A Load Balancing Algorithm for Resource Allocation in Cloud Computing

Seyedmajid Mousavi[1(✉)], Amir Mosavi[2], and Annamária R. Varkonyi-Koczy[3]

[1] Faculty of Informatics, University of Debrecen, Kassai Str. 26, Debrecen 4028, Hungary
majid.mousavi@inf.unideb.hu

[2] Kando Kalman Faculty of Electrical Engineering, Institute of Automation, Obuda University, Becsi Str. 94-96, Budapest 1431, Hungary
amir.mosavi@kvk.obuda-uni.hu

[3] Department of Mathematics and Informatics, J. Selye University, Elektrarenska cesta 2, 945 01 Komarno, Slovakia
koczy.annamaria@kvk.obuda-uni.hu

Abstract. Utilizing dynamic resource allocation for load balancing is considered as an important optimization process of task scheduling in cloud computing. A poor scheduling policy may overload certain virtual machines while remaining virtual machines are idle. Accordingly, this paper proposes a hybrid load balancing algorithm with combination of Teaching-Learning-Based Optimization (TLBO) and Grey Wolves Optimization algorithms (GWO), which can well contribute in maximizing the throughput using well balanced load across virtual machines and overcome the problem of trap into local optimum. The hybrid algorithm is benchmarked on eleven test functions and a comparative study is conducted to verify the results with particle swarm optimization (PSO), Biogeography-based optimization (BBO), and GWO. To evaluate the performance of the proposed algorithm for load balancing, the hybrid algorithm is simulated and the experimental results are presented.

Keywords: Cloud computing · Resource allocation · Optimization

1 Introduction

Cloud computing is an emerging technology and new trend for computing based on virtualization of resources [1]. 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 [2]. In fact VM is a guest program with software resources functioning similar to a physical computer. 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 [3]. 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

D. Luca et al. (eds.), *Recent Advances in Technology Research and Education*, Advances in Intelligent Systems and Computing 660, DOI 10.1007/978-3-319-67459-9_36

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 [4]. 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 [5]. The classical methods are very time consuming [6]. Traditional approximate methods are reported inconclusive and inaccurate and often trapped in local optimum [7]. Further algorithms proposed in literature for multi-objective scheduling e.g. FIFO [8] and Round-Robin [8] are in fact not effective in allocating the resources.

Therefore, In order to achieve maximum resource efficiency and scalability, exploring new meta-heuristic algorithms as well as development of novel algorithms are highly desired. Meta-heuristic optimization techniques have had an exceptional growth over the last two decades [9]. The remarkable ability of meta-heuristic techniques is motivated scientists from different fields to solve *NP*-hard problems. Furthermore, such techniques can often find optimal solutions with less computational effort than optimization algorithms, iterative methods, or simple heuristics. The question that arises is why this technique is remarkably common. The answer will be easily found in four main properties that characterize most meta-heuristics: simplicity, flexibility, derivation-free mechanism, and avoidance of entrapment in local optima.

Accordingly, this paper proposes a hybrid meta-heuristic load balancing algorithm with combination of two relatively new optimization algorithms, which can well contribute in maximizing the throughput using well balanced load across virtual machines and overcome the problem of trap into local optimum. The proposed algorithm is a hybrid of Grey Wolves Optimization (GWO) algorithm [10] and Teaching-Learning-Based Optimization (TLBO) algorithm [11]. The main idea is to integrate the ability of exploitation and exploration in GWO with the ability of the convergence in TLBO to provide a new population-base algorithm for dynamic allocation of virtual resources in cloud environment. Furthermore, the proposed algorithm well balances the priorities of tasks and effectively considers load balancing based on time, cost which consequently leads to minimal amount of waiting time of the tasks in the queue.

2 Related Works

Selecting and developing an appropriate algorithm to solve multi-objective problems is of utmost importance [12]. Therefore, meta-heuristic algorithms which have a global overview, as they ensure convergence to the solution and do not fall into the trap in local optimum, are of importance. Salimi et al. [13] introduce a multi-objective optimization algorithm for scheduling using fuzzy systems for load balancing in the distributed system. The authors aim at minimizing implementation time and costs while increasing the productivity of resources.

Cheng [14] provides an optimized hierarchical resource allocation algorithm using a general meta-heuristic algorithm. His model accomplishes workflow tasks scheduling aiming at load balancing with dividing the tasks into different levels. Gomathi and Karthikeyan [15] introduce a method for assigning tasks in a distributed environment using hybrid swarm optimization algorithm. The aim is to minimize the longest completion of task time among processors and creating load balancing. Pandey et al. [16] introduced a meta-heuristic method based on particle swarm optimization algorithm for scheduling on distributed environment resources. This optimization method is composed of two components, one of them is tasks scheduling operations and the other one is, using particle swarm algorithm (PSA) to obtain an optimal mix of the tasks to resource mapping. In this method, each particle represents a mapping of tasks to available resources. Traditional resource allocation methods [17] due to the multiple objectives and the dynamic nature of the problem and also difficulties in dealing with local optimum need advancement and major improvement. Consequently, the purpose of this paper is to address this research gap.

