A Genetic Algorithmic approach for Energy Efficient Task Consolidation in Cloud Computing

Dilip Kumar
Department of Computer Science and Engineering
National institue of Technology Jamshedpur
Jharkhand, India

Bibhudatta Sahoo
Department of Computer Science and Engineering
National institue of Technology Rourkela
Odisha, India

Bhaskar Mondal
Department of Computer Science and Engineering
National institue of Technology Jamshedpur
Jharkhand, India

Tarni Mandal
Department of Mathematics
National institue of Technology Jamshedpur
Jharkhand, India

ABSTRACT

In cloud, processing loads arrive from many users at random time instants in the form of task. A proper resource allocation policy attempts to assign this task to available VMs on different host so to complete the execution of the tasks in the shortest possible time with minimum power consumption. The complexity of the resource allocation problem with cloud increases with the number of hosts and becomes difficult to solve effectively. The resource allocation problem is a combinatorial problem and known to be NP-complete. The exponential solution space of the load balancing problem can be searched using heuristic techniques based on Genetic algorithms to obtain a sub - optimal solution in acceptable time. The novel genetic algorithm framework has been proposed for task scheduling to minimize the energy consumption in cloud computing infrastructure. The performance of the proposed GA resource allocation strategy has been compared Random and Round Robin scheduling using in house simulator. The experimental results show that the GA based scheduling model outperforms the existing Random and Round Robin scheduling models.

General Terms:

Cloud Computing, Energy Efficient

Keywords:

Cloud Computing, Energy Efficient, Genetic algorithms, Optimization, Heuristic, NP-complete.

1. INTRODUCTION

Cloud computing infrastructures are designed to support the accessibility and deployment of various service oriented applications by the users[8][15]. Cloud computing services are made available through the server firms or data centers. To meet the growing demand for computations and large volume of data, the data centers hosts high performance servers and large high speed mass storage devices [2]. These resources are the major source of the power consumption in data center along with air conditioning and cooling equipment [19]. More over the energy consumption in cloud are proportional to the resource utilization and data centers are almost the worlds highest consumers of electricity [4]. Due to the high energy consumption by data centers, it requires efficient technology to design green data center [13]. Cloud data center, on the other hand, can reduce the energy consumed through server consolidation, whereby different workloads can share the same server using actualization and unused servers can be switched off.

Generally, clouds are deployed to customers giving them three levels of access: Software-as-a-Service (SaaS),Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS). Clouds use virtualization technology in distributed data centers to allocate resources to customers as they need them. The task originated by the customer can differ greatly from customer to customer. Entities in the Cloud are autonomous and self-interested; however, they are willing to share their resources and services to achieve their individual and collective goals. In such open environment, the scheduling decision is a challenge given the decentralized nature of the environment. Each entity has specific requirements and objectives that need to achieve.

We capture the Cloud scheduling model based on the complete requirement of the environment. We further create a mapping between the Cloud resources and the combinatorial allocation problem and propose an adequate economic-based optimization model based on the characteristic and the structure of the Cloud.

In particular executing an application on required resource can be made available through two step: creating instance of virtual machine as required by the application(*VM provisioning*) and scheduling the request to the physical resources other wise known as *resource provisioning* [19].

In cloud, processing loads arrive from many users at random time instants in the form of task. A proper resource allocation policy attempts to assign this task to available VMs on different host so to complete the execution of the tasks in the shortest possible time with minimum power consumption. The complexity of the resource allocation problem with cloud increases with the number of hosts and becomes difficult to solve effectively. The resource

allocation problem is a combinatorial problem and known to be NP-complete. The exponential solution space of the load balancing problem can be searched using heuristic techniques based on Genetic algorithms to obtain sub-optimal solution in acceptable time [24, 3]. The genetic algorithm is an evolutionary algorithm, that have been proven to be a successful in generating sub-optimal solutions to many scheduling problems A genetic algorithm performs a multi-directional search by maintaining a population of potential solutions and an objective (evaluation) function which plays the role of an environmental [7, 16]. Task consolidation problem addressed in this paper is to assign n task to a set of r resources in cloud computing environment. This energy efficient load management maintains the utilization of all compute nodes and distributes virtual machines in a way that is power efficient. The goal of this algorithm is to maintain availability to compute nodes while reducing the total power consumed by the cloud.

