Research Article

Particle Swarm Optimization for Load Balancing in Distributed Computing Systems – A Survey

Vidya S. Handur[1], Santosh L. Deshpande[2], Prakash R. Marakumbi[3]

[1]KLE Technological University, India
[2]Visveswaraya Technological University, India
[3]Tontadarya College of Engineering, India
vidya_handur@kletech.ac.in[1], sld@vtu.ac.in[2], pmarakumbi@gmail.com[3]

Article History: Received: 10 November 2020; Revised: 12 January 2021; Accepted: 27 January 2021; Published online: 05 April 2021

Abstract: Development of technology like Cloud Computing and its widespread usage has given rise to exponential increase in the volume of traffic. With this increase in huge traffic the resources in the network would either be insufficient to handle the traffic or the situation may cause some of the resources to be over utilized or underutilized. This condition leads to reduced performance of the system. To improve the performance of the system the traffic requires to be regulated such that all the resources are utilized conferring to their capacity which is known as load balancing. Load balancing has been one of the concerns in the distributed computing systems where the computing nodes do not have a global view of the network. There have been constant efforts to provide an efficient solution for load balancing through the approaches like game theory, fuzzy logic, heuristics and metaheuristics. Even though various solutions exist for balancing the load, the issue is challenging as there does not exist one best fit solution. The paper aims at the study of how Particle Swarm Optimization approach is used to achieve an optimal solution for load balancing in distributed computing system.

Keywords: Distributed computing system, Load balancing, Particle Swarm Optimization, Swarm Intelligence

1. Introduction

Distributed system is a group of autonomous computing nodes that are geographically apart connected to form a network. The distributed systems can be homogeneous or heterogeneous systems. The nodes communicate and coordinate their actions through message passing to achieve sharing of resources, fault tolerance, openness, scalability and transparency [22]. Following are the characteristics of distributed systems:

Lack of global control unit: In distributed computing system, it is difficult to implement a global control unit due to large scale, dynamic and heterogeneous computing nodes.

Sharing of Resources: To execute a task, the computational nodes share their local resources like disk storage, computational capacity, data and web objects with each other in the system.

Openness and unreliability: The distributed computing systems are open and the structure of the network and nodes are unreliable causing the unauthenticated users to enter into the system. Due to the autonomy of the computing nodes, a few or more nodes may fail to function independently. Therefore fault tolerance and reliability are significant in distributed computing systems.

Heterogeneity: Some of the nodes in the distributed computing systems are heterogeneous with differences in their network components, speed, and storage. For example in social network systems due to differences in speed and bandwidth of the node the responsiveness of the nodes may be different. Therefore there is a need for maximizing the resource utilization.

Fault Tolerance: If there is a failure of any component in the distributed computing system it continues its operation by delegating the task to another node.

The advantages of such systems are high performance, low latency, extensibility and availability at low cost. Some of the applications of distributed systems are Cloud, Grid, WWW, automated banking systems etc. Over the years, there has been a rapid growth in the technology and increase in volume of data generation which is significant factor to the underutilization or overutilization of computing nodes. This aspect of overutilization or underutilization is due to the load imbalance on the computing nodes that deteriorates the overall system performance. To enhance the performance of distributed systems, it is important to keep the system load equal on each node in homogeneous distributed system. However in a number of applications involving heterogeneity, the computing nodes are equipped with different amounts of processing powers, so that a given task may complete more quickly on one node than another. Therefore allocating tasks proportional to the node's capacity in heterogeneous distributed system is essential. The assignment of task to a computing node is known as load

balancing. The existing surveys include most of the proposed approaches like Round Robin, Throttled algorithm, fuzzy logic, game theory, heuristics and metaheuristics for static and dynamic load balancing in centralized and decentralized systems. The proposed study refers predominantly as to how the Particle Swarm Optimization (PSO), a metaheuristic approach can be applied for dynamic load balancing in distributed computing systems.

The paper is organized as following: Section 2 briefs about the load balancing and its significance, Section 3 introduces Swarm Intelligence and Section 4 shows the flow of research focus and section 5 shows how PSO can be applied for balancing load followed by conclusion and references.

