A Bio-inspired Approach for Power and Performance Aware Resource Allocation in Clouds

Rajesh Kumar[1], Ashok Kumar[1,a] and Anju Sharma[1]

[1]*Computer Science & Engineering Department, Thapar University, Patiala-147004, India*

Abstract. In order to cope with increasing demand, cloud market players such as Amazon, Microsoft, Google, Gogrid, Flexiant, etc. have set up large sized data centers. Due to monotonically increasing size of data centers and heterogeneity of resources have made resource allocation a challenging task. A large percentage of total energy consumption of the data centers gets wasted because of under-utilization of resources. Thus, there is a need of resource allocation technique that improves the utilization of resources with effecting performance of services being delivered to end users. In this work, a bio-inspired resource allocation approach is proposed with the aim to improve utilization and hence the energy efficiency of the cloud infrastructure. The proposed approach makes use of Cuckoo search for power and performance aware allocation of resources to the services hired by the end users. The proposed approach is implemented in CloudSim. The simulation results have shown approximately 12% saving in energy consumption.

1 Introduction

Cloud computing enables access to shared pool of computing resources through internet on pay per usage basis [18]. The pool of resources needs to be managed efficiently in order to cut power costs. Energy consumption can be reduced by efficient allocation of resources. The aim of a resource allocation approach is to find a resource that fulfills resource demands of a given service without effecting its performance. Generally, resource allocation approach is either system centric or user centric. The former approach is traditional and emphasises on the improvement of overall system performance such a utilization. Whereas, the latter concentrates on providing quality of service to the endusers. In this paper, a novel resource allocation approach is proposed that improves the utilization of resources to enhance energy efficiency of the cloud infrastructure. Utilization of a resource and its energy consumption are highly coupled. A resource with low utilization consumes unacceptable amount of energy compared to its energy consumption when it is fully or sufficiently loaded. According to [7], an idle resource consumes up to 70% of the power consumed at full utilization. Furthermore, average resource utilization of most of the data centers is about 15-20% [21, 13] which results in wastage of huge amount of energy. According to Koomey [15], "Total data center power consumption from servers, storage, communications, cooling, and power distribution equipment accounts for 1.7-2.2% of total electricity used in U.S. in 2010". Furthermore, today's data centers, in addition to their huge energy consumption, emit as much carbon dioxide as whole of Argentina. If left on their current path, data center carbon dioxide output will quadruple by the year 2020 [13]. While the cloud energy consumption is growing quickly, industrial organizations and researchers are finding ways to reduce energy consumption. The motivation for this work stems from challenges arising due to huge amount of energy consumption. The challenge is to enhance energy efficiency of the cloud infrastructure without effecting the performance of services delivered to the end users. The proposed work improves energy efficiency of the cloud by perspicacious allocation of resources to the services. Cuckoo search is employed for optimal allocation of resources to the services. The proposed approach is implemented in CloudSim. The simulation results have shown significant reduction in energy consumption.

Rest of the paper is organized as follows: Related work is presented in Section 2. In Section 3, power and performance aware resource allocation framework is discussed. Section 4 discusses Cuckoo search meta-heuristic. Section 5 elaborates resource allocation using Cuckoo optimization. Results for performance analysis are discussed is Section 6. Conclusion and scope for future work are presented in Section 7.

2 Related Work

Gao et al. [9] proposed linear programming based multi-objective ant colony based system for virtual machine

[a] Corresponding author: ashok.khungerr@gmail.com

placement to minimize resource wastage and power consumption. The authors used only CPU processing speed and memory requirements of a VM while allocating resources to it. Feller et al. [8] presented multi-dimensional ant colony optimization based job consolidation algorithm. The algorithm uses resource utilization history to predict future resource demands and dynamically overbooks the resources. The authors tested the proposed algorithm on homogeneous PMs. Gao et al. [9] proposed multi-objective ant colony system algorithm for virtual machine placement that minimizes total resource wastage and power consumption. Ant colony optimization technique for assigning real-time tasks to heterogeneous processors is proposed by Chen et al. [6]. Local search technique is applied to improve energy efficiency of the feasible assignment solution generated by the proposed assignment algorithm. Huang et al. [11] presented genetic algorithm based adaptive sub-optimal resource management scheme to estimate number of VMs required to provide desired level of service. Kansal and Chana [12] presented artificial bee colony based energy-aware resource allocation technique to efficiently manage and enhance the utilization of resources. The authors have claimed the reduction of energy consumption and execution time of applications. Xu and Fortes [23] proposed multi-objective VM allocation algorithm. The authors have taken CPU, and memory parameters for VMs and have claimed reduction in power consumption, thermal dissipation costs, and resource wastage. In [20, 4, 19, 10], the authors proposed energy-conscious consolidation heuristics in order to conserve energy and maximize resource utilization without affecting the performance of the system. Kusic et al. [16] proposed performance and power efficient resource-management approach based on look-ahead control method for virtualized heterogeneous environments. Prediction is employed for dynamic reallocation of resources.

