IJCSI International Journal of Computer Science Issues, Volume 15, Issue 6, November 2018
ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784
www.IJCSI.org https://doi.org/10.5281/zenodo.2544616

A new conception of load balancing in cloud computing using Hybrid heuristic algorithm

Hamza Rahhali[1], Mostafa Hanoune[2]

[1]Laboratory of Information Technology and Modeling
Hassan II University – Casablanca, Faculty of Sciences Ben M'sik
Casablanca Morocco

[2]Laboratory of Information Technology and Modeling
Hassan II University – Casablanca, Faculty of Sciences Ben M'sik
Casablanca Morocco

Abstract

This article introduces the last improvement in the field of hybrid heuristics designed to optimize the resources related to the allocation of user requests (tasks) in the cloud environment. We propose an efficient method for providing a load balancing in the cloud computing environment based on combining the global search abilities of Genetic Algorithm (GA) with different heuristics such as Tabu Search Algorithm, Simulated annealing Algorithm, ABC Algorithm, ACO Algorithm and PSO Algorithm. The experimental results showed that the proposed algorithm has high efficiency for reaching a load balancing in Cloud computing.

Keywords: *Load balancing, Cloud computing, Virtual machine, task allocation, CloudSim.*

1. Introduction

Cloud Computing (CC) has become an essential part of the world interest during the past decades ([24], [25]). It is an on demand services in which resources, information and software are provided as per clients need.

Nowadays there are number of challenges faced by cloud computing, and the distribution of the dynamic workload over various nodes without overloading a single node is one of the most issues in the Cloud. To deal with this problem, load balancing is an efficient technique to equally distribute load among the resources, and it helps systems and assets by giving a Maximum throughput with minimum response time.

In order to provide a load balancing in the cloud computing environment, we suggest an efficient strategy merging the global search abilities of Genetic Algorithm (GA) with different heuristics. The goal behind the use of the heuristics is to find a good solution in a sensible time frame that is sufficient for taking care of the current problem.

In this paper our main objective is to minimize running time and consumed energy using Particle Swarm Optimization, Ant Colony Optimization, Artificial Bee colony, Tabu Search, Simulated Annealing and Genetic Algorithm hybridization technique. The overall paper is arranged in a planned way as follows: As a matter of major importance, we present the problem formulation. Next, we present the types of Meta-heuristic load balancing algorithm. Moreover, we give some heuristic and hybridization strategies which exist in the literature. Besides, we present our proposed method that is intended to solve the problem. Section 6 provides a detailed description of CloudSim simulator. Section 7 describes the simulation and exhibit a comparative study is in order to provide perfect conclusions. At last, some finishing remarks are talked about.

2. Problem Formulation

Cloud computing is a new technique to provide online resource dynamically when user demands for it. But when number of users is increase at that time load balancing is main issue for cloud computing. Load can be balance if we use proper scheduling management technique. Load balancing can be done at different levels of cloud computing. In this paper we are focusing on load balancing among VMs. Our main objective in this paper is to minimize the running time with the help of proper load balancing algorithm. Let T= {Task1, Task2, …, M}is a set of M task to be assign and process on different VMs. Here VM= {VM1, VM2,….K} is a set of K different VMs. We use non preemptive scheduling approach mean when one VM is processing one cloudlet until it will not complete its execution that VM is not assign to another cloudlet [1].

3. States of Arts

A Metaheuristic is an optimization algorithm went for taking care of difficult problems for which no more productive

traditional technique is known. Metaheuristics are frequently algorithms utilizing probabilistic inspecting. They try to locate the global optimum of a hard optimization problem, without being caught by the nearby optima. There are a wide range of Metaheuristics, ranging from simple local search to complex global search algorithms. These techniques, however, utilize a high level of reflection, enabling them to be adjusted to an extensive variety of different issues. There is a substantial number of surely Metaheuristics in the literature, we find evolutionary algorithms, among which: there are evolution strategies, differential evolution algorithms, distribution estimation algorithms, and genetic algorithms. There are also other Metaheuristics such as the ant colony, the simulated annealing, the Tabu Search, and the particle swarm optimization, etc. In our examination, we settle on six Metaheuristics. The simulated annealing technique since it can manage discretionary frameworks and cost capacities, it measurably ensures finding an ideal solution, and it for the most part gives a decent solution.