2. Load Balancing

Load balancing is a technique to find the mapping of tasks involving computations to the computing nodes which leads to an equal load on each computing node in the system. When a task arrives at a node, the task may be either carried out at the node at which it arrives or relocated to alternative node. The algorithms of load balancing are categorized into two: static and dynamic [14]. In static algorithms the tasks are allocated to the computing nodes at compile time using the task information like its amount of resources needed, arrival time, average execution time. However complete information about the requirements of the tasks may not be obtained in dynamic systems where the traffic pattern varies with time. The decision of transferring a task does not consider the system current state. Periodic reassignment of the tasks may be required as the application requirements vary. Use static algorithms in distributed computing systems may cause some of the nodes to stay heavily loaded whereas a few others may stay idle. Due to these limitations and advancement in the technology there is a huge increase in the traffic which is unpredictable and varying. Hence the static algorithms are not suitable. The other class of load balancing algorithms is the dynamic algorithms that make decision considering the current load on the computing nodes. Each node decides whether a task has to be carried out locally or to be transferred to an alternative node for remote processing. In the case of remote processing, there is a latency incurred due to result of migrating the task over the network. The dynamic load balancing algorithms can be centralized or distributed. In centralized algorithms there is a single central server which distributes the task to the nodes. The drawback of such algorithms is single point of failure of the load balancer. The algorithms can also be fully distributed where there is no global control unit. Such decentralized algorithms should be capable of making decisions autonomously which requires intelligence. To balance the load effectively and dynamically involves many key considerations as mentioned below [3]:

- Estimation of load
- Profitability Determination
- Vector Calculation for Task Transfer
- Selection of Task
- Migration of Task

(i) Estimation of Load: This is the first phase in any load balancing algorithm. There must be some estimation of load on the computing nodes to determine that a load imbalance exists.

(ii) Profitability determination: After determining the loads of the computing nodes, the existence of a load imbalance is to be identified which further can be used to make a decision whether load balancing needs to be initiated. The load balancing is initiated if the cost of load imbalance exceeds the cost of load balancing.

(iii) Vector Calculation for Task Transfer: Depending on the load estimation, the ideal task allocations essential to balance the system are computed.

(iv) Selection of Task: In this phase the tasks are identified for transfer to best satisfy vectors calculated in previous step by considering the task size.

(v) Migration of Task: After selection of the tasks, tasks are reassigned from one computing node to another.

All these key consideration must be dynamically computed in the situations where the system load is invariably changing. Therefore the class of swarm intelligence algorithms is suitable which can consider the current state of the system.

2.1. Performance Metrics

The main objective of balancing the load is to improve the system performance. Therefore the load balancing algorithm must incorporate the metrics that ensure the improvement in the system performance. The major performance metrics of load balancing are [11][12]:

- **Response time:** It is the time elapsed from the submission of a task till the response to the task is made. A good load balancing algorithm should aim at minimizing this metric
- **Throughput:** It is the rate of number of tasks completed per unit of time. An effective load balancing algorithm should aim at maximizing this metric
- **Makespan:** It is a measure of the total time elapsed in processing tasks allocated to a node. The load balancing algorithm should be designed to minimize the makespan.
- **Resource Utilization:** It is a measure of degree of system resources utilized to accomplish the task. To improve the performance through load balancing this metric must be maximized.
- **Migration time:** It is a measure of the time required to migrate the load from one computing node to another. Minimum migration time enhances the system performance.
- **Fault tolerance:** It is measure of the performance of the load balancing algorithm when there is a failure of one or more computing nodes in the system.

3. Swarm Intelligence

The domain of multi agents in the field of distributed systems has developed progressively widespread in the last few decades [1][2]. Various applications, in the fields of robotics, distributed computing and computer networks, are considered by the approaches which are constructed based on the principle derived from multi agents known as Swarms. Swarm Intelligence (SI) is a field of Artificial Intelligence (AI) that is biologically-inspired and based on the social behavior of insects such as birds, wasps, ants, termites, bees. Swarm Intelligent systems are defined as distributed systems, involving simple agents with limited computing capabilities, sensing and communication abilities that are designed to achieve a specified task [2].The basic notion behind such a system is that instead of a single sophisticated agent various tasks can be resourcefully accomplished with simple various agents which are self-governing in nature and are generally more adaptive, robust and scalable than those based on only one system which is highly accomplished. The swarm agents can be either heterogeneous or homogenous with limited capabilities. This limited capability can be collectively utilized to perform a task. Properties of SI are:

