IJCSE International Journal of Computer Sciences and Engineering Open Access
Review Paper Vol.-6, Issue-6, June 2018 E-ISSN: 2347-2693

Modern Approaches To Cloud Scheduling

A. Upadhyay[1], R. Thakur[2], A. Thakur [3]

[1*]Department of Computer Engineering, Institute of Engineering & Technology, DAVV, Indore, India
[2] Department of Computer Applications, International Institute of Professional Studies, DAVV, Indore, India
[3] Department of Computer Science and IT, School of Computer Science, DAVV, Indore, India

[*]*Corresponding Author: upadhyayarvind10@gmail.com, Tel.: +91-9827566736*

Available online at: www.ijcseonline.org

Accepted: 03/Jun/2018, Published: 30/Jun/2018

***Abstract*—** Many important real world problems are computationally "hard" and one of those is cloud scheduling. There are various approaches to cloud scheduling, but in recent time scheduling strategies based on heuristics and metaheuristics are gaining popularity because of their performance. Especially important metaheuristics are nature inspired metaheuristics which have been proved to be very efficient in solving hard problems. These metaheuristics are inspired by natural phenomenon and simulate them in an algorithmic manner. In this paper, we describe those methods and present their successful applications in cloud scheduling problem. We will also describe the formal statement of the problem so that a reader can directly correlate the algorithms with applications below.

***Keywords*—** Nature inspired algorithms; Optimization; Genetic algorithm; Cuckoo search optimization; Particle swarm optimization; Ant colony optimization; Cloud scheduling problem.

I. INTRODUCTION

Hard problems are the bottleneck in modern systems for achieving the desired efficiency. Researchers are trying very hard to solve those problems as efficiently as possible and new solutions in the form of algorithms are coming on the horizon day by day, but the achievement of extremum is still an open issue. By hard we mean NP-Complete (and/or NP-Hard). An NP-Complete problem is one which satisfies the following two properties [1]:

(1) The respective problem is in the class NP, and
(2) Every problem which is already proved to be NP-Complete is polynomial time reducible to the problem at hand.

An NP-Hard problem is one for which it is not possible to prove the property (1) described above for NP-Completeness although property (2) is probable [1]. It is an open question whether NP=P? Now, because of this there is no truly efficient solution for hard problems and it is a fact that most engineering problems are NP-Hard and or NP-Complete. One such hard problem is cloud scheduling. Cloud has emerged as most important distributed computing platforms, although the term cloud computing has no accepted definition in the literature. Cloud is defined as comprising of the following five properties [2]:

(1) Service based
(2) Scalable and elastic
(3) Shared
(4) Metered by users
(5) Uses internet technology

If we add two more properties to the above five properties, this is,

(6) Broad network access
(7) Clouds can monitor and reconfigure resources (based on service usage),

Then we get the NIST definition of cloud Central to the cloud systems is the problem of scheduling in clouds which is NP-Hard [3]. Because of its hardness there is no algorithm that can give an optimal solution in polynomial time. Deterministic algorithms are not feasible as the cost of scheduling by exhaustive search is not acceptable [4]. Modern algorithms are employed to solve this problem and obtain a good solution in reasonable time. By modern algorithms we mean metaheuristics that are capable of producing good solutions fast. NFL (No-Free-Lunch) Theorems answers how efficient an algorithm is compared to some other algorithm and we also presents its main result in this paper, besides metaheuristics which are our main aim and we will see how metaheuristics are applied to the problem of cloud scheduling for good solutions in most cases

This paper is organized as follows; section II formally presents the cloud scheduling problem which is followed by section III in which modern techniques are presented. Section IV is devoted to the applications of those techniques.

Section V presents NFL theorems and section VI is devoted to the conclusion and future work.

