment model of cloud that provides a distinct and secure cloud environment which is used by only one organization over the Internet or a private internal network.
- *Community cloud.* Community deployment model refers to a shared cloud environment among several organizations from a specific group such as banks or heads of trading firms.
- *Public cloud.* Public cloud model is the standard cloud computing model which is the most recognizable model of cloud computing that cloud provider makes virtualized resources using pooled shared physical resources.
- *Hybrid cloud*. This model of cloud is a combination of other deployment models on-premises with multiple providers. Because of complexity in the most of the enterprises, they

[1] Amazon Web Services (AWS)

prefer to use the hybrid cloud solution where the advantage of each model (the public, private or on-premises infrastructure) supports a single application.

1.3. Resource allocation

In cloud environment the physical machines run multiple virtual machines (VM) which are presented to the clients as the computing resources. The architecture of a VM is based on a physical computer with similar functionality [4]. In fact VM is a guest program with software resources functioning similar to a physical computer [5,6,7]. Resource allocation technique is an important process to allocate resources based on user's application demands to achieve an optimal number of servers in use.

In cloud computing, there are two technical restrictions [8]. First, the capacity of the machines is physically limited; secondly, priorities for the implementation of the tasks should be in direction with maximizing the efficiency of resources. Ultimately, the waiting time and the completion time are to be reduced, in order to decrease the cost of system implementation [9]. Classical methods are very time consuming for achieving fully optimized solution and in some cases are impossible. Traditional approximate methods [10] are reported inconclusive and inaccurate for solving optimization problems and are often trapped in local optimum.

1.4. Load balancing

This process is done dynamically for the purpose of load balancing of non-preemptive tasks. Load balancing is an *NP*-hard optimization problem in cloud computing. This technique strives to balance the workload across VMs, which aims to minimize response time in order to keep promises and quality of service in accordance with service level agreements (SLA) between the clients and the provider. Furthermore, this process has to be carried out regularly due to the time-variant nature of the loads of Application Environments (AE). In fact cloud's clients are interested to have their jobs completed in the shortest possible time and at the minimum cost [11]. On the other hand, the cloud providers are interested to maximize the use of their resources with a lower overall cost to increase their profit.

Obviously these two objectives are in conflict and often they are not satisfied with the traditional methods of resource allocation and load balancing techniques [12,13].

2. Proposed method

Load balancing on virtual systems is influenced by various factors such as time and cost. Therefore, in order to optimize load balancing process in cloud computing, we need multi-objective optimization methods. These methods have to be solved in multi-dimensional spaces. Traditional approximation and load balancing methods due to the multi-objective and dynamic nature of the problem, and also difficulties in dealing with local optimum need advancement and major improvement [14].

There are many algorithms can be used for multi-dimensional problem-solving. Today, meta-heuristic algorithms play a pivotal role to solve multi-dimensional problems. Grey Wolf Optimizer (GWO) [15] is a multi-dimensional meta-heuristic algorithm. The hierarchical structure of grey wolves is mathematically modelled and used for design of optimization techniques. In this algorithm, the best solutions are calculated based on a multi-dimensional space and optimal solution is selected into a solution space. Grey wolf algorithm can be used for multi-dimensional problem solving but usually trapped in local optima.

Teaching–Learning-Based Optimizer (TLBO) [16] is also a meta-heuristic algorithm which is a robust algorithm to explore the space of a problem to find the settings or parameters to maximize a specific purpose. In virtualized systems, the performance of virtualized resources is dynamically changed based on their workload in time. The use of training-learning algorithms, provide better decisions on future choices. Unlike grey wolf algorithm, teaching–learning-based algorithm will not be trapped in local optima.

Our proposed method is a combination of these two multi-dimensional optimization algorithms to eliminate their weaknesses in order to make a new hybrid robust meta-heuristic algorithm. Proposed method integrates the ability of exploitation and exploration in grey wolf algorithm with the ability of the convergence and avoiding local optimum in teaching–learning-based algorithm.

One of the most important features of hybrid algorithm is that doesn't require any special controller, and only normal optimization parameters, such as population size, the number of iterations and so on are involved in its implementation, and this has led that it has the least dependency on the parameters. Proposed algorithm can improve approximate solutions into solution space for the resources allocation and load balancing in cloud computing.

