

Adaptive League Championship Algorithm (ALCA) for Independent Task Scheduling in Cloud Computing

Anup Gade^{1*}, Mundukur Nirupama Bhat¹, Nita Thakare²

¹ VFSTR Deemed to be University, Vadalamudi 522213, AP, India

² Priyadarshani College of Engineering, Nagpur 440019, MS, India

Corresponding Author Email: gadeanup@gmail.com

<https://doi.org/10.18280/isi.240316>

ABSTRACT

Received: 12 March 2019

Accepted: 28 May 2019

Keywords:

meta-heuristic algorithms, LCA, makespan, cloud utilization, job scheduling, economy of scale, resource utilization

Scheduling is a heart of cloud computing as without appropriate scheduling it is impossible to get the desired results. Primary focus of this article is to focus on minimization of makespan, minimum utilization of resources and make the cloud services economic for an independent task. Out of the various task scheduling strategies, in last few years meta-heuristic algorithms have gained recognition in successful operation of task scheduling algorithms. League Championship based Algorithm (LCA) is fascinated from sports leagues through which best team/task in this case can be find out for scheduling. Task scheduling using Adaptive League Championship Algorithm (ALCA) is employed in this article and thereby it shrinks makespan, cloud utilization and cost. ALCA is implemented with cloudsim simulator using java as a programming tool and scheduling has followed the non-preemptive approach. Implementation of ALCA results reduction in makespan by 32.95 %, 20.99 % and 7.29 % against customary Ant Colony Optimization (ACO), Genetic Algorithm (GA) and Global League Championship Algorithm (GLCA) respectively. ALCA also reduces significantly cloud utilization value and improves economy of scale. ALCA may serve as preferred choice for cloud broker as it proved to be multipurpose in the area of makespan, resource utilization and economy.

1. INTRODUCTION

Cloud computing has evolved as a technological model in which user need not have to own any kind of resources, users will have to pay for only those resources which users will utilize. This paradigm of rented services like rented cab, electricity, aeroplane services, etc. has attracted cloud computing, commercial as well as small users. Cloud computing services are available with huge infrastructure which includes servers, infinite storage capacity, large scale of CPU's, memory, etc. Whenever it has been stated that it has infinite resources it actually doesn't mean infinite it has some limitations and from the perspectives of cloud service providers efforts are usually made to minimize resource utilization, particularly in case of peak time. Multi-tenancy, on-demand services and any service-any time are the features which makes cloud computing even more happening [1]. Maintaining these huge amount resources while providing guarantee of services is a tedious task. Due to popularity of cloud services multiple issues need to take care of, issues like resource management, load balancing, task scheduling, energy efficiency, economy and security requires critical attention to satisfy customer demands. One of the most crucial and vital responsibility in cloud computing is supposed to be task scheduling.

As task scheduling is NP-hard type of problem for which providing best solution is not possible hence sub-optimal solution is taken into consideration [2]. It is possible to provide sub-optimal solution only within polynomial time in case of NP-hard problem. Task scheduling can be broadly divided into three categories as heuristic algorithms, meta-heuristic

algorithms and hybrid [3]. Heuristic algorithms can be static or dynamic whereas meta-heuristic algorithms are broadly classified into nature inspired and swarm intelligence. Recently meta-heuristic algorithms has gained fair popularity few of them are Genetic Algorithm (GA) based on Darwin's theory of fittest of the survival, Ant Colony Optimization (ACO) giving optimized path to the ants searching for the food, Particle Swarm Optimization (PSO) motivated by communal behaviour of flock, BAT, Lion optimization algorithm, Cuckoo Search algorithm are also used popularly, League Championship Algorithm (LCA) analogous to the sports league played to find out the best/fittest team of the season. In case of task scheduling it gives best task to schedule which has smaller makespan.

