International Journal of Computer & Information Technologies (IJOCIT)
www.ijocit.ir & www.ijocit.org
ISSN = 2345-3877

A Symmetric Tasks Scheduling Algorithm Based on PSO in Cloud Computing

Naghme Dashti [*1], Nastaran Avaznia [2]

MSc in Software Engineering, Technical and Vocational Training Organization [1]
MSc in Software Engineering, Azad University of Mashhad [2]
Email: naghme.dashti@gmail.com

Abstract: One of the most important challenges in tasks scheduling problem is appropriate load distribution on computing node in heterogeneous distributed environments such as cloud and grid. Cloud service providers have to pay large costs for providing services. Therefore, they are interested to improve utilizing the processing power of resources according to their processing capacity. In this paper, a PSO based scheduling approach is presented, which improves DPSO algorithm performance for independent tasks scheduling in cloud computing environment. First, the requests are sorted and prioritized, and then, they are assigned to resources by symmetrical load distribution technique. So that, the effective utilization of the resources processing capacity is provided in cloud system. It also reduces the makespan in comparison with the previous algorithm by adding some effective parameters in fitness function of DPSO algorithm, especially, when the requests entered into the system is more.

Keywords: Cloud Computing, Task scheduling, Symmetrical Load Distribution, Particle Swarm Optimization.

Introduction

Cloud computing is a new technology in IT cycle. Cloud can be defined as “a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned, and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers” [1].

It provides the security, speed, proper data storage and network computing services on internet. Services are presented in three levels in cloud computing environments; Infrastructure as

a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS)[2].

Tasks scheduling in IaaS is an important process that its aim is the implementation of the requests on the resources efficiently and with considering the characteristics of the cloud environment. Tasks scheduling considers virtual machines (VMs) as scheduling units to allocate the heterogeneous physical resources to tasks [3]. The task scheduling problem is a known NP-complete problem. Therefore, there is no exact algorithm as the optimal solution in polynomial time for it. Tasks scheduling problem can be classified into two groups according to the type of tasks; Meta-task assignment that is composed of independent tasks without data dependency, and application or workflow assignment that is composed of dependent tasks. Many approaches are presented for solving tasks scheduling problem in computational environments. Modern heuristic and Meta-heuristic approaches are optimization algorithms the efficiency and capability of which are not limited to a specific scope of issues. Some of these algorithms are used as different scheduling methods are heuristics such as Simulated Annealing (SA), Genetic algorithms (GA), Tabu Search [4], Ant Colony Optimization algorithm (ACO) [5] and Particle Swarm Optimization algorithm (PSO) [6]. In this paper, a symmetrical load distribution method based on DPSO algorithm (which is based on PSO algorithm) has been presented for independent tasks scheduling problem in cloud computing system. The rest of paper is organized as follow; Section 2 presents related works. Section 3 describes problem definition. In Section 4 the DPSO algorithm has been introduced and in Section 5 the proposed method has been described (SLDPSO). Section 6 gives the simulation results of proposed method and compares its performance with previous algorithms. And finally conclusions are made in Section 7.

2. Related Works

Finding an optimal solution for task scheduling problem is NP-complete problem in a computing environment. The quality of an allocation algorithm can be measured by various optimization criteria such as minimizing the last task completion time (makespan) or maximizing the use of resources. In recent years, in addition to the linear and heuristic approaches, many Meta-heuristic approaches are presented for efficient tasks scheduling in distributed computing environments which can easily be implemented. Sachin et al. [7] have considered different heuristic algorithms for independent tasks scheduling in cloud computing environment as a heterogeneous computing system.

They have stated most of the researches and methods which are presented for tasks scheduling in most heterogeneous distributed computing system such as grid can also be used directly in cloud computing environment. Ghorbannia and Aryan [8] presented a synthetic heuristic algorithm for independent tasks scheduling in cloud systems. Their approach optimizes makespan and also decreases the probability of task failure rate on running. Omidi and Rahmani [9] investigated independent tasks scheduling on multiple processors with a new algorithm based on PSO. Sadhasivam and Meenakshi [10] have presented a new method of job grouping using parallel PSO in grid, which in

addition to reducing the communication overhead and tasks completion time, improves utilizing the resources.