2.1. Load balancing index

Load balancing index is calculated as the evaluation index as follows (*L* is the load balancing index and *1-L* is lack of load balancing):

$$B = a*(1-L)+b*C+c*T$$

The index shows the amount of load balancing between task completion time *T*, and cost of energy to perform the task *C* and resource productivity *B*. That in the cloud computing system is defined as above, their coefficients according to the cloud computing system (a, b, c) are subject to change. According to the importance of load balancing, *L* usually has the higher value, and coefficients *a*, *b* and *c* according to the type of system and the importance of the cost and response time will change. Finally, the aim is to minimize the index *B*.

When the number of jobs increases, the complexity rises and continues to extant that resources are wasted, load imbalance reaches its maximum, and a lot of resources will be wasted. For the same

period, T, and the expenditure increases. With reducing the number of resources by load balancing in virtual servers, energy expenditure reduces in cloud computing. Load balancing makes the response time to existing resources to be reduced as much as possible. In this study, in principle, we are looking for load balancing by reducing the number of servers in our resource allocation. This problem is a kind of bin packing problem which is difficult to solve.

There is a distributed network in a cloud environment with resource systems $S_1,S_2,S_3,\ldots,S_n$. The resources are ready to service in distributed network for various nodes. Different jobs are sent for the source systems by nodes. The overall goal of this system is that, an agreed scheduling on the resources allocation obtained to perform the jobs. Consequently with the resources allocation, the load balancing shall be increased [8]. Here the scheduler is responsible to allocate one or more Jobs to one or more artificial machines in a distributed system [9].

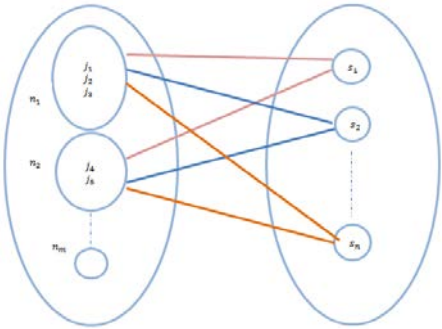

Figure 2-1- view of resource allocation in a distributed environment

Several jobs are allocated and processed in parallel with each other at time t in distributed system. The number of variables T_k is permutation between jobs and resources. This variable is called p, and its value is calculated as follows:

$P=n^m$ (n is number of tasks and m is the number of sources)

As it is described in figure 2-1 each node includes several jobs. Each job requires a series of specific resources. The problem can be introduced as follows:

$Job \rightarrow j_1,j_2,\ldots,j_n$

$Resource \rightarrow R_1,R_2, \ldots , R_m$

If in particular example, the resources $R_1,R_2, \ldots , R_m$ have the same capacity and the processing power of all is the same, and $j_1,j_2,\ldots,j_n$ needs 1% of the processor processing, the professional model can be defined in such a way that, what jobs use which resources to achieve maximum load balancing, average response time and minimum cost. For the exact solution of the problem, all possible allocation modes must be calculated to choose the best mode chosen. Due to the large number of modes (exponential), the problem is an example of bin packing problems which is *NP*-complete Type.

Optimization function is defined for resource i and the jobs j. y_i is the number of resources (package). x_{ij} represents the job and i is assigned to the resource j. C is the maximum capacity for each resource. w_i represents the amount of job i that is covered by the

resource. The objective function and mathematical programming model that should be optimized are as follows:

$$Min\ \left(B = a*\left(1-L_{(yj)}\right)+b*C_{(yj)}+c*T_{(yj)}\right)$$

S.t.

$$\sum_{i=1}^{n} w_i x_{ij} \le K y_j, \forall j \qquad x_{ij}, y_j = 0,1 \quad \forall i, j$$

$$\sum_{j=1}^{n} x_{ij} \le b_j, \forall j \qquad x_{ij} = 0,1 \quad \forall i$$

Where.

$$x_j = \begin{cases} 1 & job\ j\ is\ used \\ 0 & job\ j\ isnot\ used \end{cases} \qquad y_j = \begin{cases} 1 & resource\ j\ is\ used \\ 0 & resource\ j\ isnot\ used \end{cases}$$