The next section describes proposed resource allocation approach.

3 Power and Performance Aware Resource Allocation

Resource allocation is a process of allocating required amount of computing resources to services hired by the end users. Services are provided through virtual machines (VMs). Multiple VMs can be executed in a single PM by leveraging virtual machine monitor (VMM) such as XEN [1], KVM [14], or VMWare [22]. The proposed power and performance aware resource allocation (P2RA) approach strives to increase utilization of resources without effecting the performance of services delivered to the end users. Figure 1 shows framework of proposed approach.

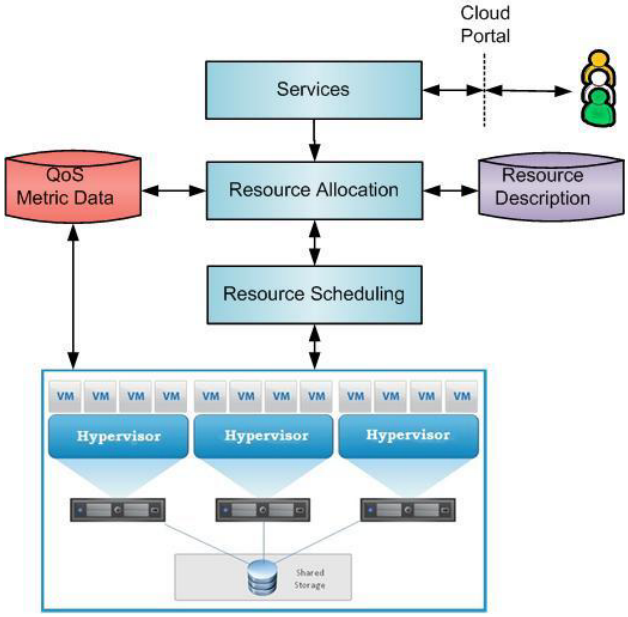

Figure 1. Energy and QoS aware resource allocation framework.

The following are the few assumptions that are taken into consideration.

i. Services are independent of each other and have computational requirements such as CPU, memory, and bandwidth.
ii. Each service can be allocated resources independently.
iii. Memory requirement of a service is rigid while CPU and network bandwidth requirements are fluid. A service can not be executed with less memory allocation than required.
iv. A service can be executed with less CPU and bandwidth allocation than the required capacity at the cost of reduced performance.
v. Idle PMs can be put to sleep mode to save energy. Energy consumption for switching a PM from *active* to sleep mode and vice versa is negligible and is not taken into consideration.
vi. Fraction of resources can be allocated to a VM.

The major components of P2RA approach are: (i) Services, (ii) Resource Allocation, (iii) Resource Scheduling, (iv) QoS Metric Data, and (v) Resource Description.

Services: The end users can hire the services through cloud portal. This module stores information about type and number of services hired by the end users.

Resource Allocation: This modules allocates resources to the services hired by the end users. A service is allocated to a resource that is most energy efficient. In order to find most suitable resource for service Cuckoo optimization is employed which discovers a resource employing Levy flight (LF). Allocation of service to the resource should result in minimal increase in power consumption while satisfying performance requirements. LF is a random walk that obeys Levy distribution. The reasons behind using LF for resource selection are:

i. Levy distribution has infinite mean and variance and hence LF are capable to explore the search space efficiently.
ii. LF is a stochastic process i.e. next state depends upon the previous state only.
iii. LF has scale invariant property and hence is used to model problems that exhibit clustering.