Visalakshi and Sivanandam [11] have presented a hybrid PSO method for solving tasks assignment problem. Their algorithm has been developed for dynamic heterogeneous tasks scheduling on heterogeneous processors in a distributed environment that will investigate load balancing as a major issue in scheduling problem. Mathiyalagan et al. [12] presented a new PSO algorithm in grid with better convergence rate and performance by improving the iteration parameter. Salman et al. [13] illustrated that the performance of PSO in solving tasks assignment problem for heterogeneous distributed computing systems is faster than GA. Izakian et al. [14] have proposed a version of PSO approach for Meta-task scheduling in heterogeneous computing systems. Their scheduler goal is to minimize makespan. Lian [15] presented a united search particle swarm optimization algorithm for multi-objectives scheduling problem on parallel machines processes. Pandey, Wu and et al. [16] proposed a PSO based scheduling heuristic for tasks assignment to resources in cloud computing environment. Wang et al. [17] presented dynamic scheduling algorithm for tasks scheduling in cloud computing environment. Mousumi et al. [18] introduced a load balancing algorithm for tasks scheduling in cloud computing. Mezmaz et al. [19] presented a genetic algorithm for parallel tasks scheduling problem on heterogeneous computing system such as cloud computing infrastructure. Their algorithm improves energy consumption in addition to decreases makespan. Page et al. [20] presented a multi-heuristic dynamic task allocation algorithm to map tasks to processors in a heterogeneous distributed system. They have combined the genetic algorithm with some common heuristics in their approach to minimize the overall execution time. Kang and He [21] introduced a new PSO algorithm for Meta-tasks allocation in heterogeneous computing systems (HCS), named DPSO that offers better results than other PSO scheduling heuristics.

In this paper, a scheduling strategy will present that prioritizes requests and add effective parameters in the presented fitness function of DPSO algorithm. Then, it will compare with previous DPSO algorithm and also PN genetic algorithm [20] for evaluating its performance in solving tasks scheduling problem.

3. Problem Definition

In this paper, $T = \{T_1, T_2, \dots, T_n\}$ is defined as a set of independent tasks contains *n* tasks which they have no data dependency and entered into the cloud system at once. Also, $R = \{R_1, R_2, \dots, R_m\}$ is considered as a set of computing nodes contains *m* nodes (resources) within the computing environment which are created dynamic on demand. The purpose of the scheduling problem is providing a solution for tasks assignment to computational resources. Different criteria can be used for evaluating the suitability of a solution or scheduling algorithm. The most important of which is minimizing the makespan, the makespan is completion time of the last task in heterogeneous computing systems. Optimal scheduler tries to reduce the makespan as much as possible. The parameters used in equations are defined in Table 1.

Table 1: Used parameters in equations

Symbol	Quantity
n	Number of tasks
M	Number of resources
T_i^{lenght}	Length of task T_i
R_j^{MIP}	Processing capacity of resource R_j
R_j^{pw}	Previous workload of resource R_j before task T_i allocated to it
T_i^{fs}	File size of task T_i
T_i^{os}	Output file size of task T_i
R_j^{bw}	Bandwidth of resource R_j
t_{ij}^{req}	Required time for sending request of task T_i from data-center to computational resource R_j
t_{ij}^{ack}	Required time for sending acknowledge from computational resource R_j to task T_i in data-center
$Delay_{ij}$	Delay between the time of receiving a request at R_j and the time of sending an acknowledge to T_i

Suppose that $C_{total}(M)_j$ is the total computation cost of the requests which are assigned to a computing machine M_j as follows:

$$C_{total}(M)_j = C_{exec}(M)_j + C_{comm}(M)_j \quad (1)$$

$C_{exec}(M)_j$ is the total execution cost of all tasks allocated to a computing node M_j, and $C_{comm}(M)_j$ is the total communication cost between tasks allocated to a computing node M_j.

Suppose that:

$T_{ij}^{ct} = \frac{T_i^{length}}{R_j^{MIPS}} + R_j^{pw}$ as illustrated [14] and $T_{ij}^{trans} = \frac{T_i^{fs}+T_i^{os}}{R_j^{bw}}$ as presented [8] and [22], So:

$$C_{exec}(M)_j = \sum_i T_{ij}^{ct} \quad \forall T_i \in T \ (1 \le i \le n) \quad (2)$$

$$C_{comm}(M)_j = \sum_i T_{ij}^{trans} \quad \forall T_i \in T \ (1 \le i \le n) \quad (3)$$

$$S_j \propto \left(\frac{1}{t_{ij}^{req}+t_{ij}^{ack}+Delay_{ij}}\right) \quad (4)$$

T_{ij}^{ct} is the time required for processing T_i on resource R_j. Also assuming T_{ij}^{trans} is transfer cost for task allocation T_i to resourceR_j, here S_j is considered as effective parameter in transfer cost which is related with composition of t_{ij}^{req}, t_{ij}^{ack} and $Delay_{ij}$. So, T_{ij}^{trans} is defined as $T_{ij}^{trans} = \frac{T_i^{fs}+T_i^{os}}{R_j^{bw} \times S_j}$.