II. CLOUD SCHEDULING PROBLEM

In this section we present the formal definition of the cloud scheduling problem, but before we do that we need to understand the formal definition of optimization problems. Optimization is needed everywhere and can be defined as the best use of resources. Formally an optimization problem can be formulated as follows [5]:

$$\text{maximize OR minimize} \quad f(x_1, x_2, \ldots x_n) \qquad \ldots(1)$$

$$\text{Subjected to} \quad h_k(x_1, x_2, \ldots x_n) = 0 \qquad \ldots(2)$$

$$g_j(x_1, x_2, \ldots x_n) \leq 0 \qquad \ldots(3)$$

$$k = 1 \text{ to } L$$
$$j = 1 \text{to } M$$

here f(x) is an objective function or cost function, h(x) and g(x) represents constraints in an L number of equalities and M number of inequalities. One way to classify the optimization problems is in the terms of objectives that is, if there is only one objective function, then the problem is single objective and when there are more than one objective function then the problem is multi objective.

Now the objective of introducing the general structure of optimization problems is the fact that the cloud scheduling problem can be transformed into an optimization problem and then the solution can be searched.

The formal definition of the cloud scheduling problem in terms of an optimization problem is [6]:
Let the directed acyclic graph *G* (*V*, *E*) where *V* denotes the set of tasks and *E* the set of directed edges represents the dependencies between tasks. The scheduling on a cloud can be formulated as follows:

$$\textit{Minimize} \quad f(s) = C_{max}(x) + \sum_{i=1}^{n} \sum_{j=1}^{m} TC_{ij}$$
$$\textit{subject to} \quad C_{max}(x) \leq U(x),$$
$$TC(x) \leq B(x),$$

here *x* represents a solution and *C(x)* represents the makespan *TC(x*) represents the total cost of the solution (can be the computational cost and the data transfer cost) is the cost of processing the i^{th} task in j^{th} machine *U(x*) represents the number of tasks stays more than the deadline and *B(x)* represents the number of the tasks that become over budget.

III. MODERN3 CLOUD SCHEDULING APPORACHES

Traditionally, various approaches are applied to solve the scheduling problem like earliest due date first (EDD) [7], dynamic programming [8], branch-and bound [9] etc and even some of them are guaranteed to find the optimal solution and these techniques are easier to implement than modern techniques. But all of them suffer from the common problem that none of them are able to find the optimal solution in reasonable time [5]

To cope with these disadvantages modern ways to solve such NP-Complete problems are introduced. By modern ways we specifically mean metaheuristics algorithms in this paper. Metaheuristic algorithms can be defined as follows [10]:

"metaheuristics are algorithms which coordinates the interaction between local improvement procedures and higher level strategies so that to escape from local optimum, and an efficient search of the solution space can be performed."

We can divide the optimization algorithms in two classes (1) deterministic algorithms and (2) stochastic algorithms (see figure 1).

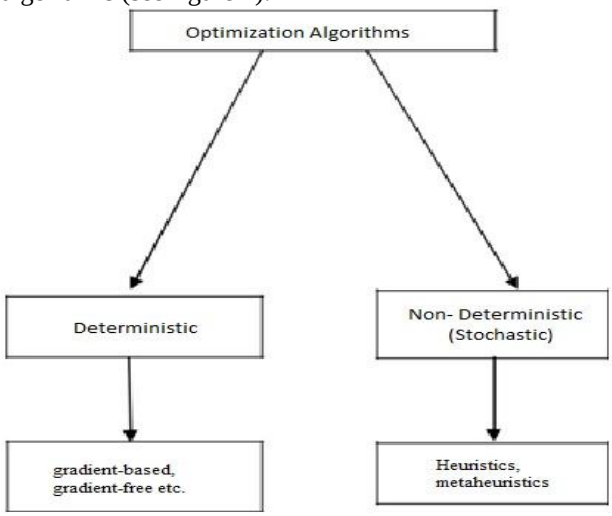

Figure 1: classes of optimization algorithms

Deterministic algorithms are those which repeat its path and value (design variables, etc.) for different runs of the algorithm [5]. A good example of deterministic algorithm is hill-climbing On the other hand stochastic algorithms are those for which the path is not traceable in the subsequent run of the algorithm on the same input although the final values don't have big differences. In essence stochastic algorithms always have some randomness in them.