In this paper, we propose the GA-based task scheduling model and introduces a suitable codification scheme for chromosome. We also explain how making an optimal task schedule and compose elements of the GA scheduling function. To generate a new population, we have generated a POP_SIZE number of random initial population and calculating the fitness value of individuals. Then, using roulette wheel selection method, parents are selected to produce offsprings using single point crossover with probability 0.8. Some of the individuals are subjected to the mutation with a probability 0.2. The population for the next generation are selected again through roulette wheel selection method. The constant population size has been maintained for a fixed number of iterations. The individual from the last generation with minimum energy value is selected to allocate the tasks to VMs.

The remainder of this paper is organized as follows. The next section discusses related research outcomes on energy aware scheduling and resource allocation for cloud computing systems. In *Section 3* we define the model of cloud computing system, task model and energy consumption of the system. Based on this system model, we have defined the problem to minimization the energy in cloud computing environment. *Section 4* discusses the Genetic algorithm used in this study with the illustration. *Section 5* discusses our simulation set up and analyses our simulation results. Finally, conclusions and directions for future research are discussed in *Section 6*.

2. RELATED WORK

galloway et al. 2011 [6] has proposed a load balancing techniques for infrastructure as a service (IaaS) for cloud computing. There are many proposed resource utilizing market-based resource management for various computing areas yeo, buyya and kusic have modeled the problem of consolidation[23, 4, 11]. The complexity of the model is too high to the optimization of controller even for a small number of nodes, that is not suitable for large-scale real-world problem. Srikant et al. 2008 [21] have studied the multitiered web-applications problem in virtualized heterogeneous systems in order to minimize energy consumption. To optimization energy consumption, the authors have proposed a heuristic for the multidimensional bin packing problem as an algorithm for workload consolidation. Song et al. 2009 [20] have proposed priorities based resource allocation to applications in a multi-application virtualized cluster. The methods requires machine-learning to obtain the optimized results. Verma et al. 2008 [22] have modeled the problem of dynamic placement of services in virtually HDC as continuous optimization. The authors have proposed a heuristic approches for the problem. they have used a bin packing problem with variable bin sizes and costs. Calheiros et al. 2009 [5] have studied the problem of mapping VMs on PH for optimizing network communication between VMs, however, the problem has not been to optimize the energy consumption.

Genetic algorithms can be successfully applied to solve job shop scheduling problem [14], and it can also apply in heterogeneous System [16], grid computing [17] and cloud computing [18]. Most of these researches assume that each task has a fixed amount of execution time (in homogeneous system). braun at el. 2001 [3] compare eleven heuristic and meta-heuristic scheduling methods including of a simple GA-based scheduler, Min-Min, Min-Max, Minimum Completion Time algorithms. The experimental study was performed for task scheduler for independent task in distributed heterogeneous computing environment. The task execution time instances have defined using the ETC matrix model proposed by ali at el. 2000 [1]. Zomaya at el. 2001 [24] proposed a dynamic load balancing framework on genetic algorithm that uses a central scheduler approach to handle all load balancing decisions. The effectiveness of central server with load-balancing has been demonstrated for homogeneous distributed computing system. kang at el. 2010 [9] have discussed in maximizing reliability of distributed computing systems with genetic algorithm based task allocation and the task have represented in task graph. This comparison of different heuristic through simulations proves the effectiveness of genetic algorithms on HDCS. Several researchers have used GA for load balancing on cloud computing systems; however the majority of the papers has no specific representation of the genetic algorithm.

3. SYSTEM MODEL

The cloud computing system is consists of fully interconnected set of m resources denoted as R These computing resources are the physical machine in cloud data center and refered as host computing system or host in this chapter. These resources are to be allocated on demand to run applications time to time. Figure 1 depicts the system model of cloud computing system, that has been referred in this Chapter. We have assumed the centralized cloud is hosted on a data center that is composed of large number of heterogeneous servers. Each of server may be assigned to perform different or similar functions.

The virtualization technologies allow the creation of multiple virtual machine on any of the available physical host. There for a task can be flexibly assigned to any server. Servers can be modeled as a system that consumes energy in idle state to perform maintenance functions and to have all the subsystems ready while it waits for task to arrive. On arrival of task , a VM processes the task and host may spend an additional amount of energy, which depends on the number of resources demanded by the task, it is represented as resource utilization in work load model[10].