Makespan can be roughly defined as finishing point (time) of last task in a group which need to be optimized, LCA has gained fair amount of results in terms of minimization of makespan time. LCA is an optimization algorithms based on sports league first proposed by Kashan [4]. Author has tailored it to the optimization of numerical function by proposing some idealized rules. This algorithm is applicable on sports league following the round robin time table. Applicability of LCA on task scheduling in cloud computing is depicted by Abdulhamid [5] but adaptivity in algorithm can make LCA even better as only minimization of makespan will not solve the purpose. This article provide scheduling algorithm which is adaptive in nature and along with the adaptivity it reduces cloud utilization for deriving it to be economic in nature.

An outstanding results given by this scheme when applied to the search space, motivates for further research in the vicinity of task scheduling in cloud computing. This article

presents novel idea of implementation of LCA in an adaptive manner with improved learning rate to minimize makespan time of the task under scheduling. Work presented in this article proved to be better than MINMIN, MAXMIN, GA, ACO, GBLCA algorithms after getting results from simulation using cloudsims. Proposed work also concentrated on minimization of cloud utilization through calculation of cloud utilization value and thereby it reduces cost by utilizing cloud resources for minimal amount of time.

The further sections will describe accordingly, section two emphasizes on related work, third section is about league championship algorithm and its description, fourth section gives proposed algorithm and experimentation, fifth section put forward the results of adaptive LCA and its comparisons with existing LCA and other meta-heuristic task scheduling algorithms. Conclusion and future scope of the article is given in sixth section.

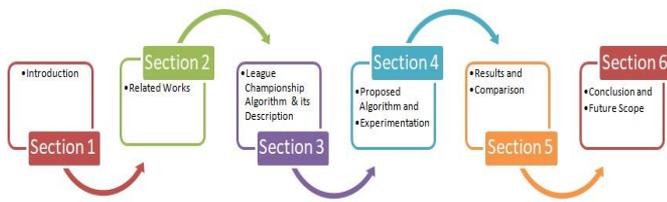


Figure 1. Organization structure of this article

2. RELATED WORKS

Task scheduling in cloud computing has gained remarkable attention of research community due to its importance in execution of cloud. Many researchers have applied heuristic algorithms and many more have applied meta-heuristic algorithms of different nature to get exceptional improvements in the existing work. Genetic Algorithm (GA) is a popular choice of research community as fitness function designing is important issue in it. Initially it considers all possible solutions as contestant a final solution to the problem, they termed it as chromosomes in case of GA. Only those chromosomes possessing particular fitness strength will go for next stage of operations. GA will have some iteration unless and until it provides best value (fittest solution) out of it by performing crossover and mutation kind of operations on it [2].

Tamanna Jena, et al. [1] used GA to find out best task-VM pair which will results in improvement of makespan and throughput. Regular FCFS policy is ignored by implementing shortest job first policy. Authors have given concept of Advance Research (AR) and Best Effort (BE) which are used for reserving the resources. AR type is designed for high priority tasks and it operates in non-preemptive mode whereas BE type task need to halt its execution if AR type task is arrived, it means that BE is preemptive in nature. Tasks in which deadline to follow strictly need to assigned to AR type. Algorithms emphasises on maintaining customer satisfaction rate by reducing waiting time.

Yujja et al. [6] applied GA for improved makespan and better load balancing by predicting execution time of task allocated to particular processor and thereby taking best suited decision over a group of tasks. Authors have implemented master scheduler which has complete view of system including processors information, data and workload related

details of CPU's. Time prediction model has statistic base of tolerable deviation. The success of the implementation relies highly on computation time required by GA. Shaminder Kaur, et al. [7] customized fundamental GA by using Shortest Cloudlet allocated to Fastest Processor (SCFP) and Longest Cloudlet assigned to Fastest Processor (LCFP) with stochastic rules to reduce computational intricacy and computing economics. It has a limitation that authors have compared their results with traditional GA only.