As already stated deterministic or traditional algorithms have serious disadvantages, so the method of choice are metaheuristics to solve hard optimization problem. Now we

had already shown that how a scheduling problem in cloud can be translated into an optimization problem, it is a fact that the metaheuristics work very well and are the methods of choice for researchers now.

Now we are ready to describe the most successful metaheuristics in detail so that the reader will have an idea of how they work. Application of these metaheuristics in cloud environment is given in section IV A. Genetic Algorithms (GA)

Genetic algorithms are inspired by Darwin's theory of natural selection and are pioneered by John Holland [11]. This simply says that the best, in some sense, will survive and contribute to the better entities. In algorithms the best solution will survive and will be used to produce better solutions.

The basic genetic algorithm is shown in figure 2.

Algorithm I Genetic Algorithm

(1) Define and encode the objective function
(2) Fitness definition
(3) Initialize the population of individuals
(4) While (stopping criteria is false)
(5) Fitness evaluation of all individuals
(6) Create a new population by performing crossover, mutation, etc. and then replace the older population with a new one and again carry out the cycle of evolution with new population.
(7) Decoding of results.

Figure 2: Genetic Algorithm

Following are the genetic operators:

(1) Crossover: it is the swapping of parts of one solution with another solution.
(2) Mutation: the change of some parts of single solution performed randomly.
(3) Elitism: selection of the best solutions.

The main role of crossover is the mixing of solutions and search subspace while that of mutation is to increase the diversity. The role of elitism is to select the best solution by some method. One iteration of the algorithm is a generation of solutions.

GA's are widely used and one of the most popular nature inspired metaheuristics.

B. Cuckoo Search Optimization (CS)

CS is one of the most successful nature inspired algorithm which is developed by Xin-She Yang and Suash Deb in 2009 [11]. CS is based on the phenomenon of obligate brood parasitism of cuckoos and flight behavior of some insects which is modeled by Lévy flights. In obligate brood parasitism cuckoo lay eggs in the nest of some other bird, called the host bird, which is often of other species. The host bird can detect the alien egg and if this happens then the bird either through the alien eggs away or abandon the nest.

The standard CS follows the following three idealized rules:

(1) Each cuckoo can lay only one egg in the host bird's nest.
(2) If the host bird detects the alien egg, then the host can, either throws the egg away or abandon the nest, but the number of nests is constant.
(3) High quality eggs correspond to the next

generation. With these rules the CS is given in figure 3.

Algorithm II Cuckoo Search Optimization

(1) Define objective function
(2) Generate an initial population of host nests
(3) **while** (stopping criteria is false)
(4) Generate a cuckoo randomly
(5) Generate a solution by Lévy flights
(6) Evaluate its solution quality or objective value (fi).
(7) Choose a nest randomly (say k)
(8) **if** (fi < fk)
(9) Replace k by the new solution i
(10) **end**
(11) Abandon a fraction of worst nests
(12) Generate new nests
(13) Keep best solutions
(14) Rank the solution and find the current best
(15) **end while**
(16) Results and visualization

Figure 3: Cuckoo Search Optimization

C. Particle Swarm Optimization (PSO)

Developed by Kennedy and Eberhart in 1995, PSO is a nature inspired algorithm based on the swarm behaviors of various animals such bird flocks [12]. The swarm behaviors exchange information and thus cooperate to achieve a particular aim. These swarm behaviors can be translated to particle swarm algorithms which are proved to be very efficient in solving various important engineering problems. The PSO algorithm is shown in figure 4

Algorithm III Particle Swarm Optimization

(1) Define objective function
(2) Initialize positions and velocity of each particle
(3) Find initial global extremum
(4) **while** (stopping criteria is false)
(5) **For** (all particles and for all dimensions)
(6) Velocity update
(7) Position update
(8) Evaluate the objective function at new positions
(9) Find current best for each particle

(10) End For
(11) Update current global best
(12) Update iteration

(13) **end while**
(14) Results and visualization

Figure 4: Particle Swarm Optimization

A particle in the algorithm refers to a solution and it can be *D* dimensional, which means that there can be a *D* number of variable components in the solution. The word swarm stands for the set of all candidate solution. The algorithm is basically a repeated update of the particles velocities and positions (line 4 – 13) with the aim to find global best solution. The exchange of information in the swarm takes place as the intermixing of Markov chains of velocity and position.