The Tabu Search because it reduces in the number of circuit simulations required to find a feasible solution. Moreover, we used the genetic algorithm for several reasons namely the capacity of simultaneous optimization of the multiple phases or properties of a material in a single run, the facility of the incremental re-optimization of the whole system as more data is made available for either additional phases or material properties not included in previous runs, and the effective global optimization in the presence of various local minima in the parameter space. The particle swarm optimization (PSO) is utilized to take care of optimization problems with the concept of particles. All of particles have fitness regards which are assessed by the fitness function to be streamlined, and have velocities which coordinate the flying of the particles. The ant colony optimization (ACO) is a probabilistic method for taking care of computational issues which can be decreased to discovering great ways through graphs.

The ACO gives a Positive Feedback accounts for rapid discovery of good solutions and guarantees the convergence. Artificial bee colony (ABC) algorithm is an improvement strategy that simulates the searching conduct of honey bees, and has been effectively connected to a few handy issues, this algorithm provide the ability to explore local solutions and the ability to handle the objective costs.

3.1 Ant Colony Optimization Algorithm

This algorithm, as its name states, is based on the behavior of ants to detect the location of under-loaded or over-loaded nodes. Then it will update the resources utilization table [2]. This algorithm is known for its scalability, but has low throughput [3].

Algorithm 1 ACO Algorithm

Input: An instance P of a CO problem model P = (S, f,Ω).
Initialize_Pheromone_Values (T)
B←NULL
While termination conditions not met **Do**
Giter← NULL
For j=1 **to** na **Do**
S ←Construct_Solution(T)
If S is a valid solution **then**
S ←Local_Search(s) {optional}
If (f (S) < f(B)) or (B= NULL) **then**
B← s
Giter← GiterU{S}
End if
End if
End for
Apply_Pheromone_Update(T ,Giter ,B)
End while
Return the best so far solution B

3.2 Particle Swarm Optimization Algorithm

Particle swarm depends on the algorithm portrayed in Kennedy and Eberhart [4], utilizing changes proposed in Mezura-Montes and Coello [5] and in Pedersen [6].

The Particle swarm algorithm starts by making the first particles, and doling out them beginning velocities.

It assesses the objective function at every particle location, and decides the best (least) function value and the best location.

It picks new velocities, in light of the present velocity, the particles' individual best locations, and the best locations of their neighbors.

It at that point iteratively refreshes the particle locations (the new location is the old one or more the velocity, altered to keep particles inside limits), velocities, and neighbors.

Cycles continue until the point when the algorithm achieves a ceasing basis.

Algorithm 2 PSO Algorithm

For every Particle
Intialize particle
End
Do
For every Particle
Calculate fitness value
If the fitness value is superior to anything the best fitness value (pBest) in history
Set current value as the new pBest
End
Pick the Particle with the best fitness value of the considerable number of particles as the gBest
For every Particle
Calculate particle velocity
*vel[] = vel[] + lf1 * rand() * (pbest[] – present[]) + lf2 * rand() * (gbest[] – present[])*
Update particle position
present[] = persent[] + vel[]
End

While minimum error criteria or maximum iterations is not achieved

---

vel[] is the particle velocity, *present[]* is the present particle (solution).
Rand () is a random number between *(0,1).*
lf1, lf2 are learning factors. Usually *lf1 = lf2 = 2*

3.3 Simulated Annealing Algorithm

Annealing is implied as treating certain combinations of metal, glass by warming over its liquefying point, holding its temperature, and afterward cooling it continuously until the point that it sets into a perfect crystalline structure. This physical/chemical procedure delivers great materials. The simulation of this system is known as simulated annealing (SA) ([8], [14]). The imperfection free crystal state identifies with the general least vitality course of action

There is a similitude of SA with an optimization approach. In general, the simulated annealing algorithms work as takes after. At each time step, the algorithm randomly choose a solution close than the current one, qualifies its quality, and while later moves to it or to stay with the present solution in light of both of two probabilities between which it picks based on the way that the new solution is preferable or more terrible over the present one. In the midst of the interest, the temperature is continuously reduced from a fundamental positive motivating to zero and influences the two probabilities: at every progression, the probability of moving to a better new solution is either kept to 1 or is changed towards a positive esteem; rather, the probability of moving to a worse new solution is dynamically changed towards zero. Here is the detailed algorithm:

Algorithm 3 SA Algorithm

Initialize a solution X
$n \leftarrow 0$
repeat
  $i \leftarrow X$
  choose j at random from the neighbors of i
  if $f(j) \leq f(i)$ **then**
    $X \leftarrow j$
  else
    if *(Random <exp(f(i)-f(j)/T)* **then**
      $X \leftarrow j$
    end if
  end if
  $n \leftarrow n+1$
until Stop criteria is met
return ***X***

3.4 Tabu search

The Tabu search proposed by Glover [25], it is a Metaheuristic that associates an area heuristic search strategy to examine the arrangement space past adjacent optimality. One of the real parts of Tabu Search is its use of flexible memory, which makes more versatile search conduct. Tabu search can be viewed as beginning correspondingly as standard adjacent or neighborhood search, proceeding iteratively from one point (arrangement) toward another until the point when a picked end criterion is fulfilled. Each arrangement x has a related neighborhood $N(x) \subset X$, and each arrangement $x' \in N(x)$ is come to from x by an action called a move. Here is the algorithm:

Algorithm 4 Tabu search

Choose an initial solution i in S (the set solutions)
$X \leftarrow i$ and $k \leftarrow 0$
repeat
  apply $k \leftarrow k + 1$ and generate a subset solutions in $N(i, k)$
  choose the best solution i' among the set of neighboring solutions $N(i, k)$
  $i \leftarrow i'$
  if $f(i) \leq f(X)$ **then**
    $X \leftarrow i$
  end if
  update the T list of Tabu movements
until Stop criteria is met
return ***X***

3.5 Genetic Algorithm

Genetic algorithm (GA) is a Metaheuristic pushed by the method of natural determination that has a place with the bigger class of transformative algorithms (EA).
Genetic algorithms are routinely used to convey high-quality solutions for advancement and pursuit issues by relying upon bio-inspired operators for example, mutation, crossover and selection.
The procedure of regular selection begins with the selection of fittest people from a populace. They convey posterity which gain the qualities of the guardians and will be added to the new generation. In the occasion that parents have better fitness, their posterity will be better than parents and have a large chance at surviving. This technique nonstop emphasizing and toward the end, a generation with the fittest individuals will be found. This idea can be associated for a search problem. We consider an arrangement of answers for an issue and pick the arrangement of best ones out of them. Five stages are considered in a genetic algorithm:

- Initial population
- Fitness function
- Selection
- Crossover
- Mutation

Here is the algorithm:

Algorithm 5 Basic Genetic Algorithm

```
Initialize population
repeat
  repeat
    crossover
    mutation
    phenotype mapping
    fitness computation
  until population complete
  selection of parental population
until termination condition
```

3.6 Artificial bee colony Algorithm

The ABC algorithm is a swarm based, meta-heuristic algorithm based on the model first proposed by [9] on the foraging behavior of honey bee colonies.
The ABC as a optimization tool gives a populace based pursuit system in which individuals called foods positions are altered by the artificial bees with time and the bee's point is find the places of food sources with high nectar amount and finally the one with the highest nectar.
In this algorithm each artificial bees represent a possible solution of the problem, which collaboratively solve complex combinatorial optimization problem by exchanging the information [10].

Algorithm 6 Artificial Bee colony Algorithm

Initialize algorithm and problem parameters (food source positions, xi (i = 1, 2, …, SN).
Evaluate the fitness (fit(xi)) of the population
cycle = 1
Repeat
Employed Bees Phase Calculation.
For each employed bee
Produce a new food source position vi.
Calculate the fitness value fit(i).
Apply a greedy selection mechanism between xi and vi.
End for
Calculate the probability values pi for the solution.
Onlooker Bees Phase Calculation.
For each onlooker bee
Choose a food source depending on pi.
Produce a new food source position vi.
Calculate the fitness value fit(i).
Apply a greedy selection mechanism between xi and vi.
End for
Scout Bees Phase Calculation.
If the scout solution is better than the employed solution **then** the employed solution is replaced by a new random source position.
End if
Memorize the best solution achieved so far.
cycle = cycle + 1.
Until cycle = Maximum Cycle Number.