Aihong Liu et al. [8] proposed an improved version of basic ant colony optimization algorithm (ACO) for task scheduling in grid, they have implemented adaptive algorithm in basic ACO which gives optimal value of evaporation rate and thereby it increases its efficiency and load balancing rate. Results proved better than traditional ACO but for getting widespread acceptability it is an important factor to compare an improved version with other existing scheduling algorithms that particular part is missing in their article. Zehua Zhang et al. [9] implemented the fundamentals of network theory by using ant colony optimization for managing load balancing issue in cloud computing which proved to be beneficial for improving customer satisfaction and better facility utilization in cloud environment.

Particle Swarm Optimization (PSO) is a candidate solution whenever there is a discussion of task scheduling, the algorithm was proposed by Kennedy and Eberhart [10]. Algorithm has wide spread acceptability due to its simplicity and ability to provide effective solution to optimization problem. It considers all probable solutions as particles, every particle has allocated position and velocity. Particle used to attain best possible solution out of the available search space, local best solution will be provided by each particle (L_b). Out of the available particles having possessing best position and velocity will be treated as global best solution (G_b). After every iteration efforts are made by each particle to produce solution better than global best solution if it will be succeeded then this newly found out local best will replaces global best. Fahimeh Ramezani et al. [11] applied PSO in cloud atmosphere so as to maintain load balancing in task-oriented approach. Authors have implemented the concept of migration of overloaded task than migration of overloaded VMs. They have given model by using PSO algorithm for migration of overloaded task which ultimately reduces time required for load balancing. Zahra Pooranian et al. [12] inspires from the fact that PSO has outstanding global search ability but when it comes to local search it needs some support. This support is provided by Gravitational Emulation Local Search (GELS) technique which has strength of performing local optima. Given approach proved to be better in minimizing makespan and it provides number of those tasks which are unable to match their predicted completion time.

League Championship Algorithm (LCA) is a kind of evolutionary algorithm used as a solution to the optimization problem. It has been designed looking towards the philosophy of sports league by Kashan [4]. Algorithm has some predefined (idealized) rules based on these rules sports league is correlated with optimization problem like number of league will resembles with number of possible solutions, number of seasons will give stopping condition, fitness function is nothing but the team's strength of winning the match and change in combination of team resembling the operation of crossover and mutation from genetic algorithm. In basic LCA each team will have to play with each other in a round robbing fashion. If there are L numbers of teams in a league then there

will be $L(L-1)/2$ number of matches played to maintain round robin approach. Identifying the new combinations played an important role in defining the strength of the team at the same time knowing team's weakness is matter of strength both are internal to the team where as knowing opponents threat and converting it in to opportunities are external factors. This is commonly referred as SWOT analysis. Abdulhamid et al. [5] have applied LCA for task scheduling in order to minimize makespan, response time and economy of use in cloud computing. Authors have gained remarkable results as compared to popular meta-heuristic algorithms like genetic algorithm, ant colony optimization and traditional algorithms like minmin and maxmin. In their article authors have applied LCA by adaptation of SWOT algorithm in order to gain improved results. In this algorithm authors have not given value of Learning Rate (LR) as they have considered same LR for every iteration. The primary goal of this article is to come across with an optimal value of LR in order to minimize cloud utilization and thereby it condenses makespan and economy of cloud utilization.

3. LEAGUE CHAMPIONSHIP ALGORITHM (LCA) AND ITS DESCRIPTION

League Championship Algorithm (LCA) is a sports inspired techniques used to find winner or in case of task scheduling it can be considered as best task for scheduling. Traditional LCA is implemented by Kashan [4] for the first time by giving six idealized rules as:

1. It is obvious that team with higher playing strength will capture the game. Here "team's playing strength" means its capabilities to overrule other team.
2. The result of match (game) is not predictable, it is not unlikely that Indian cricket team will lose the game to Afghanistan cricket team in world cup.
3. There is fair probability that team i beats team j or vice-versa by considering both team's point of view.
4. The result will be either win or lose, there will be no tie in the game.
5. Team will focus on their coming matches only irrespective of their future matches. Formation will be based on results of earlier week(s).
6. If team i crushed team j , then there must be some positive points that makes team i to win has double shortcomings caused team j to lose.