The aim is to find the minimum number of virtual machines Y_j that minimize the objective functions. The values of L and C and T (load balancing, cost and response time) are calculated based on the number of virtual resources y_j. The variable of x_{ij} demonstrates that the i^{th} job is in j^{th} virtual machines, and if its value is equal to 0, it means that there is not any resource in j^{th} virtual machine and if its value is equal to 1, it means that there is enough resource to allocate the j^{th} virtual machine. Every job has capacity of w_i. The first limitation shows that total capacity of jobs can be placed at the maximum K available resources. The second limitation shows the maximum capacity of each virtual resource. b_j is the capacity of each virtual resource. The pseudo code of the proposed algorithm is presented in Figure 2-2:

```
Initialize the grey wolf population Xi=(i=1,2,...,n)
Initialize a,A and C
Calculate the fitness of each search agent
X1=the best Search gent
X2=the second best Search Agent
X3=the third best Search agent
While t<Max number of iterations)
   For each search agent
     Update the position of the current search agent by equation
     End for
Calculate the fitness of all search agents
Update X1,X2,X3
t=t+1
If not improve solution
  Begin
sol_wolf=Solution_grey wolf
      Initialize sol_wolf for initialize_solution for TLBO
      Sol_TLBO=Do TLBO with Initialize Population with sol_wolf
        Intialize the grey wolf population Xi= Sol_TLBO,
       Initialize a,A and C
          Calculate the fitness of each search agent
X1=the best Search agent, X2=the second best Search agent,
 X3=the third best Search agent
  end
end while
return X1
```

Figure 2-2 Pseudo code of the proposed algorithm

In the initial state, a series of random numbers with uniform distribution are considered as the initial population, and a basic solution is considered for the problem. Coefficients a, A, C are initialized.

Each wolf is considered as a solution. In other words, each wolf is considered as a solution to the problem. These solutions or wolves have an answer, and now the wolves are divided into three categories:

alpha, beta and gamma. And this is on the basis of the fitness function (objective), which one of them gives a better answer to the fitness function will be find. Then, we enter the main loop, so after a few iterations the best solutions for our fitness function. Based on the equations of the gray wolf algorithm, we update the wolves' position. This means that, according to the first class wolves, we fit the positions. And we consider more value for the probability of solution. And correspondingly, we value the wolves of beta and gamma classes and new positions of wolves' community and their classification can be obtained. Now, a new fitness function is calculated for the wolves and a division of three new wolves groups is calculated. If a suitable solution is found in the new classification, we improve the algorithm again. The best solution between the wolves is considered as the initial solution (initial population) for teaching and learning algorithm. Now, we solve the problem of teaching and learning algorithm and its solution is considered as initial population to start the gray wolf algorithm again and the gray wolf algorithm starts to implement again. It should be noted that in the gray wolf algorithm, every wolf represents a solution in the solution space, which the best solution will be chosen between them at any stage according to the position of other wolves. The best solution of the gray wolf is considered as the initial solution for teaching and learning. After learning and training, its output is implemented as the initial solution for the next iteration in the gray wolf algorithm.

2.2. Results

The proposed algorithm is benchmarked on 15 benchmark function [17]. And then to verify results, the results are compared with three above optimization algorithms (GWO, particle swarm optimization (PSO) [18] and Biogeography-based optimization (BBO) [19]). The results are shown in table 2-1 and 2-2.

Table 2-1 Result of benchmark functions in number of iteration 200 and population size of 30

Function	***Hybrid method***	***GWO***	***PSO***	***BBO***
sphere	*0*	*0*	*0.064355*	*0.045546*
Chung Reynolds	*0*	*0*	*0*	*0.063455*
Schwefel 2/22	*0*	*0.049545*	*0.035231*	*0.024245*
Schwefel 2/21	*0.005634*	*0.015366*	*0.104434*	*0.223567*
Cube	*0*	*0.094653*	*0.073244*	*0.083556*
Dixon & Price	*0.064556*	*0.034444*	*0.042324*	*0.075743*
Griewank	*0.034567*	*0.043433*	*0.047651*	*0.124456*
Rosenbrock	*0.022345*	*0.022655*	*0.107431*	*0.144677*
Ackley	*0.014238*	*0.072762*	*0.052764*	*0.034357*
Rastrigin	*0.013251*	*0.094749*	*0.074321*	*0.073534*
Brown	*0.025231*	*0.030769*	*0.060328*	*0.053567*