Table 1: Parametric comparison of load balancing algorithms

Method	Challenge	Advantages
ACO	Probability distribution changes by iteration and collective interaction of population of agents.	Greedy heuristic helps find acceptable solution in early stages
PSO	Dynamic load balancing by moving tasks form overloaded Vms by minimizing execution and transfer time	Minimize the makespan
SA	Few local minima, Slow.	Can deal with arbitrary systems and cost functions, Gives a "good" solution.
Tabu	Local search procedure, aspiration conditions, maximum size of tabu list, stopping rule	For large and more difficult problems;Can be applied to both discrete and continuous solution spaces; Allow non improving solution to be accepted in order to escape from a local optimum[11].
GA	Assumes that the jobs are of same priority	Find best fit solutions
ABC	Achieves global load balancing through local server actions	Improved scalability

4. Heuristic Hybridization

The literature proposes three sorts of approaches to deal with these kinds of problems. In the first place, exact methods tend to the optimal solution in the search space and they have an exponential running time in the most pessimistic scenario. We cannot handle NP-Hard problems utilizing exact methods. In the second place, the heuristic methods wich are solution procedures that can rapidly give a solution in a sensible quality. We will speak in insight about this type of determination. The third class of methodologies of resolution is an exceptional type of heuristics, yielding a worst-case upper bound of the proportion between the cost of an inexact solution and the cost of optimal solution. This method created an error guarantee.

4.1 Heurisitics

The major focus of a heuristic is to give a solution in a sensible time span that is adequate for dealing with the present problem. This solution may not be the best of the extensive number of solutions for this issue, or it might just estimate the right solution. In any case, it is up until now a critical way

since discovering it does not require a prohibitively drawn out stretch of time.
Heuristics may give results independent from any other one, or they may be used as a piece of conjunction with upgrade calculations to improve their adequacy (e.g., they may be used to create great seed regards). Results about NP-hardness in programming engineering settle on heuristics the primary appropriate decision for a collection of complex optimization problems that should be routinely handled in obvious applications.
Heuristics underlie the whole field of Artificial Intelligence and the computer reenactment of thinking, as they may be used as a piece of conditions where there are no known calculations.

4.2 Hybridization Type

In this area we will revolve around two clarifications behind hybridization in Evolutionary Algorithms: Improvement in execution of the EAs and quality of the got solutions.
Much work has been given to the hybridization of various recombination methods in Genetic Algorithms. Some awesome cases in this line are the fuzzy logic controller proposed in [12] to control the help of different crossover operators or the investigations by Hong [13] where the aim of a few hybrid crossover operators adjusted in perspective of the progress brought into the population by using each one of them. Diverse algorithms propose island GAs where every island builds up a population by strategies of recombination operators with different characteristics, aiming to achieve a decent exchange off amongst investigation and exploitation techniques by controlling transitory procedures [20].At last, a couple of examinations propose the use of two diverse populaces: one for the problem itself and another for the operators that will be used [15].
However, hybridization isn't constrained to occurring inside the same formative perspective. A few examinations propose Hybrid Evolutionary Algorithms where no less than two distinct algorithms collaborate through the inquiry procedure. This is the situation for the GA-EDA algorithm [16] where a GA and an EDA are joined and associated with the assurance of both discrete and continuous problems. Shi et al. [17] proposed a hybrid EA-PSO algorithm where the two subsystems are done in parallel, and two or three individuals are exchanged each age. Tseng and Liang [21] proposed a hybrid approach that joins Ant Colony Optimization (ACO), a Genetic Algorithm and a Local Search for the Quadratic Assignment Problem (QAP). In their tests, elective periods of ACO and GAs are executed. The pheromone regards crucial for the ACO are likewise refreshed while the algorithm is in the GA stage to ensure the right behavior of the ACO. The Local Search improves solutions conveyed by the two algorithms.
Finally, a third technique for hybridization in Evolutionary Algorithms deals with the encoding of solutions. Studies finished by [18] on the trouble of various optimization problems have evaluated one of the parts of problem complexity by relating the contrast between fitness function values and the Euclidean distance in the solution space. Test comes about to show that an interpretation of the fitness scene can make a problem simpler or harder to deal with. However, few works have considered this issue. In [19] the authors propose a Genetic Algorithm to settle various optimization functions where the people can be encoded with a Cartesian or a pseudo-polar coding and are associated by using them two. In [22], five Genetic Algorithms are united to clarify a few event of the TSP (Travelling Salesman Problem). Four of these GAs used a whole number way portrayal, while other used a real ranking encoding.