Therefore, makespan is defined given above relation by following equation:

$$Makespan = maximum(C_{total}(M)_j) \quad \forall M_j \in R \ (1 \le j \le m) \quad (5)$$

4. DPSO

PSO is a nature-inspired evolutionary computing algorithm that was introduced by Kennedy and Eberhart [6]. This algorithm is inspired by social behavior of animals, like birds and fish. That is similar to many evolutionary algorithms like Genetic algorithm. But there is no direct re-combination of the population and doesn't have evolutionary operators such as mutation and

crossover. Instead, it relies on the social behavior of particles. PSO approach is used to solve many optimization problems and it's composed of a certain number of particles that are randomly initialized in a search space. Each particle in the population represents an optimal solution, and in every generation, adjusts its own path based on its best position (local best) and the position of the best particle (global best) of entire population. The stochastic nature of particle increases quick convergence to an optimal global minimum as a reasonable solution.

DPSO is an optimization algorithm based on PSO for independent tasks (Meta-tasks) assignment in heterogeneous computing systems that was introduced by Kang and He [21]. Firstly to make particle swarm optimization algorithm more suitable for solving task assignment problems, particles are represented as integer vectors and a new position update method is developed based on discrete domain. Secondly, an effective neighborhood descent algorithm is applied to emphasize exploitation. In addition, migration mechanism is introduced with the hope to escape from possible local optimum and to balance the exploration and exploitation. DPSO are starting to work with a random population of particles similar to PSO algorithm. The performance of each particle is measured according to a predefined fitness function which is proportional to the cost function. Each particle represent a solution for task assignment using an integer vector of n elements in which its ith position indicates the identity of the machine to which task i is assigned, jϵ{1,…, M}. Kang et al. introduced a deterministic variant of variable neighborhood search heuristic in this algorithm in order to exploration new regions of search space that maybe involves local optima. They introduce two type of neighborhood structure for found solution by DPSO and through it improve global beast particle found in entire population. The simple and direct search process of DPSO will make most individuals gradually gather around a better candidate individual. Therefore, the population diversity and its exploration of the search space are rapidly decreased. The gathered individuals are unable to reproduce the newly better individuals, resulting in non-improvement during remaining iterations. So a migration phase proposed by Lin et al. [23] is employed to regenerate a newly diverse population of individuals in order to greatly increase the exploration of the search space and decrease the selection pressure of a small population.

5. Proposed Algorithm

Whereas, the proposed algorithms in heterogeneous computing environments can also be generalized to a cloud environment, and due to lack of load balancing in DPSO algorithm, in this section, an improved DPSO algorithm is presented named SLDPSO for scheduling independent tasks in a cloud computing environment. The proposed approach in addition to reducing the makespan is more efficient in distributing the workload on the resources according to their processing capacities.

Therefore cloud services providers will be able to make maximum use of processing power of resources per time, which is important in reducing costs. Firstly, the tasks are sorted and prioritized in proposed approach, so that; the tasks with higher workload require more processing power to run faster. Second, the initial population is generated based on DPSO algorithm and its random nature and with adding

some effective parameters for requests responding in fitness function of DPSO algorithm and using the symmetrical load distribution technique. Search will continue for finding optimal solution in problem space.

6. Sorting and Grouping Requests and Generate Initial Population with Symmetrical Scheduling Based on DPSO

Since the requests with higher workload require higher processing capabilities, first, the requests entered into system are sorted in descending order, depending on their workload. On the basis of this sorting, requests with high workload have higher priority, as a result, they can be performed when cloud system has higher processing capability. In Meta-heuristics tasks scheduling algorithms, since the initial populations are created randomly in most cases, they do not have appropriate load distribution on resources according to the processing capabilities of their resources.

Since these heuristics are evolutionary, they need to be repeated more times until converging towards the optimal solution for scheduling problem. Moreover, it can also be observed that while some of the resources are overloaded, some others are low loaded. So, the computing resources with higher workload have to do more processing and also tasks must be waiting for a longer time in run queue of overloaded resources. Therefore, the overall processing time of requests will increase. So here we want to improve workload distribution on resources using the symmetrical load distribution technique for tasks scheduling problem in a cloud environment based on PSO algorithm that has fast convergence to the best solution.