D. Ant Colony Optimization (ACO)

Marco Dorigo and colleagues [13] [14] [15] in the early 1990's Pioneered ACO algorithms. ACO algorithms are nature inspired algorithms which are based on the foraging behavior of real ants.

While foraging, ants move randomly from their nests and as soon as some ants find some foods they take some of it back to their nest after evaluating its quality and quantity. As ants come back to their nests with some food they secret a chemical messenger, called pheromone, which directs other ants to the food source. It is amazing to note that the quantity of pheromone secreted by ants depends on the quality and quantity of the food discovered. This phenomenon makes it possible for ants to find the shortest paths between the food source and the nest.

In ACO ants are heuristics and the whole algorithm is basically the iteration of two actions:
(1) Candidate solutions generation using ants.
(2) Pheromone value modification in the hope that the future candidate solution will be better than the present ones.

ALGORITHM III: ANT COLONY OPTIMIZATION

(1) Define objective function
(2) Define evaporation rate *p* for pheromone
(3) **while** (stopping criteria are not met)
(4) **For (all dimension)**
(5) New solutions generation using Ants
(6) Evaluate new solutions
(7) Pheromone secretion on better routs
(8) Update pheromone value

(9) **End For**

(10) Daemon Actions

(11) **end schedule**

(12) **end while**

(13) Results

Figure 5: Ant Colony Optimization

The ACO algorithm is shown in figure 5.

For now the algorithms presented here are very exciting in their structure and the most important fact in their favor is the simulation of the respective natural phenomena, because natural phenomena are always successful in finding their aim which can be translated to a search algorithm whose aim is to search for extremum of some objective function(s). Now if natural phenomena are successful in every situation, then it must be the case that their translation in the algorithm must also be successful.

It is almost impossible to simulate the natural phenomenon in question into its entirety because it can be very complex. But one must aim for this entire simulation rather than just some of the idealized rules. This is because natural phenomena are adaptive, self-regulating and truly intelligent. These are highly desirable properties in some

algorithm which can only be achieved by making algorithms complex rather than simple. The question is "how best one can do?" This will be answer in section V.

In the next section we present successful applications of the nature inspired algorithms to cloud scheduling problems.

IV. APPLICATIONS OF MODERN APPROACHES

In this section we present the applications of nature inspired algorithms presented in the previous section. The applications are presented in table 1 so that the reader can understand it easily by directly looking at it

V. CLOUD SCHEDULING AND NFL THEOREMS

Various questions that are asked by research community is:

(i) Which is the best algorithm to solve some problem at hand, e.g. cloud scheduling?

(ii) Which is the best algorithm among all the algorithms?

These questions are natural to ask but are not natural to answer. We know that any problem can be translated into an optimization problem and then some optimization algorithm can provide us the solution, like we had translated cloud scheduling problem as an optimization problem (e.q. 4). If we solve (4) efficiently or optimally then only we have solved the scheduling problem of cloud. To solve (4) we had employed various well know metaheuristics to work and results are rerecorded in the table of previous section. Now, one can ask question (i) and (ii) and the main result of NFL theorems answers these as follows:

MAIN RESULT OF NFL-THEOREMS [28] => *"Let Z be a set of all objective functions (also called problems) and let A and B are any two algorithms, whether deterministic or nondeterministic. The number of objective functions on which algorithm A is better than algorithm B is same as the number of objective functions on which algorithm B is better than A with respect to Z."*

The above result implies that there is no best algorithm for *all problems,* and hence there is *no best algorithm.* All the algorithms are same when considered over all the problems.