5. Proposed Algorithm

A hybrid method is a search technique comprising of no less than two different search strategies. It consists in exploring the respective benefits of a few methods by consolidating their algorithms as per a synergetic approach.
A hybrid method may be bad or good relying upon the decision and parts of its segments. To characterize a compelling hybrid method, one must know how to describe the favorable circumstances and the points of confinement of every technique.
The procedure to continue to an optimal solution relies upon heuristic search in a given population and the direction of pursuit is picked by the diverse operators used

5.1 GA-Tabu

It is a mix of basic GA and Tabu search algorithm. The object is to course the optimization task into two areas, the GA at first plays out the search and after that the refinement is done by the Tabu search algorithm. Both the algorithm continue running in parallel, after n iteration of Tabu search, the nearby optimal solution is injected into the present generation. The Tabu search method finds the nearby minima which supplements the GA to get the global minima. This is the full algorithm:

Algorithm 7 GA-Tabu Search Algorithm

```
Initialize population
repeat
  repeat
    crossover
    Tabu search
    phenotype mapping
    fitness computation
  until population complete
  selection of parental population
until termination condition
```

5.2 GA-SA

Inside our Hybrid algorithm, Simulated annealing (SA) is used to optimize each single of the best N individuals in the created

population. However, it is not vital to use the SA method to optimize the best N people at each generation, for the most part the Hybrid algorithm will have a very long run time cost. The same hybridization strategy as GA-Tabu is utilized. Here is the detailed algorithm:

Algorithm 8 GA-SA Algorithm

```
Initialize population
repeat
  repeat
    crossover
    simulated annealing
    phenotype mapping
    fitness computation
  until population complete
  selection of parental population
until termination condition
```

5.3 GA-ABC

In order to achieve balance between exploration and exploitation a novel improvement is integrated in GA algorithm. It is observed that, the efficiency of ABC in solving the load balancing can be enhanced by combining another technique Genetic Algorithm (GA) to suit the structure of the problem. In this work, the ABC algorithm contributes during the mutation phase of the genetic algorithm. The consumed energy will be minimized by execution of proposed hybrid ABC-GA approach.

Algorithm 9 GA-ABC Algorithm

```
Initialize population
repeat
  repeat
    crossover
    ABC
    phenotype mapping
    fitness computation
  until population complete
  selection of parental population
until termination condition
```

5.4 GA-ACO

We propose a novel component algorithm that joins genetic algorithms (GA) and ant colony optimization (ACO) for quicker and better search capacity. The hybrid algorithm takes profit from both ACO and GA techniques. The process begins by generating a number of ants and a population in the mutation step of GA. The proposed algorithm is easily executed and as a result of utilization of a straightforward classifier in that, its computational complexity is low. Here is the algorithm:

Algorithm 10 GA-ACO Algorithm

```
Initialize population
repeat
  repeat
    crossover
    ACO
    phenotype mapping
    fitness computation
  until population complete
  selection of parental population
until termination condition
```

5.5 GA-PSO

The fundamental idea of this technique is to coordinate P.S.O and G.A methods as appeared in the accompanying algorithm. The algorithm runs P.S.O within the G.A treatment. The disadvantage in P.S.O is that when we select an initial population, say 10, we are constrained to the 10 particles. The approach we have embraced is genetic algorithm in which P.S.O can be included by an operation called mutation if we get trapped in the initial population.

Algorithm 11 GA-PSO Algorithm

```
Initialize population
repeat
  repeat
    crossover
    PSO
    phenotype mapping
    fitness computation
  until population complete
  selection of parental population
until termination condition
```

Cloud Sim

The proposed load balancing algorithms is simulated using CloudSim Tool. CloudSim is an open source simulator which has been developed by Gridbus project team and the grid Laboratory of the University of Melbourne in Australia. The CloudSim can run on Linux and Windows systems [23]. The Experiments consists of different Parameters. Here we take only one data center and different task and VM, also we consider different number of iteration for getting better result. Based on different parameters in the Next section we will show the results.