Therefore, as shown in Figure 1, after creating a sorted list of *n* requests based on their workload, this list will be divided into two equal groups of requests. Then, using the first group of tasks TG1= $\{ T_0, T_1, \dots, T_{\frac{n}{2}-1} \}$, initial population is composed randomly for PSO algorithm. It consists of *p* particle in n-dimension search space of problem (*n* is number of independent tasks), considering the specific requirements for tasks assignment to resources symmetrical. Each particle in the population in the search space is corresponding to a scheduling solution for assigning the set of tasks T= $\{T_0, T_1, \dots, T_n\}$ to the set of computational resources R= $\{R_0, R_1, \dots, R_m\}$ which provided by cloud suppliers. Based on the PSO algorithm, if the vector X_k^i indicates the location of i^{th} particle in population at iteration *k*, the value of this vector will use to specify the resource R_{kj}^i (that in proposed solution by i^{th} particle is assigned to task T_j), and vector $R_k^i = [R_{k1}, R_{k2}, \dots, R_{kn}]$ will be established.

In order to achieve symmetrical load distribution, the resource which is selected randomly for task T_j in TG1= $\{T_0, T_1, \dots, T_{\frac{n}{2}-1}\}$, is also assigned to task $T_{(n-1)-j}$ in second group of requests TG2= $\{T_{\frac{n}{2}}, T_{\frac{n}{2}+1}, \dots, T_{n-1}\}$. So, given that the tasks were sorted base on their workload in descending order, resources have load balancing in every presented solution by each particle in the population. Then, the presented fitness function of DPSO algorithm changed and the resulting, fitness function was used to obtain the best solution as the optimal response. Algorithm continues with updating different parameters of

PSO based DPSO algorithm until termination conditions have been reached.

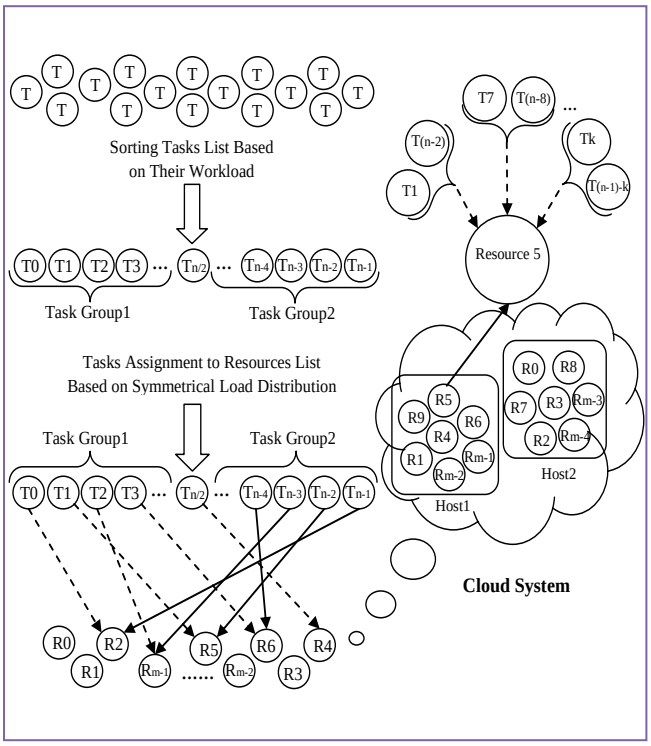

Figure 1: Symmetrical distribution of tasks on resources

T0	T1	T2	…	Tn-3	Tn-2	Tn-1
R2	R5	Rm-1	…	Rm-1	R5	R2

Figure 2: A Sample of scheduling solution as particle in SLDPSO

Finally, the best solution (P_{Gbest}) is selected as the optimal solution of independent tasks scheduling problem in cloud computing environment. Note that the following conditions must always be established during algorithm iterations.

- Any request at any moment can only be allocated to one resource.
- Each resource can perform more than one request (i.e. can be assigned to more than one task) at a time.
- According to what was described about symmetrical load distribution, the resource which is allocated to the task τ_j similarly will be assigned to task $\tau_{(n-1)-j}$.

The values of X and V vectors are updated according predefined equations in DPSO algorithm.