Table 2-2 Result of benchmark functions in number of iteration 1000 and population size of 30

Function	***Hybrid method***	***GWO***	***PSO***	***BBO***
sphere	*0*	*0*	*0*	*0*
Chung Reynolds	*0*	*0*	*0*	*2.78E-04*
Schwefel 2/22	*0*	*0*	*6.34E-03*	*3.31E-04*
Schwefel 2/21	*0*	*0*	*0*	*0*
Cube	*0.052115*	*0.115553*	*8.37E-03*	*6.22E-04*
Dixon & Price	*0*	*8.68E-03*	*5.45E-02*	*7.82E-02*
Griewank	*0*	*0.014521*	*0.012366*	*0.001343*
Rosenbrock	*0*	*0.013483*	*0.010828*	*0.013231*
Ackley	*0.003451*	*0.081342*	*0.017007*	*0.005432*
Rastrigin	*0.004783*	*0.022348*	*0.010613*	*0.024389*
Brown	*0.004532*	*0.038901*	*0.007164*	*0.043265*

2.3. Normal distribution workload

In order to evaluate proposed algorithm using normal distribution workload, we used CloudSim simulator in two different environments. CloudSim has the capability to simulate for modeling cloud

computing in the homogeneous and heterogeneous environment. Cloudsim toolkit is an open source framework that enables developers to model and different layers of cloud computing infrastructure and application services.

Table 2-3 Cloudsim setting for homogeneous cloud

Entities	Parameter	value
User	Number of Users	20
CloudLet	Number of cloudlets	120-800
Host	Number of host	5
	RAM	4096MB
	Storage	250000MB
	Network bandwidth	10000
Virtual Machines	Number of VMs	12
	MIPS	
	RAM	1024MB
	VMM	Xen
	Operation System	Linux
	Number of CPUs	2
	Number of Datacenter	2

Table 2-4 Cloudsim setting for heterogeneous cloud

Entities	Parameter	value
User	Number of Users	20
CloudLet	Number of cloudlets	120-800
Host	Number of host	5
	RAM	25GB
	Storage	2TB
	Network bandwidth	10GB
Virtual Machines	Number of VMs	20
	MIPS	
	RAM	128MB to 15GM
	Bandwidth	128MB to 15GM
	VMM	Xen
	Operation System	Linux
	Number of CPUs	2
	Number of Datacenter	2

Table 2-3 and table 2-4 presents cloudlets, cloud users, virtual machines, and host and data center properties for two different homogeneous and heterogeneous cloud environments respectively. However we used this simulation environment for normal distribution workloads on our algorithm, but also the efficiency of the algorithm under uniformly distribution was investigated.

The comparison of throughput between proposed algorithm, GWO, PSO, and BBO is shown in figure 2-3 and figure 2-4 in heterogeneous and homogeneous respectively. This comparison is based on normal distribution and workloads in the cloud system. The measurement of this factor is calculated based on the number of the tasks which are done and input workloads.

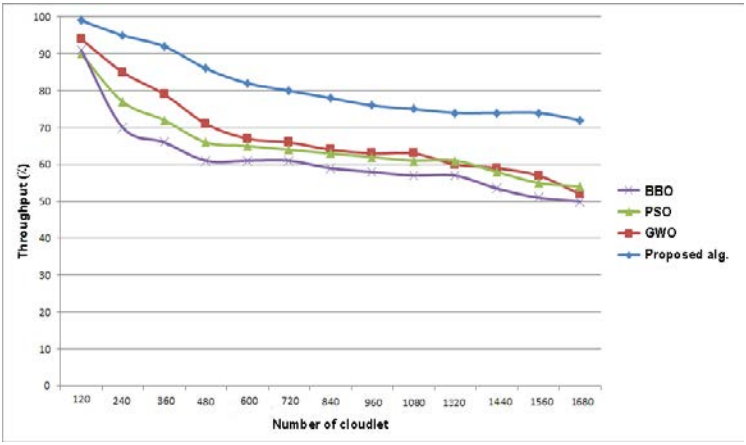

Figure 2-3 comparison of throughput in heterogeneous environment with normal distribution workloads