Although a cloud can span across multiple geographical locations (i.e., distributed), the cloud model in our study is assumed to be confined to a particular physical location.We assume that resources are homogeneous in terms of their computing capability and capacity; this can be justified by using virtualization technologies [12]. It is also assumed that a message can be transmitted from one resource to another while a task is being executed on the recipient resource, which is possible in many systems[12]. The maximum and minimum energy consumption of the server in cloud computing system are denoted as *pick load state* and *idle state*.

Total Energy E consumed by CPU utilization in time τ by the cloud computing infrastructure by an efficient allocation of resources to the set of tasks. The resource allocation problem on cloud computing are based on following assumptions.

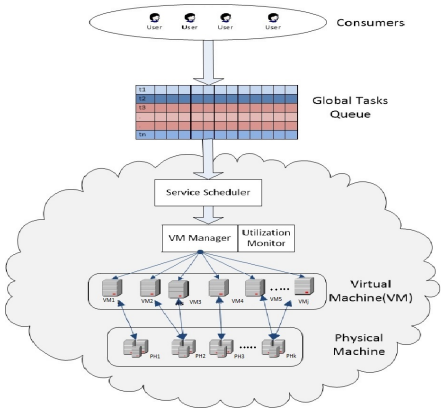

Fig. 1. Cloud Computing Architecture

—Virtualization technologies allow the creation of multiple virtual machines on any of the available host.

—Each host may be assigned to perform different or similar services.

—Hosts consumes energy in an idle state to perform maintenance functions and denoted as P_{min}.

—Hosts consumes more energy as per utilization of the CPU by the tasks.

—Hosts consumes maximum energy at the pick level and denoted as P_{max}.

—Hosts put the task in waiting queue, if its CPU utilization is at pick level.

The work load submitted to the cloud is assumed to be in the form of tasks. These tasks are submitted service scheduler. The service schedular allocates the tasks to VMs on different computing hosts. We have assumed the task as the computational unit to execute on the allocated VM. The task model refered in this chapter are with following assumption.

—A task represents a users computing or service request.

—A task is an independent scheduling entity and its execution cannot be preempted.

—The tasks can be executed on any node.

—Arriving task t_j is associated with a *task ID, arrival time, CPU utilization, and expected time to compute* as shown in figure 2 for example.

—Tasks arrival rate is Poisson.

—Resource utilization by task is normal distribution between 10% and 100%.

—The resource allocated to a particular task must sufficiently provide the resource usage for that task. If resources are not sufficient, providing the resource usage for a particular task, then task putted in waiting queue.

As shown in figure 2 one row of the task arrival list contains the task id, task arrival time, resource utilization by task and estimated execution times for a given task on each machine.
The ETC(t_j,1) indicates the task id, ETC(t_j,2) indicates the task arrival time which is poisson, ETC(t_j,3) indicates the resource utilization by the task t_j and ETC(t_j,4) indiactes the estimated execution times on VM1, and so on.

Task ID	Task Arrival time	Resource utilization(%)	Task Execution Time on VM									
			M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
1	1	54	12	12	12	12	12	12	12	12	12	12
2	1	62	5	5	5	5	5	5	5	5	5	5
3	1	31	7	7	7	7	7	7	7	7	7	7
4	1	51	12	12	12	12	12	12	12	12	12	12
5	1	97	9	9	9	9	9	9	9	9	9	9
6	2	59	8	8	8	8	8	8	8	8	8	8
7	2	57	11	11	11	11	11	11	11	11	11	11
8	2	31	8	8	8	8	8	8	8	8	8	8
9	2	54	10	10	10	10	10	10	10	10	10	10
10	2	66	10	10	10	10	10	10	10	10	10	10
11	2	71	17	17	17	17	17	17	17	17	17	17
12	3	45	17	17	17	17	17	17	17	17	17	17
13	3	43	13	13	13	13	13	13	13	13	13	13
14	3	99	9	9	9	9	9	9	9	9	9	9
15	3	13	7	7	7	7	7	7	7	7	7	7
16	3	90	12	12	12	12	12	12	12	12	12	12
17	4	93	12	12	12	12	12	12	12	12	12	12
18	4	82	11	11	11	11	11	11	11	11	11	11
19	4	18	6	6	6	6	6	6	6	6	6	6
20	4	33	14	14	14	14	14	14	14	14	14	14