Computational Results

We have compared the performance of the hybrid GA-Tabu, the hybrid GA-ACO , the hybrid GA-ABC, the hybrid GA-PSO and hybrid GA-SA with some of the recent algorithms in

the literature that are shown in the next Table in terms of execution time and the consumed energy.

The properties of the physical machines are shown below:

Parameter	Value used
Number of physical machines	110
Maximum capacities of CPU (in MIPS) and Memory (in MB)	2000 MIPS / 2500Mo
Powers delivered by physical machines according to CPU frequencies	Heterogeneous

The properties of virtual machines are presented below:

Parameter	Value used
Number of virtual machines	400
Capacities (CPU in MIPS and Memory in MB) of virtual machines	200/400/600/800 (MIPS and MB)
The maximum percentage of reconfiguration of the CPU capacity accepted by the virtual machines	20% of the CPU capacity of the virtual machine concerned
Number of instructions to be executed by virtual machines	between 10 and 110 the CPU capacity of the virtual machine
Maximum migration time	1 second

Finally, the configuration of the genetic algorithm for allocating the virtual machines at each moment of reallocation is as follows:

Parameter	Value used
Size of the random population of departure	1500
Size of the working population	120
Number of mutations	100
Number of crossings	90
Fixed number of generations	600

A simple neighborhood is used for our problem, it is the set of solutions that can be built by swapping two virtual machines in a given solution.

Table 2: The performance of Meta-heuristic Load balancing Algorithms and the proposed hybrid Algorithms

Method	Running time	Consumed energy
ACO	1563,64	4971668,408
PSO	1509,18	4705558,312
ABC	1502,45	4304232,144
SA	18	4633565,5
Tabu	52	4898627
GA	1566	4395289,5
GA-Tabu	1291	4296607,5
GA-SA	1116	4545965
GA-ABC	1380,93	4154869,48
GA-ACO	1484,01	4430672,06
GA-PSO	1357,50	4245934,84

Conclusion

Our primary objective in this paper was to consider the load balancing in the cloud computing. For this reason, an exceptionally viable way was proposed in recognizing the global optimum solution to difficult function optimization problems comprises on make the Genetic Algorithm (GA) participate with another nearby search algorithm. We have accomplished this idea by building five exceedingly hybrid algorithms, the GA with the Tabu search (TS), the GA with the Simulated Annealing (SA), the GA with the Artificial Bee colony (ABC), the GA with the Particle Swarm Optimization (PSO) and the GA with the Ant Colony Optimization (ACO). The table demonstrates that our new hybrid algorithms give better results in terms of running time (specially for GA-SA) and better results in terms of minimization of energy (specially for GA-ABC).

This new approach comprises generally in presenting a modeling in light of Qos parameters, for example, the energy consumption, robustness, dynamism and the response time, and besides the combination and comparison of five hybrid heuristics with well-known heuristics with a specific end goal to achieve a nearby optimum as far as running time and consumed energy.

Additionally work center around the utilization of other hybridization strategies and the parallelization of the hybrid heuristics utilized in this paper, by abusing the openness of various processors so as to improve the running time.

Acknowledgment

The authors wish to thank the anonymous referee for valuable comments and corrections.

References

[1] Jigna Acharya and chirag patel, " An Artificial Bee Colony based Load balancing in cloud computing", National conference In-novative & Emerging Technologies, june-2016

[2] Nishant, K., et al. "Load Balancing of Nodes in Cloud Using Ant Colony Optimization. In Computer Modelling and Simulation" (UKSim), 2012 UKSim 14th International Conference on. 2012. IEEE.

[3] A. Aditya, U. Chatterjee, S. Gobata, "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 (2015).

[4] Kennedy, J., and R. Eberhart. "Particle Swarm Optimization." Proceedings of the IEEE International Conference on Neural Networks. Perth, Australia, 1995, pp. 1942–1945.

[5] Mezura-Montes, E., and C. A. Coello Coello. "Constraint-handling in nature-inspired numerical optimization: Past, present and future." Swarm and Evolutionary Computation. 2011, pp. 173–194.

[6] Pedersen, M. E. "Good Parameters for Particle Swarm Optimization." Luxembourg: Hvass Laboratories, 2010.