The results clearly illustrate the proposed algorithm outperforms in comparison other methods in both heterogeneous and homogeneous environments. And also the number of performed tasks in proposed algorithm is very high in comparison with other algorithms. The results reflect high efficiency in the proposed algorithm.

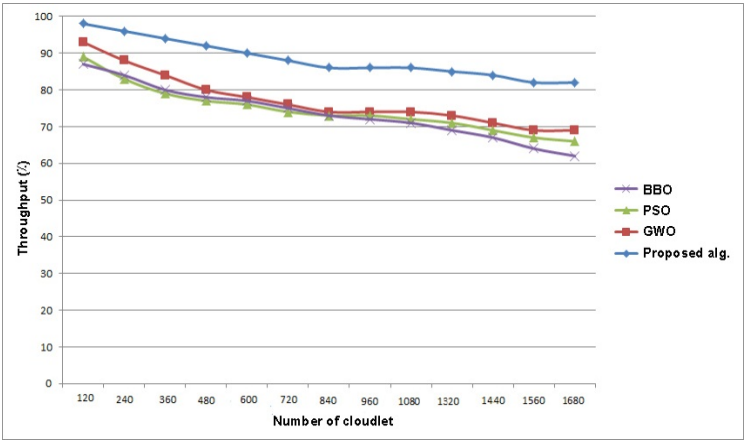

Figure 2-4 comparison of throughput in homogeneous environment with normal distribution workloads

3. Suggestions and future works

Our main focus in this study was on suggestion a new hybrid meta-heuristic algorithm to improve the performance of resource allocation in cloud computing environments. We believe that this study can be a considered as the starting point a significant variety of researches in the future, from which we mention some:

- Trying to extend this method on different distributed environment.
- Extension this method for workflows with dependent Jobs.
- Implementing of the algorithm on OpenStack platform.

References

[1] Hogan, Michael, et al: Nist cloud computing standards roadmap, NIST Special Publication 35, 2011.

[2] J. Yao and H. Ju-Hou: Load balancing strategy of cloud computing based on artificial bee algorithm, Information Management, 2012, pp. 185-189.

[3] S. Zhao, X. Lu, and X. Li: Quality of service-based particle swarm optimization scheduling in cloud computing", Networks, 2015, pp. 235-242.

[4] SM. Mousavi, F. Gábor: A novel algorithm for Load Balancing using HBA and ACO in Cloud Computing environment. International Journal of Computer Science and Information Security. 2016, 14(6), pp. 48.

[5] S. Zhang and Y. Zhou: Grey Wolf Optimizer Based on Powell Local Optimization Method for Clustering Analysis, Discrete Dynamics in Nature and Society, 2015.

[6] U. Ayesta, M. Erausquin, E. Ferreira, and P. Jacko: Optimal dynamic resource allocation to prevent defaults, Operations Research Letters, 2016, pp. 451-456.

[7] A. Abur, and A.G. Exposito: Power system state estimation: theory and implementation. CRC press, 2004.

[8] A. Khetan,V. Bhushan, and S. Gupta: A Novel Survey on Load Balancing in Cloud Computing, Journal of Engineering & Technology, 2013, pp.1-9.

[9] B. Cheng: Hierarchical cloud service workflow scheduling optimization schema using heuristic generic algorithm state Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, 2012, pp. 92-95.

[10] E. Emary, Hossam M. Zawbaa, Crina Grosan, and Abul Ella Hassenian: Feature subset selection approach by grey-wolf optimization. Industrial Advancement, Springer International Publishing, 2015 pp. 1-13.

[11] LD DB, Krishna PV: Honey bee behavior inspired load balancing of tasks in cloud computing environments. Applied Soft Computing. 2013 May 31;13(5), pp. 2292-303.