Fig. 2. Example of arrival tasks list

Energy efficient resource allocation for cloud computing can be reprsented as Linear programming problem to minimize the total enegy consumed E, and represented as equation 1

$$Minimize\ E\ =\ \sum_{\tau=1}^{\tau}\sum_{i=1}^{m} E_i(\tau) \tag{1}$$

Subjected to:

$$E_i(\tau) = (P_{max} - P_{min}) * \frac{U_i(\tau)}{100} + P_{min} \tag{2}$$

$$U_i(\tau)\ =\ \sum_{j=1}^{n} u_{(i,j)}\ \leq\ peakload\,at\,time\,\tau,\ \forall\, R_i\,\epsilon\,R\,and\,\forall\, t_j\,\varepsilon\,T \tag{3}$$

$$u_{(i,j)}\ =\ 0;\ when\,the\,task\,j\,is\,not\,assigned\,to\,node\,R_i. \tag{4}$$

$$u_{(i,j)}\ =\ u_{ij};\ when\,the\,task\,j\,is\,assigned\,to\,node\,R_i. \tag{5}$$

The above equation 1 show that the minimization of energy is subjected to the utilization of resources by the task for the time τ.

4. A GENETIC ALGORITHM

The algorithm 1 described in this section is straightforward with two parts: initialization and looping. After initialization, it generates the feasible solution randomly, and then find the fitness value for best solutions. In looping parts, it checks whether the termination condition is met. If looping continues, selection, crossover (algorithm 4) and mutation (algorithm 5) operators are applied in a sequence. Then the better solution is saved during this iteration. At the end of the program, the saved best solution will be output as the optimized result.

4.1 Encoding

A chromosome in this GA consists of $|C|$ genes, each represents the allocated resource ID (VM ID) to the task. The value of a gene is a positive integer between 1 and VM_MAX, representing the virtual machine where the task is allocated. Figure 3 shows an example of

Algorithm 1 workflow of a Genetic Algorithm

1: Find the fitness value of each chromosome in the population.
2: Reproduce a new population by repeating the following steps.
3: Select two individual chromosomes from a population according to selection method, roulette wheel selection.
4: Cross over the selected parents if crossover probability met, to produce a new child. Otherwise, the children are an exact copy of parents.
5: Mutate each new offspring (child) if a mutation probability met, at each locus (position in the chromosome).
6: Place new child in a reproduced population.
7: Store Best individual solution.
8: If the Maximum number of generation reached, stop, and return the best solution, else Go to step 2.

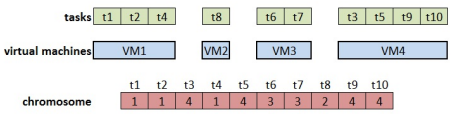

Fig. 3. Individual Encoding(chromosome)

task scheduling and its corresponding chromosome. In this example, task t1, t2, t3, t4, t5, t6, t7, t8, t9 and t10 is placed on VM1, VM1, VM4, VM1, VM4, VM3, VM3, VM2, VM4, and VM4 respectively.

4.2 Fitness Function

This Fitness function finds the makespan of giving task execution pattern.

Algorithm 2 Fitness Function Algorithm

Input: Task sequence and ETC Matrix
Output: Makespan
1: initialize makespan(R^*) = 0
2: for each resource R_j find the makespan using equation 6

$$makespan(R_j) = \sum_{i=1}^{n} ETE_{(i,j)} \quad (6)$$

where R_j is j^th resource and i is task id.
3: return MAX(R^*)

4.3 Initial Population

In this thesis, an initial population of individuals is generated randomly using the algorithm 3

4.4 Selection

In our GA, the roulette wheel selection method is used to select the population for the reproduction of the next generation.

4.5 Crossover

Our GA adopts a midpoint crossover (single point) operator with crossover probability 0.8, which is described in algorithm 4. In fig 4 show the chromosomes of parents and children before and after crossover respectively.