• It is an Agent Based Model
• Agents interact locally with each other to exhibit global behavior
• Agents are self-organized
• Agents follow simple rules
• Agents are very adaptive
• Agents are decentralized and artificial or natural

These properties of SI allow the swarm to address complex problems that require the individual agents to work together collectively. The developments of the SI algorithms have attracted the researchers in the recent years for its characteristics. Some of the SI algorithms are Particle Swarm Optimization, Artificial Bee Colony Optimization, Ant Colony Optimization, Cuckoo Search Algorithm, Glowworm Swarm Optimization[4].These different kinds of optimization algorithms have been practically applied for various optimization problems [15], [17]. Figure 1 shows some of the applications of Swarm Intelligence

Figure 1. Applications of Swarm Intelligence

4. Proposed Methodology

Considering the significance of load balancing in distributed systems, there are various approaches to minimize the load imbalance. The algorithms designed are either static or dynamic. Further the dynamic algorithms are centralized or distributed. Various methods namely metaheuritic, heuristics or game theory or fuzzy logic is proposed. With reference to figure 2 the survey in the paper narrows down to metaheuritics approach. Among the many metaheuristic approaches namely Ant Colony Optimization (ACO), Greedy Randomized Adaptive Search Procedure (GRASP), Particle Swarm Optimization (PSO), Artificial Bee Colony Optimization(ABCO),Simulated Annealing(SA),this study aims at presenting PSO and its related work for load balancing in distributed computing environment. The paper focuses on the advances over the PSO, in the form of either modification of PSO, hybridization of PSO with other metaheuristic methods, extensions of PSO or convergence analysis and parameter selection. The performance metrics majorly considered by various researchers are response time, CPU and memory utilization, migration time, makespan and convergence analysis.

5. Study of Particle Swarm Optimization

This section presents the basic principle of PSO, its variants in the research for achieving load balancing according to [5]-[22]

5.1 Particle Swarm Optimization

Kennedy and Eberhart introduced PSO in 1995, an optimization technique built on Swarm Intelligence which simulates the behavior of fish schooling and birds flocking to guide the particles to search for an optimal solutions globally [4][5][13].It has drawn lot of attention in the field of research to solve complex problems.

PSO has several advantages.

- It has few parameters to be set hence simple to implement
- It can be used for concurrent processing.
- It is significant in global search
- It is not affected by the scaling of design variables.

However PSO tends to result in fast and premature convergence. PSO has its application in the field of image processing, control systems, networking, machine learning, power systems and many others [16]

1) Basic Principle of PSO algorithm:

The basic PSO algorithm comprises of generating velocities and positions for each particle, calculating fitness values of each particle and updating position and velocity in each iteration. A particle here denotes a point in the search space. Depending on velocity updates particle changes its position from one move to another. In this algorithm the new search space is influenced by the control parameters namely: Inertia weight (w), Number of population, number of iterations, c_1 the self-confidence factor, and c_2 swarm confidence factor [10]. Algorithm 1 shows the operations of the PSO. Some advances of PSO have been proposed for load balancing to overcome the limitations of low accuracy and early convergence of PSO.

2) Improved convergence PSO with Random Sampling mechanism

This mechanism adopts particle swarm optimization algorithm to improve the convergence. The proposed study uses the strategy of random sampling for control parameters to increase the randomness to update the position and velocity of the particle. To ensure convergence, in each iteration the sampling range for inertia weight is identified after the acceleration factors have been sampled in their particular interval. Also in order to utilize dimension information of particles with better velocity and position, the stochastic correction method is assumed on each dimension for the population prime value. The experimental results have shown that the method suggested improves the rate of convergence with increase in the accuracy of convergence as compared to basic PSO [5]