3 Proposed Method

The proposed methodology is based on bonding the algorithms of TLBO and GW. These two algorithms are currently used as approximation algorithm for establishing load balancing based on time and cost between resources and efficiency. With such hybridization it is aimed at speeding up the process while maintaining the improvement of local optimization and increasing the accuracy [18–20]. The TLBO and GWO algorithms are later introduced as the primary solutions to the described problem.

The optimization problem is described as a distributed network in a cloud environment with resource systems *S1, S2, S3, ..., Sn*. The resources are ready to service in the distributed network for various nodes. Different jobs are sent for the source systems by nodes. Here the scheduler is responsible to allocate one or more jobs to VM in a distributed system [21]. Scheduler provides a scheduling for resource allocation [22]. Several jobs are allocated and processed in parallel with each other at time t in the 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:

$$P = n^m \quad (n \text{ is number of tasks and } m \text{ is the number of sources}) \tag{1}$$

As it is described each node includes several jobs $j_1, j_2, \ldots j_n$. Each job requires a series of specific resources $R_1, R_2, \ldots R_m$. If in the particular example, the resources R_1, R_2, ... R_m have the same capacity and processing power and jobs j_1, j_2, ... j_n all need 1% of the processor processing, the professional model can be defined in the form of 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 and the best mode chosen. Due to the large number of exponential modes, the problem is an example of set packing problems which is of *NP*-complete type.

Optimization function is defined for resource i and job j. y_i is the number of resources. x_{ij} represents that job j is assigned to resource i. 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:

$$Min\, B = a * \left(1 - L_{(y_j)}\right) + b * C_{(y_j)} + c * T_{(y_j)} \tag{2}$$

$$\begin{aligned} &\sum_{i=1}^{n} w_i x_{ij} \leq \ Ky_j, \forall j \\ &\sum_{j=1}^{n} x_{i\,j} \leq \ b_j, \forall j \end{aligned} \qquad x_{ij}, y_i = 0, 1 \,\forall\, i, j \tag{3}$$

where $x_j = \begin{cases} 1 & job\ j\ is\ used \\ 0 & job\ j\ is\ not\ used \end{cases}$, $y_j = \begin{cases} 1 & resource\ j\ is\ used \\ 0 & resource\ j\ is\ not\ used \end{cases}$

The aim is to find the minimum number of virtual machines y_j that minimize the objective functions. The values of L, C, and T (load balancing, cost, and response time) are considered based on the number of virtual resources y_j, where a, b, c are variable based on cloud system. 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 no any resource in j^{th} virtual machine and if its value is equal to 1, it means that there is enough resource to allocate in 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.

3.1 Grey Wolf Algorithm

Mirjalili et al. [10] introduced GW for solving engineering problems. GW is a new optimization algorithm inspired by behaviour of grey wolves' hunting and their rule hierarchies. The hierarchical structure and social behaviour of wolves is modelled during hunting process in the form of mathematical models and is used for design of optimization algorithm.

3.2 Teaching-Learning-Based Optimization Algorithm

Rao et al. [11] introduced a novel approach to explore a problem space to find the optimal settings and parameters to satisfy the problem's objectives. This algorithm is inspired from modelling the teaching and learning problem mathematically and presents a new model for solving optimization problems. The algorithm operates in two phases, the first phase is the teacher share to develop class knowledge level and the second phase is the review of courses by students in the same class.

3.3 The Proposed Hybrid Algorithm

Given more convergence power in the global optimality, the GWO algorithm is used as base algorithm in the proposed algorithm. This algorithm can also perform multi-objective optimization [23]. The proposed algorithm is:

```
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 agent
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
```

The Steps are as Follows: In the initial state, a series of random numbers as the initial population are considered with uniform distribution and a basic solution is considered for the problem. Coefficients a, b, c are initialized. Each wolf is considered as a solution to the problem. In other words, each wolf is considered as a solution to the problem. These solutions or wolves have an answer. Wolves are divided into three categories; alpha, beta, and gamma. Yet on the basis of the fitness function (objective), one of them gives a better answer to the fitness function. Further, the solution enters the main loop where after a few iterations best solution for the fitness function is discovered. Based on the equations in the GWO algorithm, the wolves' position is updated. According to the wolves of first class, the new positions are fitted. Later on more values for the probability of solution are considered. Correspondingly, the values of beta and gamma classes, and new positions of wolves' and their classifications can be obtained. In addition, a new fitness function for the wolves and division of three new wolves groups is calculated. If a suitable solution is found in the new classification, the algorithm is to be improved further. The best solution between the wolves is considered as the initial solution (initial population) for teaching and learning algorithm. Further the problem using teaching and learning algorithm is solved and the solution is considered as initial population to start again. In this stage the GWO algorithm is implemented.