Algorithm 3 Generation of initial population algorithm

Input: population size(popsize), chomosome length(chlength)
Output: initial population,P
1: **for** $j = 1$ to $popsize$ **do**
2: **for** $i = 1$ to $chlength$ **do**
3: p(j,i) = round(random()*VM_MAX)
4: **end for**
5: **end for**
6: return P

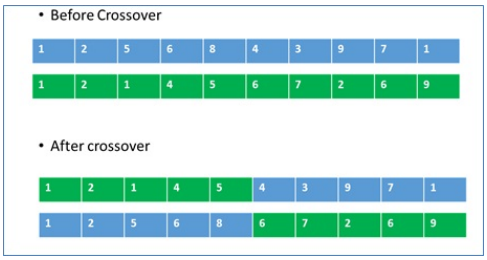

Fig. 4. Example of mid-point crossover(single point)

Algorithm 4 Mid-Point uniform crossover(Single Point) Algorithm

Input: two parent chromosome,C1,C2
Output: two child chromosome,CC1,CC2
1: cl ← length(C1)
2: crossover point, cp = cl/2
3: CC1 ← C1(1 : cp) U C2(cp : cl)
4: CC2 ← C1(cp : cl) U C2(1 : cp)
5: return CC1,CC2

4.6 Mutation

The mutation operator randomly picks up a gene in the chromosome and inverts the value of the chosen gene. Algorithm 5 shows how the mutation operator works. Constraints 1) make sure that each task will be assigned to one and only one virtual machine; constraints 2) guarantee that the total CPU workload on the VM_j will not exceed the maximum utilization capacity. In fig5 show an example, task t5 initially allocated to VM5, mutated to VM6.

Algorithm 5 Mutation at random point with 0.2 mutation probability

Input: a chromosome,C
Output: a mutated chromosome, CM
1: CM ← C
2: randomly generate a task id i, where 1 ≤ i ≤ $|C|$
3: randomly generate a real value between 0 and 1, mp
4: **if** (mp < 0.5) **then**
5: randomly generate a virtual machine j, where 1 ≤ j ≤ VM_MAX
6: replace CM(i) ← j
7: **end if**
8: output CM

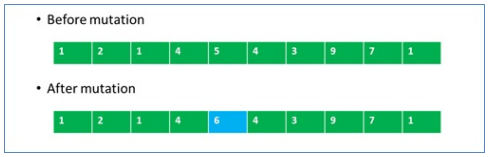

Fig. 5. Example of mutation at random point with 0.2 mutation probability

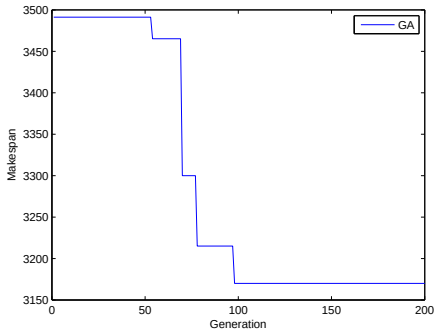

Fig. 6. Generation vs fitness level

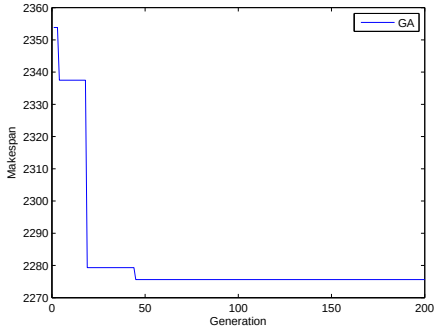

Fig. 7. Generation vs fitness level

4.7 Stopping condition

We have done the following simulation experiments for 500 tasks on 50 VMs, initial population size is 500, mutation probability is 0.2 and the single point crossover with crossover probability 0.8 to decide the stopping criteria. Figure 6, 7, and 8 show that the optimal result can be found after the 100 generation. In our GA, the stopping condition is decided by the maximum number of generations(MAX_GEN), which is equal to 100.

5. SIMULATION RESULTS

In this section, we simulated our experiments using the discrete event system modeling [22] for the genetic algorithm based task scheduling and conducted the various experiments. We have also compared those results with the random and RR scheduling model. The following parameters are taken in our simulation experiments: initial population size is 500, number of VMs are 50 and 100, stopping condition is 100 generations. A total 2500 tasks were generated using the ETC Model proposed by Zomaya [18]. The figure 9 and figure 10, shows the experimental result of Random scheduling, RR scheduling and GA based scheduling on 50 VMs and 100 VMs respectively.