Table 2: Notations

Symbol	Description
N	Number of tasks
M	Number of resources
P	Size of the PSO population
X[i]	Position vector of the i^{th} particle in the PSO population
R[i]	Resource vector for the i^{th} particle (thus, R[*i*][*j*] presents the resource index that is assigned to the j^{th} task in the i^{th} particle)
V[i]	Velocity vector of i^{th} particle
$P_{Gbest}[\,]$	Global best position during the implementation of the algorithm
$P_{Lbest}[i]$	Local best position visited by i^{th} particle

7. Fitness Function

Quality of presented solution by each particle of population is evaluated by fitness function. In this

algorithm, the purpose is minimizing makespan by selecting the appropriate resources which are allocated to tasks. Therefore, the fitness function is selected with respect to the defined equations in section 3, as follows:

$$fittness = Minimize(makespan)$$

SDLPSO
{
Initialize parameters, tasks list T [] and resources list R []
// **Sorting and Grouping Requests and Generate Initial Population**
Scheduler sorts the tasks according to their workload in descending order
Create task groups TG1 and TG2
For each $T_i \in TG_1$ **and** $1 \leq P$
{
Initialize V[*i*][*j*] randomly
Initialize R[i][j] randomly from R[]
// **Symmetrical load Distribution**
Initialize X[*i*][*j*] and X[*i*][(n-1)-*j*] with a copy of R[i][j]
Initialize $P_{Lbest}[i]$ = X[*i*]
}

Evaluate fitness[*i*] and Find P_{Gbest} **so that fitness [**P_{Gbest}**] > fitness[*i*] using (1), (5)** ($\forall i \leq P$)
Repeat
{
Generate velocities V randomly
Update X[i] according to DPSO algorithm
Calculate R[i] for each particle according to obtained X[i]
Evaluate fitness[*i*] using (1), (5)
For each particle $P_{Lbest}[i]$ **= X[*i*] if (fitness[*i*] > fitness**$P_{Lbest}[i]$**)** ($\forall i \leq P$)
Update P_{Gbest} **if fitness [**P_{Gbest}**] < fitness[*i*]** ($\forall i \leq P$)
Apply VND to the global best particle (P_{Gbest}) based on DPSO algorithm
Apply Migration Algorithm of DPSO if necessary.
} until a stopping criterion is satisfied
Return P_{Gbest} as the optimal solution
}

Figure 3: Pseudo code for SLDPSO algorithm

Figure 3 depicts pseudo code of SLDPSO algorithm with considering defined parameters in Table 2. The solution which has better fitness depending on the goal (that is maximizing or minimizing a specific amount) is selected as the best solution in each iteration of algorithm.

8. Simulation and Experimental Results

To evaluate the performance of SLDPSO, the results of performed simulations are compared with the simulation results of DPSO algorithm [21] and genetic algorithm PN [20], in two criteria: decreasing makespan and appropriate load distribution.

Table 3: Execution parameters in experiments

Parameter	Quantity
Task Length	20000 ~ 40000 (Number of Million Instructions)
Task File size and Output Size	400 ~ 800 (MB)
Task Memory Size	1 ~ 100 (MB)
Node Processing Capacity	100 ~ 500 (Million Instructions Per Second)
Node Memory Size	256 ~ 512 (MB
Node Storage Size	500 ~ 10000 (MB)
Bandwidth	100 ~ 500 (Mbps)
Request Time	0.6 ~ 0.8 (Sec)
Acknowledge Time	0.7 ~ 0.9 (Sec)
Delay	0.1 ~ 0.3 (Sec)

SLDPSO, DPSO and PN algorithms are all coded in Java and compiled using NetBeans IDE 6.8 and CloudSim 2.1.1 in Win32 mode.

All tests were performed under Microsoft Windows 7 on a Personal Computer with an Intel Core i5-2600M 2.7 GHz processor and 4 GB of RAM. All algorithms were run in single process model. Comparison of obtained makespans of proposed approach with obtained makespans of DPSO and PN algorithms for several numbers of tasks (cloudlets). Table 4 depicts the results of comparison among SLDPSO, DPSO and PN algorithms with the aim of reducing makespan. Each of the three tested algorithms has been implemented on nine different classes of tasks and single resource class independently.

For each class of tasks, each algorithm ran 10 times. Obtained results in Table 4 include: theaverage makespan as $makespan_{Avg}$, the best makespan as $makespan_{Best}$ and also the percent of deviation for each algorithm for each class in comparison with proposed approach.