For cloud scheduling there can different formulations but finite, which are subset of the set Z. but still it is impossible to tell in advance which algorithm is the best on that subset. This is because there is not best algorithm according to NFL-Theorems.

Even if there is no best algorithm on can constantly make progress by incorporating new elements into the algorithm to make it best for some specific set of problems.

Our aim must be to find novel ways presents in some phenomenon that does not correspond to already proposed technique so that the novelty will enhance the performance of the algorithm and accelerate the research. On the contrary researchers are proposing new algorithm for the purpose of publication only in which just the name changes but the basic way of computation does not change. This is clearly a waste of time and efforts. This attitude is distracting the research direction which must not happen. Some mew algorithm must be proposed if and only if it adds novelty to the already proposed algorithms in a way that enhances the performance and accelerates the research.

Table 1 Summary of Literature Survey

S.No	Reference	Scheduling algorithm	Optimization Criteria	Tool	Remarks
1	J. Gu et al.[16]	VM load balancing based on genetic algorithm	load balancing and proper Resource utilization.	OpenNebula	Proposed method can better realize load balancing and proper resource utilization
2	c. Zhao et al.[17]	GA based	Time utilization and resource utilization	Cloudsim	The optimal fitness value at the 100th generation reached 2.9.The utilization of resources is highly developed.
3	k. Zhu et al.[18]	Multi-agent genetic algorithm (MAGA Hybrid Genetic algorithm)	CPU utilization and memory load balancing	Not mention	CPU utilization and memory load balancing for MAGA is much better than Min_min Scheduling
4	L.Zuo et al.[19]	multi-objective optimization algo based on ACO	Makespan, costs, deadline violation rate,resource utilization.	Cloudsim	Proposed algorithm is compared with FCFS,Min-Min,ACO and results proved effectiveness of algorithm
5	Xin Lu, Zilong Gu[20]	load-adaptive cloud resource scheduling model based on ACO	CPU usage	Cluster of four PCs	Quickly find the nearest idle node by the ACO which helps in load balancing.
6	H. Liu et al.[21]	use of ACO to optimize service flow scheduling	Reliability, Response Time, Cost, Security	PC	Reliability of service flow is optimized gradually with the increase of iteration time. optimum global solution can be gained and the convergence speed is fast
7	K. Li et al.[22]	Load Balancing (LBACO) algorithm	Makespan	Cloudsim	LBACO outperforms FCFS and ACO algorithms
8	X. Zuo et al.[23]	self-adaptive learning particle swarm optimization (SLPSO)	CPU and memory utilization rate	MATLAB	This approach is able to obtain a high quality scheduling solution by adaptively selecting velocity updating strategies to update each particle.
9	S. Pandey et al.[24]	particle swarm optimization (PSO) based heuristic to schedule workflow applications	Cost	JSwarm	Proposed algorithm achieved at least three times cost savings as compared to Best Resource Selection (BRS) Based mapping For application workflow. PSO balances the load on compute resources by distributing tasks to available resources
10	A.I.Awad et al.[25]	Load Balancing Mutation (balancing) a particle swarm optimization (LBMPSO)	Average execution time, average cost, average round trip time and average makespan	Cloudsim	LBMPSO compared with standard PSO, random algorithm and Longest Cloudlet to Fastest Processor (LCFP) algorithm to show that LBMPSO can save in, makespan execution time, round trip time, transmission cost
11	Z.Wang et al.[26]	Particle Swarm Optimized based energy aware and	Energy	Cloudsim	proposed algorithm can save energy consumption reducing by 67.5% and increase revenue

		Revenue Enhancing scheduling			enhancing by 8.14 times averagely based on the consideration of communication and QoS
12	R. Raju et al.[27]	Hybrid algorithm which combine ACO and Cuckoo search	Makespan	lab of 10 clients,layer created by Xen Cloud Platform (XCP)	Makespan gets reduced by using a hybrid algorithm.