Winner or loser determination is an important aspect of LCA and it will take place using stochastic approach. It is very natural that team with higher strength will have higher probability of winning. Let us assume teams i and j (job in case of scheduling) playing a match m with the team formations X_m^i and X_m^j having playing potential $f(X_m^i)$ and $f(X_m^j)$ respectively. Let P_i^m gives probability of team i winning with team j in match m , considering an ideal value

$$\frac{f(x_i^m) - \hat{f}}{f(x_j^m) - \hat{f}} = \frac{p_i^m}{p_j^m} \quad (1)$$

From LCA rules it can be derived that

$$P_i^m + P_j^m = 1 \quad (2)$$

Using Eq. (1) and (2), value of P_i^m can be derived as

$$P_i^m = \frac{f(x_i^m) - \hat{f}}{f_i^m + f_j^m - 2\hat{f}} \quad (3)$$

By using idealized rules stated above one number is generated in between 0 and 1, if this number is less than P_i^m then team i will have the probability of win and team j will lose else j will won and i will lose.

Global League Championship Algorithm [5] has implemented successfully with effective reduction in makespan. The algorithm is given below:

Initialize parameter of LCA
 Number of Leagues is represented as NL
 Number of Seasons is represented as NS
 Learning Rate is represented as LR

Step I:-

for season 1 to NS
 for league 1 to NL
 generate a set of VM's of different capacity randomly
 execute the task on VM's and find the fitness function (f)

$$f = \frac{\text{Total task to execute}}{\text{Capacity of VM's}} \quad (4)$$

whereas,

$$\text{Total task to execute} = \text{No. of tasks} * \text{time taken by each task} \quad (5)$$

$$\text{Capacity of VM's} = \text{MIPS of VM's} * \text{No. of CPU's} \quad (6)$$

Step II:-

Find the mean fitness (Mf) as

$$Mf = \frac{\sum f_i}{NL} \quad (7)$$

Find Threshold fitness (THf)

$$THf = Mf * LR$$

Step III:-

if
 $f_i > THf$
 then
 League need to be changed
 else
 League can play next season
 Repeat the steps for all seasons then
 Select the league with minimum fitness or maximum capacity with respect to total task.

Here, in case of global league scheduling algorithm it has been observed that authors have paid no attention towards Cloud Utilization Value (CUV) which can produce better learning rate.

4. PROPOSED ALGORITHM AND EXPERIMENTATION

4.1 Proposed algorithm

Step I:-

Initialize learning iteration (LI)

Step II:-

Generate a random value of learning rate (LR) for each iteration in LI

Step III:-

Run the LCA with this LR value and obtain the Cloud Utilization Value (CUV)

where

CUV= No. of repetition needed by the cloud VM's to complete the entire task set

Step IV:-

Repeat this LI iteration and select the value of LR which has minimum value of CUV

This value of LR will provide the optimal learning rate for minimal cloud utilization

Table 1. Parameter matching of LCA and traditional evolutionary algorithm (EA)

LCA	Traditional Algorithm (EA)	Evolutionary
League (<i>L</i>)	Population	
matches (<i>m</i>)	Iteration	
team (<i>i</i>)	<i>i</i> th member in population	
formation (<i>X_i^m</i>)	Solution	
winning strength <i>f(X_i^m)</i>	fitness value	
number of seasons (<i>S</i>)	maximum iteration	

Here, optimal learning rate is obtained by using adaptive LCA to minimize the cloud's makespan and ultimately cloud utilization which provides us economy of scale.

Based on idealize rules formed in earlier section LCA is able to identify which task is scheduled to which VM but before that it is necessary to understand the parameter matching of LCA and traditional evolutionary algorithm so that it would be easy to understand the implementation details of LCA. In LCA it has been cleared by using stochastic approach that team with higher playing strength will have higher probability of win as compared to other team. Table 1 shows parameter matching for ease of understanding in order to apply LCA to any of the optimization problem.