A random walk, consisting of series of consecutive random steps, is expressed as stochastic equation (1):

$$x_i(t+1) = x_i(t) + \alpha L(s, \lambda), \quad (1)$$

here, $x_i(t)$ and $x_i(t+1)$ represents a resource at time t and $t+1$, respectively. α is the scaling factor for step size s. Levy exponent, λ, is a constant. The step size s determines how far a final solution can go from initial solution in a fixed number of iterations. Obviously, if step size is large, next solution $x_i(t+1)$ will be far away from the current solution $x_i(t)$. The value of α is selected in compliance with the characteristic scales of the problem of interest.
Resource Scheduling: Scheduling modules controls the execution of the services. It schedules the services using round robin scheduling policy.
QoS Metric Data: It stores the usage statistics of resources. Usage statics of resources helps to make decisions about how to improve performance, energy efficiency, etc. of the cloud infrastructure under consideration.
Resource Description: This module stores information about the resources like type of resource, number of resources, etc.

3.1 Power Model

Power consumed by a resource is given by equation (2):

$$P_{total} = P_{dynamic} + P_{static}, \quad (2)$$

static power consumption (SPC), P_{static}, is due to leakage current and is independent of clock frequency and usage scenario. SPC can be reduced by switching idle resources to sleep mode [2]. However, dynamic power consumption (DPC), $P_{dynamic}$, is due to circuit activity and it depends on resource utilization. Power consumption of resource, given by equation (3), increases linearly with its utilization [3].

$$P = P_{idle} + (P_{max} - P_{idle})U , \quad (3)$$

where, P_{idle} is the power consumption when the resource is idle, P_{max} is the power consumption at 100% utilization, and P is the power consumption at utilization U ε(0, 1) of the PM. Power consumption of a resource, when it is idle, is about 70% of the power consumed at full load [3]. Whereas, high utilization of a resource causes performance degradation because services running on it do not get sufficient resources [3]. So, a resource should not be operated at too low or very high utilization.

3.2 Fitness Function

The proposed resource allocation approach allows to group services hired by end users over the small number of resources and thus turning off those resources that are not used. P2RA approach considers performance as well as energy consumption while allocating resources to the service. The suitability of the resource r for service s is evaluated from fitness function f (equation (4)).

$$f_{s,r} = \frac{\left[\frac{\Re_r^a}{\Re_s^d}\right]^{\theta}}{\left[\gamma \triangle E_{s,r} + (1-\gamma)\kappa_r(1 - \sum_{i \in S, i \neq s} \Re_{i,r})\right]^{1-\theta}}, \quad (4)$$

where, $\Re_r^a$ is available processing power of resource r, $\Re_s^d$ is processing demand of service s, $\Delta E_{s,r}$ is energy contribution of service s on resource r, κ_r is energy affinity, $\Re_{i,r}$ is fraction of processing power allocated to service i on resource r, and $0 \leq \gamma \leq 1$ is a weight value to control the importance of energy contribution of service s. $\frac{\Re_r^a}{\Re_s^d}$ gives performance affinity which is desired to be greater than 1 for better performance. $0 \leq \theta \leq 1$ is used to control the importance between performance and energy. If θ is set to 1 then performance is considered while allocating resources. However, if it is set to 0 importance is given to energy consumption. Energy affinity, κ_r, is set to energy consumption of resource r when it is idle which inclines P2RA towards resources with lower energy consumption in idle state.

4 Cuckoo Search

Cuckoo search is a nature inspired population based meta-heuristic algorithm developed by Xin-She Yang and Suash Deb in 2009 [24]. It is based on brood parasitic behavior of bird cuckoo. Cuckoo lay eggs in the nests of other host birds with amazing capability of selecting recently spawned nests. On the other hand, some host birds upon recognizing alien eggs either through them out or build new nests in other locations. Cuckoo reproduction process is simplified by three idealized rules:

- Each cuckoo lays one egg at a time and dumps it in a randomly chosen nest.
- Nests with high quality eggs are carried over to the next generations.
- The host bird can discover the alien eggs with probability P_a ε (0, 1). Once an alien egg is discovered either the egg is thrown out or the nest is abandoned. A fraction P_a of the n poor quality nests are replaced by new nests.