VI. CONCLUSION AND FUTURE WORK

This work introduces the applications and comparisons of nature inspire algorithms in the field of cloud scheduling. It is clear that this area is very interesting and fruitful for research and society alike.

We also noted in this paper that the cloud scheduling problem is NP-Complete so there is no polynomial time algorithm to solve it, this fact inspires researchers to develop more and more efficient algorithms to solve this problem and this is also one of our aims in this paper, i.e. to inspire researchers. More efficient algorithms will definitely prove fruitful in solving cloud scheduling problem.

REFERENCES

[1] Hopcroft, J.E., Motwani, R. and Ullman, J.D., 2001. Introduction to automata theory, languages, and computation. *ACM SIGACT News*, *32*(1), pp.60-65.

[2] Benatallah, B. ed., 2011. *Cloud computing: methodology, systems, and applications*. CRC Press.

[3] Garey, M.R. and Johnson, D.S., 1979. A Guide to the Theory of NP-Completeness. *WH Freemann, New York*, *70*.

[4] Yu, J., Buyya, R. and Ramamohanarao, K., 2008. Workflow scheduling algorithms for grid computing. In *Metaheuristics for scheduling in distributed computing environments* (pp. 173-214). Springer Berlin Heidelberg.

[5] Yang, X.S., 2014. *Nature-inspired optimization algorithms*. Elsevier.

[6] Rahman, M., Li, X. and Palit, H., 2011, May. Hybrid heuristic for scheduling data analytics workflow applications in hybrid cloud environment. In *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on* (pp. 966-974). IEEE.

[7] Elsayed, K.M. and Khattab, A.K., 2006. Channel-aware earliest deadline due fair scheduling for wireless multimedia networks. *Wireless Personal Communications*, *38*(2), pp.233-252.

[8] Schuetz, H.J. and Kolisch, R., 2012. Approximate dynamic programming for capacity allocation in the service industry. *European Journal of Operational Research*, *218*(1), pp.239-250.

[9] Shi, D. and Chen, T., 2013. Optimal periodic scheduling of sensor networks: A branch and bound approach. *Systems & Control Letters*, *62*(9), pp.732-738.

[10] Gendreau, M. and Potvin, J.Y., 2010. *Handbook of metaheuristics* (Vol. 2). New York: Springer.

[11] Holland, J.H., 1992. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.

[12] Kennedy, J. and Eberhart, R.C., 1995. Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, Piscataway, NJ, USA; 1995. p. 1942–48.

[13] Dorigo, M., 1992. Optimization, learning and natural algorithms. *Ph.*
D. Thesis, Politecnico di Milano, Italy.

[14] Dorigo, M. and Maniezzo, V., *Colorni (1991) A positive feedback as a search strategy*. Technical report 91-016, Politecnico di Milano,
Italy.

[15] Dorigo, M., Maniezzo, V. and Colorni, A., 1996. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *26*(1), pp.29-41.

[16] Gu, J., Hu, J., Zhao, T. and Sun, G., 2012. A new resource scheduling strategy based on genetic algorithm in cloud computing environment. *Journal of Computers*, *7*(1), pp.42-52.

[17] Zhao, C., Zhang, S., Liu, Q., Xie, J. and Hu, J., 2009, September. Independent tasks scheduling based on genetic algorithm in cloud computing. In *Wireless Communications, Networking and Mobile Computing, 2009. WiCom'09. 5th International Conference on* (pp. 1-4). IEEE.

[18] Zhu, K., Song, H., Liu, L., Gao, J. and Cheng, G., 2011, December. Hybrid genetic algorithm for cloud computing applications. In *Services Computing Conference (APSCC), 2011 IEEE Asia-Pacific* (pp. 182-187). IEEE.

[19] Zuo, L., Shu, L., Dong, S., Zhu, C. and Hara, T., 2015. A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing. *IEEE Access*, *3*, pp.2687-2699.

[20] X. Lu and Z. Gu, "A load-adapative cloud resource scheduling model based on ant colony algorithm," IEEE International Conference on Cloud Computing and Intelligence Systems, pp. 296–300, 2011.