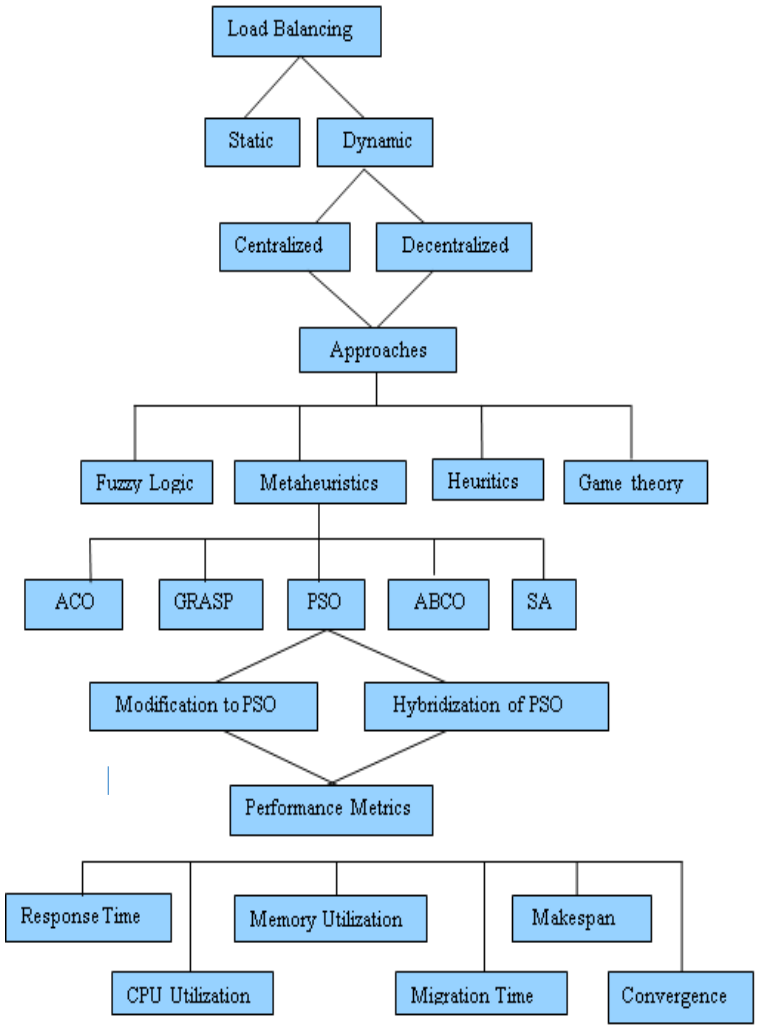

Figure 2. Flow of Study of Load Balancing

Algorithm 1. The pseudo-code of basic PSO

```
FOR each particle i
    FOR each dimension d
      Initialize position x_id randomly within permissible range
      Initialize velocity v_id randomly within permissible range
    End FOR
END FOR
Iteration k=1
DO
   FOR each particle i
       Calculate fitness value
       IF the fitness value is better than p_best_id in history
         Set current fitness value as the p_best_id
       END IF
   END FOR
   Choose the particle having the best fitness value as the g_best_d
   FOR each particle i
       FOR each dimension d
                Calculate velocity according to the equation
                v_id(k+1)= w v_id(k)+c1 rand1(p_id-x_id)+c2 rand2 (p_gd-x_id)
                Update particle position according to the equation
            x_id(k+1)= x_id(k)+v_id(k+1)
       END FOR
   END FOR
k=k+1
WHILE maximum iterations or minimum error criteria are not attained
```

3) Multi-PSO based task scheduling in cloud computing

The proposed method consists of two phases: In the first phase the basic PSO is used with a population of 100 particles. The results of the best mutation of first phase population are performed, and the mutated results are used as the initial population for the next phase. The remaining particles are generated using a deterministic method where in the tasks are allocated to a lightly loaded virtual machine (VM) considering the processing capability and usage of the VM. After allocation of task to VM, the load of VM is updated. In this approach, for each VM the probability of assigning task to is calculated and stored. The task is allocated to the VM with highest probability. The authors have measured the makespan and resource utilization which have been proved to be better than the Round Robin, Min-Min, Genetic Algorithm, basic PSO and FCFS [7].

4) Hybridization of firefly and improved PSO mechanism

The researchers here proposed a system based on the Improved PSO and firefly algorithm. In this approach, high convergence rate but slow down rate close to the optimal point of the search space feature of firefly algorithm and IPSO with high sensitivity to initial conditions have been combined to obtain effective and efficient results. Therefore in order to get good response, the firefly algorithm has been used for initializing best initial population and for task scheduling IPSO has been used. The time complexity of this hybrid method is O (2*(t*n)) where

t: denotes total iterations

n: initial population.