Taking above parameters matching into consideration adaptive LCA has been implemented according to the flow chart given in fig. 2 as shown below. Here optimization parameters used for implementation are reducing makespan, cloud utilization and economy of scale. Makespan is nothing but the completion time of last task in execution.

$$Makespan = \max\{F_i\} \tag{8}$$

where F_i denotes the finishing time of task i

4.2 Experimentation

Cloud utilization refers to the numbers VM's, CPU's and other resources were used to fulfil execution of designated tasks.

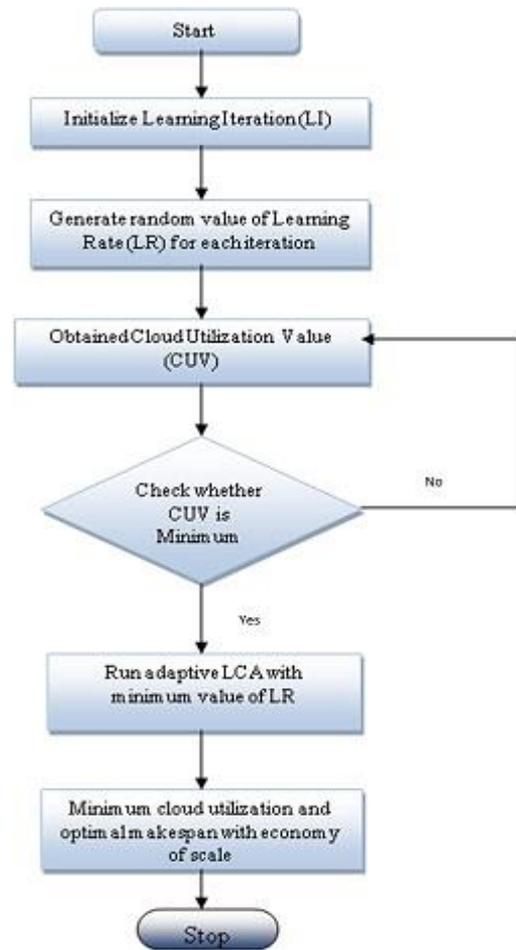


Figure 2. Flowchart of implementation details of adaptive LCA algorithm

Table 2. Parameters of scheduling algorithms under consideration

Sr. No.	Scheduling Scheme	Parameter Considered	Value
1.	GA	Population volume	1000
		Maximal iteration	1000
		Cross over rate	0.5
		Mutation rate	0.1
2.	ACO	Presence of ants in colony	10
		Evaluation factor ρ	0.4
		Pheromone tracking weight α	0.3
		Heuristic information weight β	1
		Pheromone updating constant Q	100
3.	GBLCA	Retreat constant Ψ_1	0.5
		Approach constant Ψ_2	0.5
		Rate of change p_c	0.01
		League size L	1000
4.	ALCA	Retreat constant Ψ_1	0.5
		Approach constant Ψ_2	0.5
		Cloud utilization value CUV	adaptive in nature(best value is considered)
		League size L	100

Along with minimization of makespan, the algorithm focuses on reducing cloud utilization value (CUV) which ultimately provides reduction in cost of execution of task on cloud. Algorithm is implemented by using CloudSim simulator platform popularly used for execution of cloud projects and eclipse editor using java as a programming tool. In order to determine the competency of proposed algorithm makespan, cloud utilization and economy are the parameters under consideration. Experiments are conducted repeatedly for 60 numbers of times and average of the same is considered for getting analogous results. Here table 2 shows some of the selected scheduling algorithm's parameter settings. Parameter settings of Genetic Algorithm (GA) are inspired from [13, 14] whereas in case Ant Colony Optimization (ACO) parameters are taken from [15, 16] and that of the same from GBLCA are derived from [5].