Table 4: Comparison of Makespan Amount among SLDPSO, PN and DPSO Algorithms with Several Numbers of Cloudlets

Number of Cloudlet	SLDPSO		PN			DPSO		
	$Makespan_{Avg}$	$Makespan_{Best}$	$Makespan_{Avg}$	$Makespan_{Best}$	*PD%*	$Makespan_{Avg}$	$Makespan_{Best}$	*PD%*
200	464.5	436.2	497.2	497.2	14.0%	489.5	455.5	4.4%
300	672.9	645.2	740.7	690.2	6.9%	718.8	673.7	4.4%
400	872.9	823.9	932.2	899.2	9.1%	927.7	871.3	5.7%
500	1100.8	1045.4	1136.3	1095.2	4.7%	1136.5	1105.2	5.7%
600	1295.0	1255.8	1372.0	1350.7	7.5%	1366.0	1339.6	6.7%
700	1493.0	1464.2	1544.9	1544.9	5.5%	1613.4	1563.9	6.8%
800	1697.8	1652.0	1776.5	1738.0	5.2%	1823.0	1792.3	8.5%
900	1924.4	1851.8	2025.6	2025.6	9.3%	2095.8	2019.1	9.0%
1000	2084.0	2069.8	2223.1	2197.4	6.1%	2268.9	2235.3	8.0%

On average, the improvement of makespan in SLDPSO in comparison with DPSO and PN was 6.59 % and 7.63 % in all classes, respectively.

Figure 4: Simulation Results from SLDPSO, PN and DPSO Algorithms with Several Numbers of Cloudlets.

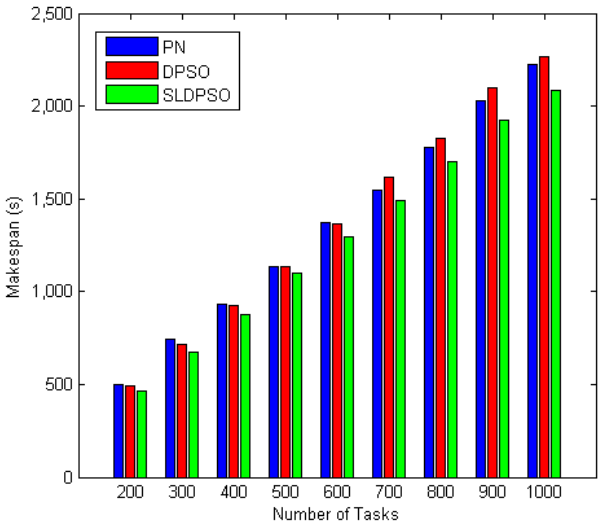

Figure 4: Simulation Results from SLDPSO

Obtained chart from the results is shown in Figure 4. Better makespan from SLDPSO can be seen in the bars when the numbers of requests increase. That means it has great importance in relation to the large number of applications in cloud computing systems and costs of providing services in these systems.

Speedup parameter has been used for evaluating appropriate load distribution on resources. This parameter shows the execution time of requests on single resource to execution time of them on multiple resources in parallel.That is calculated using the following equation [24]:

Speedup= (Serial-length / makespan)

Serial-length is execution time of requests on single resource unit [24]. Table 5 displays obtained Speed Up by SLDPSO, DPSO and PN algorithms for nine different classes of tasks. The computed values of the Speedup consider the obtained makespan for each class in the first experiment.

Table 5: Computing of load distribution on computing nodes in SLDPSO, PN and DPSO algorithms.

Number of Cloudlet	Serial Length	SLDPSO	PN		DPSO	
		Speed Up	*Speed Up*	*Percent Deviation*	*Speed Up*	*Percent Deviation*
200	12322.2	28.24	24.7	14.33 %	27.04	4.43 %
300	18697.9	28.97	27.08	6.97 %	27.77	4.32 %
400	24256.4	29.44	26.97	9.15 %	27.84	5.74 %
500	30855.6	29.51	28.17	4.75 %	27.91	5.73 %
600	37461.4	29.83	27.73	7.57 %	27.96	6.68 %
700	43814.9	29.92	28.36	5.50 %	28.01	6.81 %
800	49556.8	29.99	28.49	5.26 %	27.64	8.50 %

Number of Cloudlet	Serial Length	SLDPSO	PN		DPSO	
		Speed Up	*Speed Up*	*Percent Deviation*	*Speed Up*	*Percent Deviation*
900	55881.4	30.17	27.91	8.09 %	27.67	9.03 %
1000	62441.4	30.16	28.41	6.15 %	27.93	7.98 %

Obtained chart from Table 5 is shown in Fig. 5. Obtained Speedup values of this experiment indicate the SLDPSO has better performance to distribute the requests workload on computing nodes based on their processing capability, on average, in comparison with the DPSO and PN algorithms the improvement is about 6.58% and 7.53%, respectively. Therefore the SLDPSO will increase the efficient use of processing power of resources. It will reduce the load processing costs for cloud service suppliers. Moreover, it will reduce the makespan.