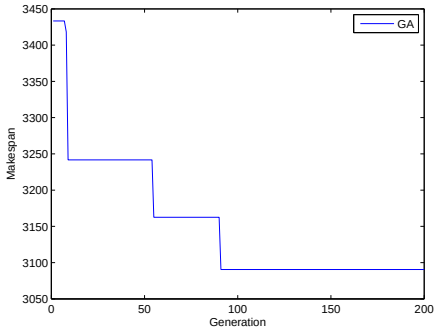

Fig. 8. Gneration vs fitness level

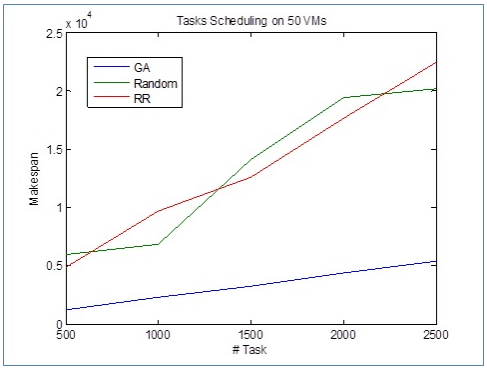

Fig. 9. Task scheduling on 50 VMs in cloud computing infrastructure.

initial population size = 500
number of VMs =50
generation = 100
stopping condition =100 generations
no of tasks = 2500 (generated using the ETC model)
crossover = single point crossover with 0.8 probalility
mutation = random point mutation with 0.2 probalility

initial population size = 500
number of VMs =100
generation = 100
stopping condition =100 generations
no of tasks = 2500 (generated using the ETC model)
crossover = single point crossover with 0.8 probalility
mutation = random point mutation with 0.2 probalility

6. CONCLUSIONS

The Task consolidation in cloud computing becomes a major research issue to utilize the ideal computing resources and hence the minimize the energy uses in cloud computing. The genetic algorithm is used the assigned the jobs to the VMs by minimizing the makespan. A number of experiments were conducted to examine the performance of genetic algorithms. This paper presents an experimental results for Poisson arrived 2500 tasks were scheduled using the Genetic Algorithm, Random, and RR. The outcomes of our experiments in (figure 9 and figure 10) show that the Genetic

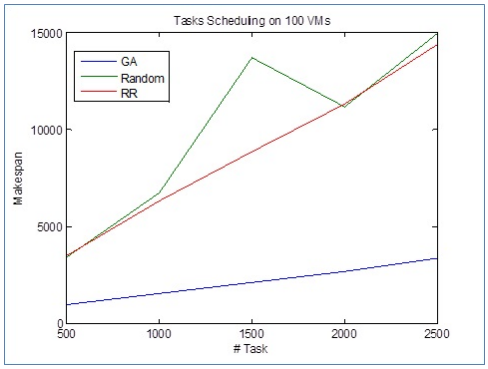

Fig. 10. Task scheduling on 100 VMs in cloud computing infrastructure.

Algorithm based scheduling model outperforms the existing Random and RR scheduling models.

7. REFERENCES

[1] Shoukat Ali, Howard Jay Siegel, Muthucumaru Maheswaran, and Debra Hensgen. Task execution time modeling for heterogeneous computing systems. In *Heterogeneous Computing Workshop, 2000.(HCW 2000) Proceedings. 9th*, pages 185–199. IEEE, 2000.

[2] Anton Beloglazov, Jemal Abawajy, and Rajkumar Buyya. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future Generation Computer Systems*, 28(5):755–768, 2012.

[3] Tracy D Braun, Howard Jay Siegel, Noah Beck, Ladislau L Bölöni, Muthucumaru Maheswaran, Albert I Reuther, James P Robertson, Mitchell D Theys, Bin Yao, Debra Hensgen, et al. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *Journal of Parallel and Distributed computing*, 61(6):810–837, 2001.

[4] Rajkumar Buyya, Anton Beloglazov, and Jemal Abawajy. Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges. *arXiv preprint arXiv:1006.0308*, 2010.

[5] Rodrigo N Calheiros, Rajkumar Buyya, and César AF De Rose. A heuristic for mapping virtual machines and links in emulation testbeds. In *Parallel Processing, 2009. ICPP'09. International Conference on*, pages 518–525. IEEE, 2009.