[21] Liu, H., Xu, D. and Miao, H.K., 2011, December. Ant colony optimization based service flow scheduling with various QoS requirements in cloud computing. In *Software and Network Engineering (SSNE), 2011 First ACIS International Symposium on* (pp. 53-58). IEEE.

[22] Li, K., Xu, G., Zhao, G., Dong, Y. and Wang, D., 2011, August. Cloud task scheduling based on load balancing ant colony optimization. In *Chinagrid Conference (ChinaGrid), 2011 Sixth Annual* (pp. 3-9). IEEE.

[23] Zuo, X., Zhang, G. and Tan, W., 2014. Self-adaptive learning PSO-based deadline constrained task scheduling for hybrid IaaS cloud. *IEEE Transactions on Automation Science and Engineering*, *11*(2), pp.564-573.

[24] Pandey, S., Wu, L., Guru, S.M. and Buyya, R., 2010, April. A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In *Advanced information networking and applications (AINA), 2010 24th IEEE international conference on* (pp. 400-407). IEEE.

[25] Awad, A.I., El-Hefnawy, N.A. and Abdel_kader, H.M., 2015. Enhanced particle swarm optimization for task scheduling in cloud computing environments. *Procedia Computer Science*, *65*, pp.920-929.

[26] Wang, Z., Shuang, K., Yang, L. and Yang, F., 2012. Energy-aware and revenue-enhancing Combinatorial Scheduling in Virtualized of Cloud Datacenter. *Journal of Convergence Information Technology*, *7*(1), pp.62-70.

[27] Raju, R., Babukarthik, R.G., Chandramohan, D., Dhavachelvan, P. and Vengattaraman, T., 2013, February. Minimizing the makespan using Hybrid algorithm for cloud computing. In *Advance Computing Conference (IACC), 2013 IEEE 3rd International* (pp. 957-962). IEEE.

[28] Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), 67-82.

Authors Profile:

Mr Arvind Upadhyay pursed Bachelor of Engineering from Rajiv Gandhi Prodyogiki Vishwavidyalay Bhopal in 2004 and Master of Engineering in 2010. He is currently pursuing Ph.D. from Devi Ahilya Vishwavidyalaya Indore and currently working as Associate Professor in Department of Computer Science & Engineering, Institute of Engineering & Science, IPS Academy Indore. He is a member of computer society of India since 2013. He has published more than 15 research papers in international journals & conferences. His main research work focuses on Cloud Scheduling & Networking. He has 14 years of teaching experience.

Dr. Ramesh Kumar Thakur was born in Samistipur, Bihar, India, in 1974. He received the B.E. degree in Computer Science and Engineering, the M.E. degree in computer engineering, and Ph.D. in computer engineering, from Devi Ahilya University, Indore, and Indore. In 1998, he was appointed as Asst. Prof at SVITS, Indore India. In 2001, he joined as a Asst. Prof in Department of computer Engineering at Devi Ahilya University, Indore, India, since 2007 he is working as Associate Prof at IIPS Devi Ahilya University, Indore He is involved in coordinating graduate-level and postgraduate-level training program in computer science for the university. During 2011 he was visiting professor at Indian Institute of Technology, Indore, India. He has published many research papers in various national and international journals & participated in so many conferences. His area of research is Information Extraction.

Dr. Archana Thakur was born in Indore, Madhya Pradesh, India, in 1975. She received the Bachelor of Science in computer science, Master of Science in computer science, M.Tech. degree in computer science, and Ph.D. in computer science, from Devi Ahilya University, Indore in 2010. She was appointed as Asst. Prof at School of Computer Science & IT, DAVV since 2002. She is working as Assistant Professor at School of Computer Science & IT, Devi Ahilya University, Indore She is involved in coordinating graduate-level and postgraduate-level training program in computer science for the university. She has published many research papers in various national and international journals & participated in many conferences. Her areas of research are Artificial Intelligence, Machine learning, Data Mining and Soft Computing.