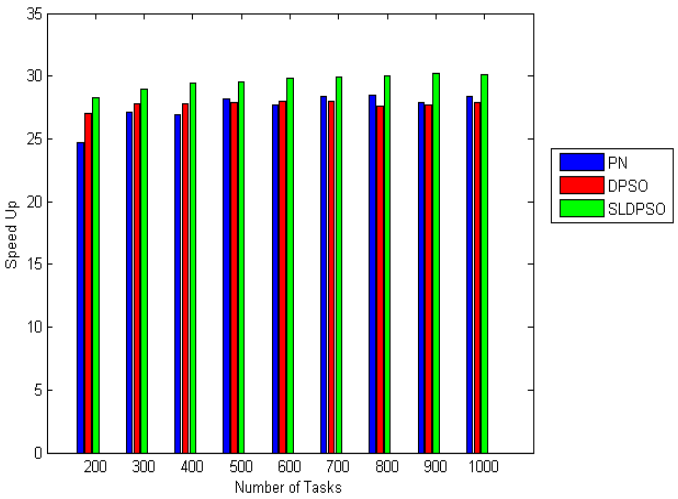

Figure 5: Load distribution on different computing nodes

9. Conclusions

In this paper, a new method based on PSO for tasks scheduling problem in cloud computing environments as a heterogeneous distributed computing system (SLDPSO) by improving DPSO algorithm has been proposed which presents better results in comparison with DPSO and PN scheduling algorithms. This approach creates the initial population of sorted tasks list according to their workload. Then it achieves better results by symmetrical load distribution on resources and also by adding effective parameters in fitness function of DPSO. The results of the simulation and comparisons with similar algorithms show that the proposed approach will minimize the makespan of scheduling algorithm and is more efficient in load balancing and has appropriate load distribution on computational nodes. Finally it will lead to effective use of resources, cost decreasing and improvement of response time to tasks by increasing the convergence rate. In the future, we hope to achieve better results for tasks scheduling problem by combination of partitioning and symmetrical load distribution methods for requests set in cloud system.

References

[1] Buyya R., Yeo C.S., Venugopal S., Broberg J., Brandic I., "Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility". Future Generation Computer Systems, Elsevier Science 25(6): 2009, pp. 599-616.

[2] Keahey, K., Freeman, T. (2008). " Science clouds: early experiences in cloud computing for scientific applications". in proceedings of Cloud Computing and Its Applications, Chicago.

[3] Zhong, H., Tao, K., Zhang, X. (2010). "An approach to optimized resource scheduling algorithm for open-source cloud systems". IEEE The Fifth Annual China Grid Conference.

[4] Braun, T.D., Siegel H.J., Beck N., Hensgan D.A., Freund, R.F., "A comparison of eleven static heuristics for mapping a class of independent tasks on heterogeneous distributed systems". Journal of Parallel and Distributed Computing, 2001, pp: 810-837.

[5] Liu, L., Yang, Y., Lian, L., Wanbin, S. (2006). "Using ant colony optimization for job scheduling in computational grid". Proceedings of IEEE Asia-Pacific Conference on Services Computing (APSCC06).

[6] Kennedy, J., Eberhart, R. (1995). "Particle swarm optimization". In IEEE International Conference on Neural Networks, Vol. 4, (1995), pp: 1942-1948.

[7] Sachin, A., Solanki, V., Minal Gour, B., Mahajan, A.R. (2010). "An overview of different job scheduling heuristics strategies for cloud computing environment". [Online]. Available: http//:www.icett.com.

[8] Ghorbannia Delavar, A., Aryan Y., "A synthetic heuristic algorithm for independent task scheduling in cloud systems". International Journal of Computer Science Issues (IJCSI), Vol. 08, Issue. 06, No. 02, 2011, pp. 289-295.

[9] Omidi, A., Rahmani, A.M. (2009). "Multiprocessor independent task scheduling using a novel heuristic PSO algorithm". 9th IEEE International Conference on Hybrid Intelligent Systems.