[6] Jeffrey M Galloway, Karl L Smith, and Susan S Vrbsky. Power aware load balancing for cloud computing. In *Proceedings of the World Congress on Engineering and Computer Science*, volume 1, pages 19–21, 2011.

[7] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[8] K. Hwang, G.C. Fox, and JJ Dongarra. *Distributed and Cloud Computing: From Parallel Processing to the Internet of Things*. Morgan Kaufmann, 2012.

[9] Qin-Ma Kang, Hong He, Hui-Min Song, and Rong Deng. Task allocation for maximizing reliability of distributed computing systems using honeybee mating optimization. *Journal of Systems and Software*, 83(11):2165–2174, 2010.

[10] Dilip Kumar and Bibhudatta Sahoo. Energy efficient heuristic resource allocation for cloud computing. *Artificial Intelligent Systems and Machine Learning*, 6(1):32–38, 2014.

[11] Dara Kusic, Jeffrey O Kephart, James E Hanson, Nagarajan Kandasamy, and Guofei Jiang. Power and performance management of virtualized computing environments via lookahead control. *Cluster computing*, 12(1):1–15, 2009.

[12] Young Choon Lee and Albert Y Zomaya. Energy efficient utilization of resources in cloud computing systems. *The Journal of Supercomputing*, 60(2):268–280, 2012.

[13] Liang Liu, Hao Wang, Xue Liu, Xing Jin, Wen Bo He, Qing Bo Wang, and Ying Chen. Greencloud: a new architecture for green data center. In *Proceedings of the 6th international conference industry session on Autonomic computing and communications industry session*, pages 29–38. ACM, 2009.

[14] Siriluck Lorpunmanee, Mohd Noor Sap, Abdul Hanan Abdullah, and Chai Chompoo-inwai. An ant colony optimization for dynamic job scheduling in grid environment. *International Journal of Computer & Information Science & Engineering*, 1(4), 2007.

[15] Peter Mell and Timothy Grance. The nist definition of cloud computing (draft). *NIST special publication*, 800(145):7, 2011.

[16] Zbigniew Michalewicz. *Genetic algorithms+ data structures= evolution programs*. springer, 1996.

[17] Kuntal Mukherjee and G Sahoo. Mathematical model of cloud computing framework using fuzzy bee colony optimization technique. In *Advances in Computing, Control, & Telecommunication Technologies, 2009. ACT'09. International Conference on*, pages 664–668. IEEE, 2009.

[18] Hisatoshi Ohmae, Yoshitomo Ikkai, Norihisa Komoda, Kuniyoshi Horiuchi, and Hidenobu Hamamoto. A high speed scheduling method by analyzing job flexibility and taboo search for a large-scale job shop problem with group constraints. In *Emerging Technologies and Factory Automation, 2003. Proceedings. ETFA'03. IEEE Conference*, volume 2, pages 227–232. IEEE, 2003.

[19] Ivan Rodero, Juan Jaramillo, Andres Quiroz, Manish Parashar, Francesc Guim, and Stephen Poole. Energy-efficient application-aware online provisioning for virtualized clouds and data centers. In *Green Computing Conference, 2010 International*, pages 31–45. IEEE, 2010.

[20] Ying Song, Hui Wang, Yaqiong Li, Binquan Feng, and Yuzhong Sun. Multi-tiered on-demand resource scheduling for vm-based data center. In *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pages 148–155. IEEE Computer Society, 2009.

[21] Shekhar Srikantaiah, Aman Kansal, and Feng Zhao. Energy aware consolidation for cloud computing. In *Proceedings of the 2008 conference on Power aware computing and systems*, volume 10. USENIX Association, 2008.

[22] Akshat Verma, Puneet Ahuja, and Anindya Neogi. pmapper: power and migration cost aware application placement in virtualized systems. In *Middleware 2008*, pages 243–264. Springer, 2008.

[23] Chee Shin Yeo and Rajkumar Buyya. A taxonomy of market-based resource management systems for utility-driven cluster computing. *Software: Practice and Experience*, 36(13):1381–1419, 2006.

[24] Albert Y. Zomaya and Yee-Hwei Teh. Observations on using genetic algorithms for dynamic load-balancing. *Parallel and Distributed Systems, IEEE Transactions on*, 12(9):899–911, 2001.