[12] P-Y. Yin, S-S. Yu, P-.P Wang, and Y-T. Wang: A hybrid particle swarm optimization algorithm for optimal task assignment in distributed systems. Computer Standards & Interfaces 28, 2006, pp. 441-450.

[13] R.C. Eberhart and J. Kennedy: A new optimizer using particle swarm theory. micro machine and human science, vol. 1, 1995, pp. 39-43.

[14] S. Xue: An improved algorithm based on NSGA-II for cloud PDTs scheduling, Journal of Software, 2014, pp. 443-450.

[15] S. Mirjalili, SM. Mirjalili, A. Lewis: Grey wolf optimizer. Advances in Engineering Software. 2014 Mar 31, 69, pp. 46-61.

[16] RV. Rao, VJ. Savsani, DP. Vakharia: Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Computer-Aided Design. 2011 Mar 31, 43(3):pp.303-15.

[17] Jamil, M., Yang, X. S: A literature survey of benchmark functions for global optimisation problems. International Journal of Mathematical Modelling and Numerical Optimisation, 2013, 4(2), pp.150-194.

[18] J. Kennedy: Particle swarm optimization. In Encyclopedia of machine learning, 2011, pp. 760-766. Springer US.

[19] D. Simon: Biogeography-based optimization. IEEE transactions on evolutionary computation, 2008, 12(6), pp. 702-713.

Registry number: DEENK/195/2017.PL
Subject: PhD Publikációs Lista

Candidate: Seyedmajid Mousavi
Neptun ID: C0BUCH
Doctoral School: Doctoral School of Informatics

List of publications related to the dissertation

Foreign language Hungarian book chapters (1)

1. **Mousavi, S.**, Fazekas, G.: Increasing QoS in SaaS for low Internet speed connections in cloud. In: Proceedings of the 9th International Conference on Applied Informatics January 29 - Februar 1, 2014. Eger, Hungary Volume I. [elektronikus dokumentum]. Ed.: by Kovács Emőd, Kusper Gábor, Kunkli Roland, Tómács Tibor, Eszterházy Károly Főiskola, Eger, Vol 1 195-200, 2015. ISBN: 9786155297182

Foreign language scientific articles in Hungarian journals (1)

2. **Mousavi, S.**, Mosavi, A., Várkonyi-Kóczy, A., Fazekas, G.: Dynamic resource allocation in cloud computing.
Acta polytech. Hung. 14, 80-101, 2017. ISSN: 1785-8860.
IF: 0.745 (2016)

Foreign language scientific articles in international journals (2)

3. **Mousavi, S.**: Cloud computing auditing roadmap and process.
J. Engin. Appl. Sci. "Accepted by Publisher", 2017. ISSN: 1816-949x.

4. **Mousavi, S.**, Fazekas, G.: Dynamic resource allocation in cloud computing: A survey and taxonomy.
J. Engin. Appl. Sci. "Accepted by Publisher", 2017. ISSN: 1816-949x.

Other journal articles (1)

5. **Mousavi, S.**, Fazekas, G.: A novel algorithm for load balancing using HBA and ACO in cloud computing environment.
IJCSIS. 14 (6), 48-52, 2017. ISSN: 1947-5500.

List of other publications

Foreign language abstracts (2)

6. **Mousavi, S.**, Mosavi, A., Várkonyi-Kóczy, A.: A load balancing algorithm for resource allocation in cloud computing.
   In: 16th International Conference on Global Research and Education. Proceedings, [Springer], [Cham], [1-8], 2017, (Advances in Intelligent Systems and Computing)
7. **Mousavi, S.**: A new architecture for Iran's communication Infrastructures for national Cloud Computing.
   In: 1st International Conference on New Research Achievements in Electrical and Computer Engineering (ICNRAECE). Proceedings, Curran Associates Inc., [Tehran], 1-5, 2017. ISBN: 1509027033

Total IF of journals (all publications): 0,745
Total IF of journals (publications related to the dissertation): 0,745

The Candidate's publication data submitted to the iDEa Tudóstér have been validated by DEENK on the basis of Web of Science, Scopus and Journal Citation Report (Impact Factor) databases.

27 June, 2017