[10] Sadhasivam, G.S., Meenakshi, D.k. (2009). "Load balancing, efficient scheduling with parallel job submission in computational grids using parallel particle swarm optimization". IEEE.

[11] Visalakshi, P., Sivanandam, S.N., "Dynamic task scheduling with load balancing using hybrid particle swarm optimization", International Journal Open Problems Compt.Math, Vol. 2, No. 3, September 2009.

[12] Mathiyalagan, P., Dhepthie, R., Sivanandam, S.N., "Grid scheduling using enhanced PSO algorithm". International Journal of Computer Science and Engineering, Vol. 02, No. 02, 2010, pp: 140-145.

[13] Salman, A., Ahmad, I., Al-Madani, S. (2002). "Particle swarm optimization for task assignment problem". Microprocessors and Microsystems, 26(8), Science Direct, November 2002, pp: 363-371.

[14] Izakian, H., Abraham, A., Snášel, V. (2009). "Scheduling meta-tasks in distributed heterogeneous computing systems: a meta-heuristic particle swarm optimization approach". IEEE International Conference on Computer Science and Information Technology.

[15] Lian, Z., "A united search particle swarm optimization algorithm for multi objective scheduling problem". International Journal Applied Mathematical Modelling 34, ELSEVIER, 2010, pp. 3518-3526 .

[16] Pandey, S., Wu, L., Guru, S., Buyya, R. (2010). "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments". 24th IEEE International Conference on Advanced Information Networking and Applications (AINA) 2010, pp: 400-407.

[17] Wang, W., Zeng, G., Tang, D., Yao, J., "Dynamic trusted scheduling for cloud computing". Journal Expert Systems with Applications, ELSEVIER, 2011.

[18] Mousumi, P., Debabrata, S., Goutam, S., "Dynamic job scheduling in cloud computing based on horizontal load balancing". IJCTA, International Journal Computer. Technology, Application, Vol. 2, September 2011, pp: 1552-1556.

[19] Mezmaz, M., Melab, N., Kessaci, K., Lee, Y.C., Talbi, E.G., Zomaya, A.Y., Tuyttens, D., "A parallel bi-objective hybrid meta-heuristic for energy-aware scheduling for cloud computing systems". Journal of Parallel and Distributed Computing, ELSEVIER, 2011.

[20] Page, A.J., Keane, T.M., Naughton, T.J., "Multi-heuristic dynamic task allocation using genetic algorithms in a heterogeneous distributed system". Journal of Parallel and Distributed Computing 70, ELSEVIER, 2010, pp: 758-766 .

[21] Kang, Q., He, H., "A novel discrete particle swarm optimization algorithm for meta-task assignment in heterogeneous computing systems". Journal of Microprocessors and Microsystems 35, ELSEVIER, 2011, pp: 10-17.

[22] Ghorbannia Delavar, A., Rahmany, M., Halaakouie, A., Sookhtesaraei, R. (2010). "An

optimized genetic-based algorithm for scheduling in distributed grid". IEEE 2nd International Conference on Computer Technology and Development (ICCTD), November 2010, pp: 365-369.

[23] Lin, Y.C., Hwang, K.S., Wang, F.S. (2004). "A mixed-coding scheme of evolutionary algorithms to solve mixed integer nonlinear programming problems". Compute. Math. Appl. 47, 2004, pp: 1295-1307.

[24] Wen, Y., Xu, H., Yang, J., "A heuristic-based hybrid genetic-variable neighborhood search algorithm for task scheduling in heterogeneous multiprocessor system". Information Sciences 181, ELSEVIER, 2011, pp: 567–581.

[25] Ghorbannia Delavar, A., Dashti B., "A heuristic approach based on PSO with effective parameters for independent tasks scheduling in cloud computing". Journal of Computing, Vol. 04, Issue. 05, ISSN 2151-9617, May 2012, pp.31-38.

Authors:

Naghme Dashti received her B.Sc. in software engineering from Payame Noor University, Mashhad, IRAN, in 2007, and received her M.Sc in software engineering from Payame Noor University, Tehran, IRAN, in 2013. Her research interests are in distributed and computational systems, grid computing and cloud computing.

Nataran Avaznia received her B.Sc. in IT engineering from Azad University of Mashhad, IRAN, in 2010, and received her M.Sc degree in

software engineering from Azad University, mashhad, IRAN, in 2013. Her research interests are in distributed systems and cloud computing.