5 Resource Allocation using Cuckoo Optimization

In the proposed resource allocation approach, services are assumed eggs and resources are considered nests of host birds. Suppose, a set *S* of services and each service has some requirement attributes such as CPU, memory, and network bandwidth. The services are required to be mapped on set *R* of resources. Each resource *r* ε *R* has some processing speed, memory and network bandwidth, respectively. Our aim is to allocates resources to services in such a way to reduce overall energy consumption without effecting performance of the services. The major steps followed for QoS and energy aware resource allocation to services are:

a. Consider population (search space) of *n* resources.
b. Select a resource i from the search space using LF (equation (1)).
c. Randomly choose a resource j from search space of *n* resources.
d. Calculate fitness value $f_{s,i}$ of resource *i* for service *s* using equation (4).
e. Calculate fitness value $f_{s,j}$ of resource *j* for service *s* using equation (4).
f. If $f_{s,i} > f_{s,j}$ then resource i is considered better than resource *j*.
g. Remove all those resources from the search space whose utilization is above upper green threshold (UGT).
h. Add resources, equal to number of removed resources, to the search space.
i. Rank the solutions and find current best solution.
j. Repeat step (c) through (i) until maximum iteration and the best solution does not change for three consecutive iterations. Iterations.
k. Instantiate the service on the best solution (resource).

The pseudo code for resource allocation using Cuckoo optimization is given in Algorithm 1. The steps (b) through (k) are repeated until all services are allocated resources or until maximum number of iterations have been performed. A resource is discovered that best suits the requirements of the service. Initially, a resource is selected from search space using LF (equation (1)). The selected resource represents a tentative solution for resource allocation problem. Our aim is to discover the best solution (resource) which results in minimum increase in energy consumption while satisfying QoS requirements of the service. Second resource is then selected randomly from the search space. Out of the two selected resources, a resource with lesser increase in power consumption is considered better. Then some of the resources, having utilization greater than UGT, are removed from population and equal number of new resources are added to the population. Resources removed from the population are not considered for resource allocation in subsequent iterations. A resource that is removed from population can be added to the population later on, when its utilization falls below lower green threshold (LGT). Population is a set of fittest *n* resources, out of which a resource is discovered for allocating resources to a job under consideration. Total number of resources in use are variable and changes with number of jobs and their resource requirements.

Algorithm 1 Pseudo code for Resource allocation using Cuckoo Search

```
Input: List of services S_j(1 ≤ j ≤ N_j)
Generate initial population of n resources (R_i, i = 1, 2, ..., n).
set I ← 0
while (I < MaxIteration ) do
  Select a service s
  Select resource i (R_i) using Levy flight equation (1).
  Select resource j (R_j) randomly.
  Determine fitness f_{s,i} and f_{s,j} of resources i and j, respectively.
  if (f_{s,i} > f_{s,j}) then
    Replace solution j with solution i
  end if
  for i = 1 to n do
    if (R_i.Utillization > UGT) then
      Remove R_i from population of resources
      Add a new resource to the population of resources at location i
    end if
  end for
  Rank the solutions and find the best solution (resource).
  update I ← I + 1
end while
Allocate service to the resource referred by best solution.
```

6 Performance Evaluation and Comparative Analysis

The proposed approach is implemented in CloudSim. CloudSim is a framework for modeling and simulation of cloud computing infrastructure and services [5]. For performance analysis, P2RA is compared with First Fit Decreasing (FFD). The specifications of four types of resources, which are simulated using PowerHost class of CloudSim, are as given in Table 1. Power models corresponding to these servers, available in CloudSim, are used to evaluate energy consumption. To conduct comparative analysis, P2RA and FFD are tested with services having different resource requirements. Simulation is repeated 25 times and in each simulation run, parameters are assigned values as per Table 2.

Table 1. Specification of Resources.

Type	CPU	Cores	RAM	Storage	BW
R1	2933	4	8	500	10
R2	3067	4	8	500	10
R3	2933	12	12	500	10
R4	3067	12	16	500	10

CPU, processing speed in mips; Cores, number of processing cores; RAM, random access memory in GB; Storage, Permanent storage capacity in GB; BW, network bandwidth in gbps.