Task sets are taken from workload archive [17] having 43,800 jobs. The NASA Ames iPSC/860 log has made this task set available from their archive, moreover the intent behind using this data set is that it is available in universally accepted Standard Workload Format (SWF) and it has been acknowledged by CloudSim simulator. The workload encompasses the data which has the fields like CPU time, no. of jobs with its wait and run time, number of node used and required. Experimental parameters are considered as per the details given in table 3.

Table 3 Experimental parameters

Sr. No.	Entity	Parameter under Consideration	Values
1.	User	Number of users	1000
		Broker	2
2.	Task	Number of tasks	200-2000
		Length	1000000
		File Size	300
3.	Host	Memory Size	2048 MB
		Host Storage	1000000
		Bandwidth	10000
4.	Virtual Machine (VM)	Number of VMs	2
		Policy	Time or space shared
		VM RAM	512
		VMM	Xen
		Operating System (OS)	Linux
		Number of CPUs	10
5.	Data Center	Number of Data Center	2
		Number of Hosts	2

Makespan: - Makespan indicates the time at which last task finishes its execution. Reducing makespan is primary responsibility of any task scheduling algorithm. Minimum makespan time indicates minimum utilization of cloud i.e. minimum use of cloud which ultimately indicates economic use. As given in equation (8) above mathematically it can be written as

$$\text{Makespan} = \max\{F_i\}$$

where F_i denotes the finishing time of task i

Performance Improvement (PI) percentage: It can be defined as improvement in makespan for technique under consideration i with respect to other existing technique k . Percentage performance improvement is given by the equation (9) as shown below:

$$PI (\%) = (f_{max}(f_k) - f_{max}(f_i)) \times \frac{100}{f_{max}(f_i)} \quad (9)$$

5. RESULTS AND COMPARISON

Experimental results are computed for various task scheduling models in cloud computing as (MINMIN, MAXMIN, GA, ACO, GBLCA and ALCA) these techniques are used most popularly for task scheduling in cloud computing. Figure 3 represents the makespan calculated for above mentioned six techniques. Figure shows that there is significant increase in makespan as increased in number of tasks. From the results it has been cleared that ALCA is taking lesser makespan as compared to other techniques. It can be stated from the figure that MINMIN algorithm is taking maximum time for completion of its allocated tasks whereas ALCA algorithm is performing fairly better than GBLCA by updating its learning rate to minimum acceptable value.

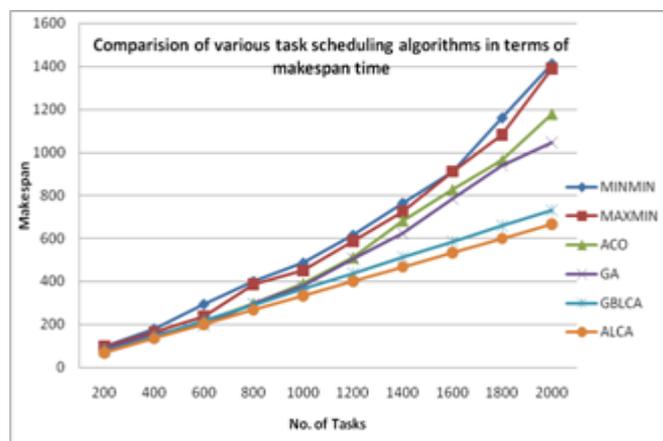


Figure 3. Makespan calculations using various scheduling techniques

The results show that ALCA is having minimum cloud utilization value which reduces the time period of cloud utilization. As it is clear that minimum cloud utilization will reduce cost of cloud services.