4 Computational Experiments

Here the main purpose is comparison on the performance of hybrid algorithm with three algorithms of GWO, particle swarm optimization (PSO) [24] and Biogeography-based optimization (BBO) [25]. At the first experiment, the mathematical models of algorithms are implemented using MATLAB (2014) and then it is run for 11 benchmark functions [26]. At the second experiment, the algorithms are simulated for resource allocation using CloudSim tool and the results are compared. Each algorithm has been investigated on a number of generations 200 and a constant population size of 50. It is noteworthy that the algorithms have been run 20 times on a benchmark function, and the final result has been obtained from the average of 20 times of running so that the rate of error decreases. Benchmark functions are divided into two groups: unimodal and multimodal.

Table 1. Results of benchmark functions for 200 generations and population size of 50.

	Function	Hybrid method	GWO	PSO	BBO
Unimodel	Sphere	0.007653	0.052347	0.064355	0.045546
	Chung reynolds	0.004355	0.062645	0.052134	0.063455
	Schwefel 2/22	0.008455	0.049545	0.035231	0.024245
	Schwefel 2/21	0.005634	0.015366	0.104434	0.223567
	Cube	0.002347	0.094653	0.073244	0.083556
	Dixon & price	0.064556	0.034444	0.042324	0.075743
Multimodal	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

According to the results presented in Table 1, the hybrid algorithm outperforms in comparison with all other algorithms in unimodal and multimodal functions. Computational results showed that concerning unimodal functions like sphere and Chang Reynolds, which are simple functions with no local optima, if we have many or few iterations or large or small population, hybrid algorithm outperforms other algorithms. This rule also applies to Schwefel 2/21 function because not only is it a simple function, but it also does not have any local optima. In conclusion, hybrid algorithm is the best algorithm to solve a problem for the simple functions that do not have any local optima. Regarding Schwefel 2/22 function, hybrid algorithm delivered better results than other algorithms. This function is a bit more complex than Sphere function. Therefore, hybrid algorithm could be used to solve a bit complex and problems that contain local optima. Hybrid algorithm outperforms in Restring and Ackley functions. These two functions also have many local optima same as Schwefel 2/22. In Griewank function, which is a rather complex function, hybrid algorithm still delivers the best results. Figure 1 shows an example of proposed algorithm using bin packing problem.

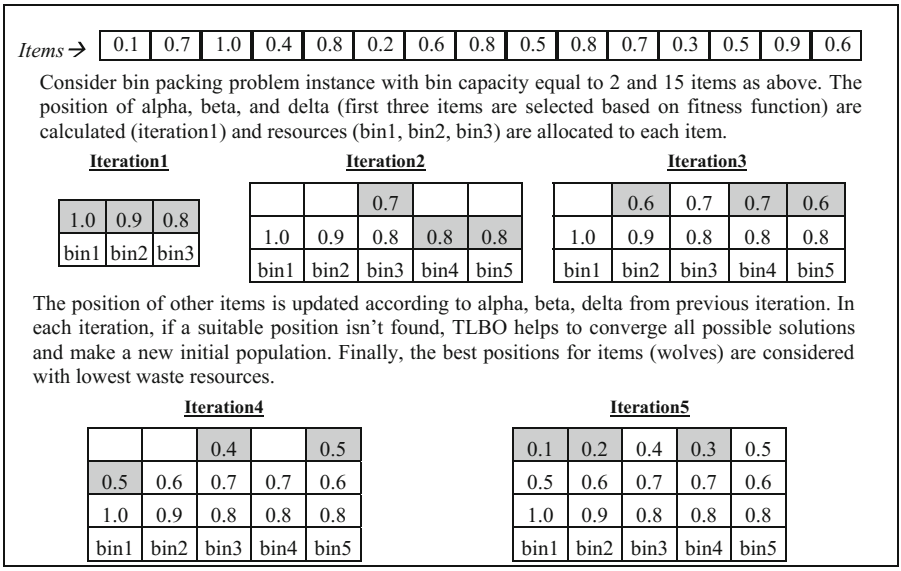

Fig. 1. An example of the execution process of proposed algorithm using bin packing problem

In addition to the following example the novel algorithm has been recently used in a number of industrial applications e.g. creating predictive decision models [27], and materials design innovation [28].

5 Conclusion

For an effective dynamic resource allocation in cloud computing a novel algorithm is proposed. The evaluation of experimental results indicates the novel hybrid approach has better performance comparing to the existing algorithms, in particular, in high-volume data of cloud scheduler. It is concluded that the main problem in the resource allocation of cloud scheduler is the lack of convergence in the optimal solution.