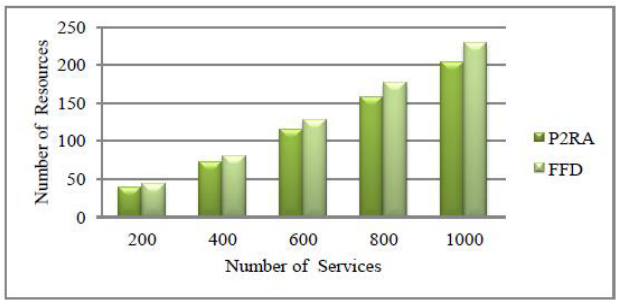

Figure 2. Comparison of number of Resources required for given number of Services

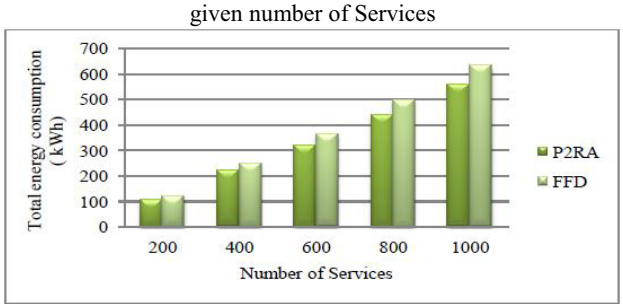

Figure 3. Comparison of total energy consumption

Table 2. Simulation parameters.

Number of services (VMs)	200-1000	Varied in simulation runs
Number of resources	100-200	Number of resources may also vary
Simulation Span	86400s	Time period of simulation run
Idle Time	10 min.	Time to switch PM to sleep mode
UGT	0.90	Upper Green Threshold limit
LGT	0.20	Lower Green Threshold limit

Figure 2 shows the number of PMs used by FFD and P2RA to fulfill the requirements of given number of services. Number of services are varied from 200 to 1000 in step of 200. It is observed that number of resources used increases with number of resources. It has been experimentally established that P2RA uses 10.12% lesser number of resource than and FFD.

Figure 3 shows total energy consumption comparison of FFD and P2RA. It has been observed that P2RA is approximately 12% more energy e_cient than FFD. Therefore, P2RA is more energy efficient and environment friendly.

Figure 4 shows comparison of time taken by FFD and P2RA to map given number of services to resources. Number of services are varied from 200 to 1000. It is observed that