Table 4. Statistical picture after 60 trails of ALCA

No. of Tasks to be Executed	Best	Worst	Mean	Median	Mode
200	62.6438	83.004	72.837	73.571	73.571
400	125.287	166.009	145.675	147.142	147.142
600	187.931	249.013	218.513	220.712	220.712
800	250.575	332.018	291.351	294.283	294.283
1000	313.219	415.022	364.189	367.854	366.648
1200	375.863	498.027	437.027	441.425	441.425
1400	438.506	581.032	509.865	514.995	514.995
1600	501.150	664.036	582.703	588.566	588.566
1800	563.794	747.041	655.541	662.137	660.196
2000	626.438	830.045	728.379	735.708	735.708

In order to have statistical hold for the data obtained after performing experimentation repeatedly for 60 numbers of trails, inference can be drawn and it will be in a state to suggest robustness of proposed technique. Hence ALCA has been executed for different number of jobs numbered from 200 to

2000 and based on the results best (to showcase best value of makespan for particular number of tasks), worst (gives maximum makespan for given number of jobs), mean (average value come-out for mentioned number of tasks), median (will provide middle value in the list) and mode (most frequently resulted value) all are deliberated and its values are given in table 4. From the table it has been cleared that values calculated from the execution are quite close to each other. The experimentation results pursued the normal distribution and sturdiness of proposed scheme in optimization.

After performing multiple numbers of trails it is mandatory to calculate performance improvement of one strategy over another. Table 5 gives details of performance improvement of ALCA over remaining strategies under consideration in fact it provides comparative analysis of all techniques under consideration.

Table 5. Performance improvement in percentage for makespan time

Makespan Performance improvement in %	Name of Algorithm					
	MIN MIN	MAXMIN	ACO	GA	GBLCA	ALCA
	6295	5968	5326	4847	4298	4006
PI % over MINMIN	-	5.47	18.18	29.87	46.46	57.13
PI % over MAXMIN	-	-	12.05	23.12	38.85	48.97
PI % over ACO	-	-	-	9.88	23.92	32.95
PI % over GA	-	-	-	-	12.77	20.99
PI % over GBLCA	-	-	-	-	-	7.29

From the table it is clear that ALCA has 57.13 %, 48.97 %, 32.95 %, 20.99 % and 7.29 % improvement over MINMIN, MAXMIN, ACO, GA and GBLCA respectively. The result depicts that ALCA show outstanding performance for the issue of makespan minimization as compared to MINMIN, MAXMIN, ACO, GA and GBLCA task scheduling algorithms.

6. CONCLUSION AND FUTURE SCOPE

This article offered implementation of adaptive league championship algorithm on task scheduling of cloud computing model. Experimentation is performed using cloudsim simulator which is popularly used for simulation of cloud computing model. Results are compared with five existing algorithms which are used traditionally for task scheduling in cloud computing namely MINMIN, MAXMIN, ACO, GA and GBLCA. Comparison shows that proposed ALCA performed far better than MINMIN, MAXMIN and ACO algorithms where as it performs significantly better than GA and it gives marginally better result in comparison with GBLCA algorithm.

The primary objective of this paper is to reduce the makespan and offer the cloud services in economic manner. Result shows that ALCA proved to be optimal as compared to other techniques. The proposed algorithm is designed for independent tasks of non-preemptive nature only.

Authors are recommending two important aspects for future research first is prediction of load which will make it even more intelligent and responsive. There is always a probability that load may be increased suddenly and in that case algorithm must be able to handle the situation effectively. Machine learning approach will be helpful in order to predict the load and accordingly scheduling algorithm will be prepared for the execution. Second is normalization of various parameters under consideration will also be helpful in finding more appropriate results as whenever there are multiple sub-parameters associated with single parameter normalization of associated sub-parameters will make it more effective.

ACKNOWLEDGMENT

Gade Anup thanks both the research supervisors Dr. M. Nirupama Bhat and Dr. Nita Thakre for their time to time guidance and motivation. Also Gade Anup expresses his gratitude towards Head of Computer Science and Engineering Department Prof. Dr. Venkatesulu Dondeti for his support and cooperation.