[7] G. Galante, L. C. E. d. Bona. A survey on cloud computing elasticity. in: Proceedings of the IEEE Fifth International Conference on Utility and Cloud Computing, pp. 263–270, 2012.

[8] V. Cerny. "Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm". J Optim Theory Appl, pp. 41–51, 1985.

[9] H.Mehta,P. Kanungo, and M. Chandwani, "Decentralized content aware load balancing algorithm for distributed computing environments", Proceedings of the International Conference Workshop on Emerging Trends in Technology (ICWET), February 2011, pages 370-375.

[10] Dušan Teodorović, 2009 "Bee Colony Optimization (BCO)", Innovations in Swarm Intelligence Studies in Computational Intelligence Volume 248, Pages 39-60.

[11] Hansen,P.1986. "The Steepest Ascent Mildest Descent Heuristic for Combinatorial Programming." Congress on Numerial Methods in Combinatorial Optimization,Capri,Italy.

[12] F. Herrera and M. Lozano. "Adaptation of genetic algorithm parameters based on fuzzy logic controllers." Genetic Algorithms and Soft Computing, pp. 95–125, 1996.

[13] T. Hong, H. Wang, W. Lin, and W. Lee. "Evolution of appropriate crossover and mutation operators in a genetic process." Applied Intelligence, pp. 7–17, January 2002.

[14] S. Kirkpatrick , Jr. CD. Gelatt, MP. Vecchi. "Optimization by simulated annealing." Science, pp. 671–680, 1983

[15] S.Koh, S.Leow,and K.Loke. "An adaptive genetic algorithm for permutation based optimization problems." In Proceedings of the AIAI2005--Second IFIP Conference on Artificial Intelligence Applications and Innovations, 2005.

[16] V. Robles, J. Pena, P. Larraaga, and V. Herves. "A new hybrid cooperative search evolutionary algorithm. Towards a New Evolutionary Computation." Advances in the Estimation of Distribution Algorithms. Series: Studies in Fuzziness and Soft Computing, pp. 187–219, 2004.

[17] X. Shi, Y. Liang, H. Lee, C. Lu, and L. Wang. "An improved ga and a novel pso-ga-based hybrid algorithm." Information Processing Letters, pp. 255–261, 2005.

[18] T. Jones and S. Forrest. "Fitness distance correlation as a measure of problem difficulty for genetic algorithms." Proceedings of the Sixth International Conference on Genetic Algorithms, pp.184–192,1995

[19] T. Schnier and X. Yao. "Using multiple representations in evolutionary algorithms." In Proceedings of the 2000 Congress on Evolutionary Computation, pp. 479–486,2000.

[20] E.Takashima, Y.Murata, N. Shibata, and M. Ito. "Self adaptive island ga." In Proceedings of the 2003 Congress on Evolutionary Computation 03, pp. 1072-1079. 2003.

[21] L. Tseng and S. Liang. "A hybrid metaheuristic for the quadratic assignment problem." Computational Optimization and Applications, pp. 85–113, 2006.

[22] A.LaTorre,V.Robles, and S. Muelas. "Using multiple offspring sampling to guide genetic algorithms to solve permutation problems." Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, pp. 1119–1120, 2008

[23] Rodrigo N Calheiros, Rajiv Ranjan, Anton Beloglazov, Cesar AF De Rose et Rajkumar Buyya. "CloudSim : a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms". Dans : Software : Practice and Experience 41.1 (2011), p.23-50 .

[24] L. Badger, T. Grance, R. Patt-Corner, J. Voas. Draft cloud computing synopsis and recommendations. NIST special publication, 2011.

[25] Fred Glover. "Future Paths for Integer Programming and Links to Artificial Intelligence". Computers and Operations Research. pp. 533–549. 1986.

Hamza Rahhali received a M.Sc. in networks and telecommunications from Aix Marseille University, Marseille, France, in 2011. He is currently a computer teacher in HEM Business School, Casablanca, Morocco. His research interests include load balancing algorithms and Scheduling algorithms in cloud computing

Mostafa Hanoune He is currently a Professor in the Department of Computer Science and Maths, University Hassan II of Casablanca, Morocco. His research interests are in the areas of computer sciences, data mining, Information Technology and Cloud computing