Acknowledgement. This work has partially been sponsored by the Research & Development Operational Program for the project "Modernization and Improvement of Technical Infrastructure for Research and Development of J. Selye University in the Fields of Nanotechnology and Intelligent Space", ITMS 26210120042, co-funded by the European Regional Development Fund.

References

1. Mousavi, S.M., Gabor, F.: A novel algorithm for Load Balancing using HBA and ACO in Cloud Computing environment. Int. J. Comput. Sci. Inf. Secur. **14**, 48–55 (2016)
2. Bertsimas, D., Gupta, S., Lulli, G.: Dynamic resource allocation: a flexible and tractable modeling framework. Eur. J. Oper. Res. **236**, 14–26 (2014)

3. Ayesta, U., Erausquin, M., Ferreira, E., et al.: Optimal dynamic resource allocation to prevent defaults. Oper. Res. Lett. **44**(4), 451–456 (2016)
4. Chen, Q., Zhang, X.: The local optimum in topology optimization of compliant mechanisms. Mech. Mach. Sci. **12**, 621–632 (2016)
5. Dhinesh, B., Venkata, K.: Honey bee behavior inspired load balancing of tasks in cloud computing environments. Appl. Soft Comput. **13**, 2292–2303 (2013)
6. Mosavi, A.: The large scale system of multiple criteria decision making. IFAC Proc. **43**, 354–359 (2010)
7. Selvaraj, C.: A survey on application of bio-inspired algorithms. Int. J. Comput. Sci. Inf. Technol. **67**, 366–370 (2014)
8. Mosavi, A., Varkonyi, A.: Learning in robotics. Int. J. Comput. Appl. **157**, 8–11 (2017)
9. Izakian, H., et al.: A discrete particle swarm optimization approach for Grid job scheduling. Int. J. Innov. Comput. Inf. Control **55**, 4219–4252 (2014)
10. Mirjalili, S., et al.: Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. Expert Syst. Appl. **47**, 106–119 (2016)
11. Rao, R.V.: Teaching-Learning-Based Optimization Algorithm, vol. 45, pp. 9–39. Springer (2016)
12. Mousavi, S., et al.: Dynamic resource allocation in cloud computing. Acta Polytech. Hung. **14**, 38–59 (2017)
13. Salimi, R., et al.: Task scheduling using NSGA II with fuzzy adaptive operators for computational grids. Parallel Distrib. Comput. **74**, 23–50 (2014)
14. Cheng, B.: Hierarchical cloud service workflow scheduling optimization schema using heuristic generic algorithm. Prz. Elektrotech. **88**, 92–105 (2012)
15. Gomathi, B., Karthikeyan, K.: Task scheduling algorithm based on hybrid particle swarm optimization in cloud computing. Appl. Inf. Techno. **55**, 33–38 (2013)
16. Pandey, S.: A particle swarm optimization-based heuristic for scheduling workflow applications in Cloud Computing. Inf. Netw. **76**, 400–407 (2010)
17. Mousavi, S., Mosavi, A.: A novel algorithm for cloud computing resource allocation. Open J. Cloud Comput. **5**, 123–144 (2017)
18. Mosavi, A., Vaezipour, A.: Developing effective tools for predictive analytics and informed decisions. University of Tallinn, Technical report (2014)
19. Mosavi, A.: Application of data mining in multiobjective optimization problems. Int. J. Simul. Multisci. Des. Optim. **5**, 15–28 (2014)
20. Mosavi, A., et al.: Combination of machine learning and optimization for automated decision-making. In: Multiple Criteria Decision Making MCDM, vol. 7 (2015)
21. Suleiman, M.H., Mustafa, Z., Mohmed, M.R.: Grey Wolf optimizer for solving economic dispatch problem. APRN Appl. Sci. **65**, 1619–1628 (2015)
22. Yagoubi, B., Slimani, Y.: Task load balancing strategy for grid computing. J. Comput. Sci. **8**, 186–194 (2007)
23. Malarvizhi, M., Uthariaraj, V.R.: Hierarchical load balancing scheme for computational intensive jobs in Grid computing. Adv. Comput. **14**, 97–104 (2009)
24. Kennedy, J.: Particle swarm optimization. Mach. Learn. **87**, 760–766 (2011). Springer
25. Simon, D.: Biogeography-based optimization. Evol. Comput. **12**, 70–79 (2008)
26. Jamil, M., Yang, S.: A literature survey of benchmark functions for global optimisation problems. J. Math. Model. Optim. **4**, 150–194 (2013)
27. Mosavi, A: Predictive decision model (2015). https://doi.org/10.13140/RG.2.2.21094.63047
28. Mosavi, A., Rabczuk, T.: Learning and Intelligent Optimization for Material Design Innovation. Theoretical Computer Science and General Issues. LION11 (2017)