REFERENCES

- [1] Jena, T., Mohanty, J.R. (2017). GA-based customer-conscious resource allocation and task scheduling in multi-cloud computing. Arab J Sci. Eng., 43(8): 4115-4130. <https://doi.org/10.1007/s13369-017-2766-x>
- [2] Kalra, M., Singh, S. (2015). A review of metaheuristic scheduling techniques in cloud computing. Egyptian Informatics Journal, 16(3): 275-295. <https://doi.org/10.1016/j.eij.2015.07.001>
- [3] Dubey, K., Kumar, M., Sharma, S.C. (2017) Modified HEFT algorithm for task scheduling in cloud computing. In Science Direct, Procedia Computer Science, 125: 725-732. <https://doi.org/10.1016/j.procs.2017.12.093>
- [4] Kashan, A.H. (2009) League championship algorithm: A new algorithm for numerical function optimization. 2009 International Conference of Soft Computing and Pattern Recognition, Published in IEEE Computer Society, pp. 43-48. <https://doi.org/10.1109/SoCPaR.2009.21>
- [5] Abdulhamid, S.M., Latiff, M.S.A., Abdul-Salaam, G., Madni, S.H.H. (2016). Secure scientific applications scheduling technique for cloud computing environment using global league championship algorithm. PLoS ONE, 11(7): e0158102. <https://doi.org/10.1371/journal.pone.0158102>
- [6] Ge, Y., Wei, G. (2010). GA-based task scheduler for the cloud computing systems. In: International Conference web Information System Mining, WISM 2010. pp. 181-186. <https://doi.org/10.1109/WISM.2010.87>
- [7] Kaur, S., Verma, A. (2012) An efficient approach to genetic algorithm for task scheduling in cloud computing environment. I.J. Information Technology and Computer Science, 10: 74-79. <https://doi.org/10.5815/ijitcs.2012.10.09>
- [8] Liu, A., Wang, Z. (2008). Grid task scheduling based on adaptive ant colony algorithm. In: International conference on Management E-commerce E-government Grid, pp 415-418. <https://doi.org/10.1109/ICMECG.2008.50>

- [9] Zhang, Z., Zhang, X. (2010). A load balancing mechanism based on ant colony and complex network theory in open cloud computing federation. In: 2nd International Conference on Industrial Mechatronics and Automation, pp 240–243. <https://doi.org/10.1109/icindma.2010.5538385>
- [10] Kennedy, J., Eberhart, R. (1995). Particle swarm optimization. Proceedings of ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia, Australia. <http://dx.doi.org/10.1109/ICNN.1995.488968>
- [11] Ramezani, F., Lu, J., Hussain, F.K. (2014). Task-based system load balancing in cloud computing using particle swarm optimization. *Int J Parallel Program*, 42:739–754. <https://doi.org/10.1007/s10766-013-0275-4>
- [12] Pooranian, Z., Shojafar, M., Abawajy, J.H., Abraham, A. (2015) An efficient meta-heuristic algorithm for grid computing. *J Comb Optim* (2015), New York. <https://doi.org/10.1007/s10878-013-9644-6>
- [13] Gąsior, J., Seredyński, F. (2013). Multi-objective parallel machines scheduling for fault-tolerant cloud systems. *Algorithms and Architectures for Parallel Processing*: Springer. pp. 247–256. https://doi.org/10.1007/978-3-319-03859-9_21
- [14] Chen, Z.G., Du, K.J., Zhan, Z.H., Zhang, J. (2015). Deadline constrained cloud computing resources scheduling for cost optimization based on dynamic objective genetic algorithm. *Evolutionary Computation (CEC), 2015 IEEE Congress on*; 2015: IEEE. <https://doi.org/10.1109/CEC.2015.7256960>
- [15] Liu, X.F., Zhan, Z.H., Du, K.J., Chen, W.N. (2014). Energy aware virtual machine placement scheduling in cloud computing based on ant colony optimization approach. *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*; 2014: ACM. <https://doi.org/10.1145/2576768.2598265>
- [16] Li, K., Xu, G., Zhao, G., Dong, Y., Wang, D. (2011). Cloud task scheduling based on load balancing ant colony optimization. 2011. IEEE. pp. 3–9. <https://doi.org/10.1109/ChinaGrid.2011.17>
- [17] The NASA Ames iPSC/860 log by CS Huji labs parallel workload.