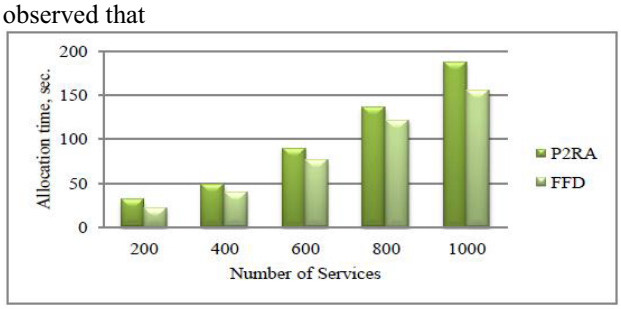

Figure 4. Comparison of total mapping time

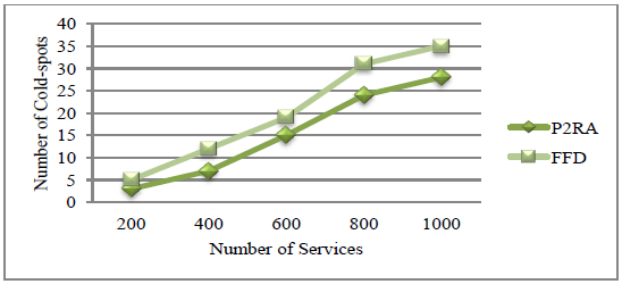

Figure 5. Comparison of number of Cold-spots

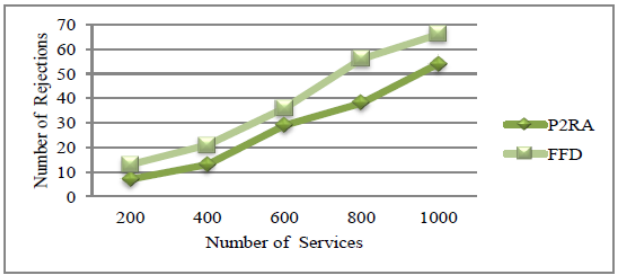

Figure 6. Comparison of number of Rejections

P2RA takes more time to map given of services. This due to optimal allocation of resources. However, energy consumed during migration is negligible compared to energy saving achieved P2RA approach. Figure 5 shows the comparison of number of cold-spots observed for FFD and P2RA with respect to the number of services submitted for execution. A PM is defined as a cold-spot if its resource utilization is below LGT (=20%). The 20% limit is taken because average resource utilization of typical data center is between 20% and 30% [17]. The number of cold-spots portrays the extent of resource wastage. Lesser number of cold-spots createdin P2RA shows its ability to allocate the resources efficiently. Figure 6 shows the comparison of number of job rejections determined for FFD and P2RA as the number of services increases from 200 to 1000. A service is rejected when sufficient resources are not available for its execution. It depicts that number of services that are not admitted due to unavailability of resources. In case of P2RA lesser number of rejections are observed than FFD.

7 Conclusion and Future Scope

In this work, a bio-inspired resource allocation approach for enhancing energy efficiency and QoS of Clouds is introduced. The services are allocated resources in compliance with their demands. Cuckoo optimization is employed for optimal allocation of resources. The proposed resource allocation approach is implemented in CloudSim and tested with services having different QoS and resource requirements. The results have shown 12% saving of energy consumption In future, the proposed resource allocation methodology can be extended by identifying dependency among the services. The performance of proposed methodology can be evaluated on private cloud set up using any open source cloud environment such as OpenNebula, Eucalyptus, etc.

References

1. P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield, SIGOPS Oper. Syst. Rev., **37**, 164-177(2003).
2. A. Beloglazov. Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing. PhD thesis, University of Melbourne, (2013).
3. A. Beloglazov, J. Abawajy, R. Buyya, FGCS, **28**,755-768(2012).
4. A. Beloglazov, R. Buyya, IEEE Transactions on Parallel and Dirtributed Systems, **24**, 1366-1379(2013).
5. R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose, R. Buyya, Software: Practice and Experience (SPE), **41,** 23-50(2011).
6. H. Chen, A. M. K. Cheng, Y. Kuo, JPDC, **71,**132-142(2011).
7. X. Fan, W. D. Weber, L. A. Barroso, In Proceedings of the 34th Annual International Symposium on Computer Architecture, ISCA '07, 13-23(2007).
8. E. Feller, L. Rilling, C. Morin, In 12th IEEE/ACM International Conference on Grid Computing (GRID), 26-33(2011).
9. Y. Gao, H. Guan, Z. Qi, Y. Hou, L. Liu, JCSS, **79**, 1230-1242(2013).
10. C. H. Hsu, S. C. Chen, C. C. Lee, H. Y. Chang, K. C. Lai, K. C. Li, C. Rong, In IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), 115-121(2011).
11. C. J. Huang, EAAI, **26**,382-389(2013).
12. N. J. Kansal, I. Chana, CCPE, **27,** 1207-1225(2015).
13. J. M. Kaplan, W. Forrest, N. Kindler, Technical report, McKinsy & Company, 2008.
14. A. Kivity, Y. Kamay, D. Laor, U. Lublin, A. Liguori, In Proceedings of the Linux Symposium, **1**, 225-230(2007).
15. J. Koomey. Growth in data center electricity use 2005 to 2010, 2014.
16. D. Kusic, J.O. Kephart, J.E. Hanson, N. Kandasamy, G. Jiang, In International Conference on Autonomic Computing, ICAC '08., 3-12(2008).
17. D. Meisner, B. T. Gold, T. F. Wenisch, In Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS XIV, 205-216(2009).
18. P. Mell, T. Grance. The NIST definition of cloud computing, 2011.
19. R. Nathuji, K. Schwan, ACM SIGOPS Operating Systems Review, **41**,265-278(2007).
20. S. Takeda, T. Takemura, IPSJ Transactions on Advanced Computing Systems, **3**, 138-146(2010).
21. W. Vogels. Beyond server consolidation. Queue, **6**,20-26(2008).
22. B. Walters. VMware virtual platform. Linux J., 1999(63es), 1999.
23. J. Xu, J. A. B Fortes, In IEEE/ACM International Conference on Green Computing and Communications (GreenCom), 179- 188(2010).
24. X. S. Yang, S. Deb. Cuckoo Search via Lvy Flights. In NaBIC, 210-214 (2009).