The load balancing metrics measured in this study are the response time, turnaround time, and CPU and memory utilization. This hybridization has been proved to be better than the Firefly and PSO algorithms when not combined [6]

5) αPSO: Task Based Load Balancing mechanism

The authors proposed a method that finds an optimal solution in the cloud environment for migration of tasks from an over loaded virtual machine to a under loaded virtual machine. To generate the particle velocity and its position the basic PSO is used. The values chosen for control parameters are as shown in Table 1.

Table 1. Control parameters

Parameters	Values
ω	0.95
c1	0.8
c2	0.8

The values of positions generated in each iteration are continuous in nature; the authors in this study adopt a small position rule (SPV) to convert the continuous values to discrete values. These values are further used to allocate task to VM. The objective functions considered are minimization of task transfer time and task execution time [18].

6) Random Forest and PSO approach

The proposed approach utilizes particle swarm optimization and Random forest for load balancing across available resources in cloud environment. Initially the random forest algorithm is used to schedule the tasks which are then served for execution into cloud. In order to execute the tasks if the cloud resources are unavailable, then PSO is applied to find the resources. The proposed study of combining PSO with random approach proves to perform better than the improved PSO and Random search algorithms. The performance parameters measured are makespan, flow time and degree of load balancing [19]

7) Modified PSO for load balancing in Cloud computing

In this methodology the authors propose a modified PSO for allocating tasks to the available virtual machines in Cloud. The method works based on partitioning the whole model into different buffers which contain information of tasks like task transfer speed, expected execution time and also information like memory, CPU, bandwidth and MIPS of the available resources. Using this information the PSO is applied to schedule the tasks. The control parameters chosen by the authors are shown in Table 2. The fitness functions are designed to maximize the resource utilization and minimize the makespan [21]

Table 3 summarizes the performance metrics measured by various PSO methodologies that have been proposed.

Table 2. Control parameters

Parameters	**Values**
ω	0.73
c1	2
c2	02

Table 3. Performance Metrics

Proposed by	**Year**	**Performance metrics**					
		Response Time	CPU Utilization	Memory utilization	Makes pan	Conver gence	Migration time
Lijun Sun et.al[5]	2019					√	
Subhadarshini Mohanty et.al[7]	2018	√	√	√			
Mahya Mohammadi Golchi et.al [6]	2019	√	√	√			
Alguliyev, R.M et.al[18]	2019	√					√
Bala, K et.al[19]	2017				√		
Arabinda Pradhan et.al[21]	2020		√	√	√		

6. Conclusion

Load balancing has a significant role with the advent of new technologies and huge increase in the traffic. The severity of the problem increases as the traffic pattern goes unpredictable. Therefore a suitable solution is needed. The paper gives the details of few variants of PSO that can be used in providing an effective solution for load balancing. Although PSO is proposed by many researchers, convergence analysis is the part which is less focused. Therefore there is a need to improve the convergence rate to find an optimal solution for load balancing.

References

1. Dirk Sudhold, “Theory of Swarm Intelligence”, Proceedings of the 13th annual conference companion on Genetic and Evolutionary Computation, July (2011) Pages, 1381–1410.
2. Yaniv Altshuler, Vladimir Yanovsky, Israel A. Wagner, and Alfred M. Bruckstein,”Swarm Intelligence Searchers, Cleaners and Hunters”, Studies in Computational Intelligence (SCI) 26, (2006), 93–132.
3. Jerrell Watts and Stephen Taylor, ”A Practical Approach to Dynamic Load Balancing”, IEEE Transactions On Parallel And Distributed Systems, Vol. 9, No. 3, MARCH (1998).
4. Mohd Nadhir Ab Wahab, Samai Nefti-Meziani, Adham Atyabi “A Comprehensive Review of Swarm Optimizaion Algorithm”, PLOS ONE, May 18, (2015).
5. Lijun Sun, Xiaodong Song and Tianfei Chen, ”An Improved Convergence Particle Swarm Optimization Algorithm with Random Sampling of Control Parameters” Hindawi Journal of Control Science and Engineering,(2019), Article ID 7478498, 11 pages.

6. Mahya Mohammadi Golchi, Shideh Saraeian, Mmehrnoosh Heydari, “A hybrid of firefly and improved particle swarm optimization algorithms for load balancing in cloud environments: Performance evaluation”, Computer Networks, 162 (2019) 106860.
7. Subhadarshini Mohanty, Prashanta Kumar Patra, Mitrabinda Ray. Subasish Mohapatra,”A Novel Meta-Heuristic Approach for Load Balancing in Cloud Computing”, International Journal of Knowledge-Based Organizations, Volume 8, Issue 1, January-March (2018).
8. Divya Chaudhary and Bijendra Kumar, ”A New Balanced Particle Swarm Optimisation for Load Scheduling in Cloud Computing”, Journal of Information & Knowledge Management Vol. 17, No. 1 (2018) 1850009.
9. Arulkumar, V., Bhalaji, N. “Performance analysis of nature inspired load balancing algorithm in cloud environment”, J Ambient Intell Human Comput. (2020).
10. M. Ala’Anzy and M. Othman, "Load Balancing and Server Consolidation in Cloud Computing Environments: A Meta-Study, " IEEE Access, vol. 7, pp. 141868-141887, (2019).
11. Ali M. Alakeel,”A Guide to Dynamic Load Balancing in Distributed Computer Systems”, IJCSNS International Journal of Computer Science and Network Security, Vol.10 No.6, June (2010),pages 153-160.
12. Fatemeh Ebadifard, Seyed Morteza Babamir,”A PSO-based task scheduling algorithm improved using a load- balancing technique for the cloud computing environment”, Concurrency Computat: Practice and Experience,(2017).
13. Aditya, A., Chatterjee, U., Gupta, S.: “A comparative study of different static and dynamic load balancing algorithm in cloud computing with special emphasis on time factor”, Int. J. Curr. Eng. Technol. 3(5), 64–78 (2015).
14. A. Salah Farrag, S. A. Mahmoud and E. S. M. El-Horbaty, "Intelligent cloud algorithms for load balancing problems: A survey," 2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS), Cairo, 2015, pp. 210-216, doi: 10.1109/IntelCIS.2015.7397223.
15. Cholavendhan Selvaraj, Siva Kumar R, Karnan M,” A Survey on Application of Bio-Inspired Algorithms”, International Journal of Computer Science and Information Technologies, Vol. 5 (1) , (2014), 366-370.
16. S. Khanam and M. Hasan, "Nature Inspired Load Balancing Algorithms- A Review," 2019 2nd International Conference on Power Energy, Environment and Intelligent Control (PEEIC), Greater Noida, India, (2019), pp. 94-99.
17. Alguliyev, R.M., Imamverdiyev, Y.N. & Abdullayeva, F.J. “PSO-based Load Balancing Method in Cloud Computing”. Aut. Control Comp. Sci. 53, 45–55 (2019).
18. Bala, K. & Kumar, A. “A Hybrid Approach For Load Balancing: Using Random Forest And PSO Approach (RFPSO)”, International Journal of Advanced Research in Computer Science, (2017), 8(5), 1554-1559.
19. Singhal S., Sharma A.“Load Balancing Algorithm in Cloud Computing Using Mutation Based PSO Algorithm”. In: Singh M., Gupta P., Tyagi V., Flusser J., Ören T., Valentino G. (eds) Advances in Computing and Data Sciences. ICACDS (2020). Communications in Computer and Information Science, vol 1244. Springer, Singapore.
20. Arabinda Pradhan, Sukant Kishoro Bisoy,”A novel load balancing technique for cloud computing platform based on PSO”, Journal of King Saud University - Computer and Information Science, (2020), in press.

Authors

Vidya S. Handur has received under graduation and post-graduation in Computer Science and Engineering. Her areas of interest include distributed systems, cyber security, swarm intelligence.

S.L. Deshpande is doctorate in Computer Science & Engineering. His areas of research include distributed systems, Network Security, Computer Networking, Architecture, Cloud Computing, and Decision Support System.

Prakash R. Marakumbi is working as faculty in the department of Electronics and Communication Engineering at Tontadarya College of Engineering, Gadag, Karnataka