A Genetic-inspired Negotiation Algorithm for QoS and Energy Consumption Tradeoffs in Virtualized Service Centers

Georgiana Copil, Tudor Cioara, Ionut Anghel, Ioan Salomie, Daniel Moldovan, Diana Borza
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
{georgiana.copil, tudor.cioara, ionut.anghel, ioan.salomie, daniel.moldovan, diana.borza}@cs.utcluj.ro

Abstract—**This paper proposes a genetic inspired algorithm for negotiating the tradeoffs between the workload Quality of Service requests and the service center computing resources energy consumption with the goal of allocating the service center computing resources in an energy efficient manner. The bilateral negotiation algorithm has two main parties: the workload task's Quality of Service request, as a client, and the service center servers available computing resources, as a provider. Both the provider and the client are represented by agents and their offers/requests are modeled as chromosomes. A chromosome gene represents the value of the computing resources subject of negotiation. The genetic inspired negotiation process has an initial phase and a bargaining phase. In the initial phase, an initial chromosome population is generated for both the provider and the client and the values of their associated goal chromosomes are set. In the bargaining phase, the client and provider chromosomal populations are evolved using a cognitive process similar to the genetic evolution. An agreement is reached when the distance between one of the received offer/request chromosomes and a corresponding goal chromosome is below a predefined threshold.**

Keywords—negotiation algorithm; genetic inspired; QoS vs energy consumption tradeoff, alternating-offer, bargaining.

I. Introduction and Related Work

The energy used by the service centers has increased drastically in the recent years, being directly related with the number of hosted servers and their workload. The IT analysts report that for every $1.00 spent on new server equipment, another $0.50 is spent on power and cooling expenses [1]. Since computing demand and energy costs are continuously growing, the energy efficiency of IT systems and service centers have become a high priority. One of the main factors contributing to the service center high energy consumption is the low utilization of computing resources.

In this paper we approach the problem of service center computing resources under utilization by proposing a genetic inspired algorithm for negotiating the tradeoffs between the workload QoS (Quality of Service) requests and the service center computing resources energy consumption. The proposed algorithm offers a solution for provisioning the service center computing resources in an energy aware manner this way increasing the service center resources utilization.

The computing resource provisioning is a critical service center management task. It involves the appropriate allocation of computing resources (processor, memory, disk, and network bandwidth) for satisfying workload applications QoS performance goals. A common practice in nowadays service centers is to over-provision the computing resources to handle the infrequent peak workload values. Over-provisioning has caused several implications: increased service center size, servers running at 15-20% of their capacity, increased energy consumption and more necessary cooling resources.

The problem of optimally allocating the computational resources while considering different constraints like the energy efficiency or optimal QoS levels has been proven to be a NP-hard problem [2]. Virtualization is seen as a solution for providing the required isolation layer to consolidate applications running on a large number of low utilization servers to a smaller number of highly used servers [3]. In [4] the problem of power-aware application placement is investigated and an application placement controller called pMapper is presented. The controller dynamically places applications to minimize power consumption while meeting their performance requests. Enterprise applications typically employ a multi-tier architecture where distinct components of a single application are placed on separate servers, and the amount of resources needed to achieve their QoS goals might be different at each tier and may also depend on availability of resources in other tiers [5]. An adaptive resource control system that dynamically adjusts the resource shares to individual tiers in order to meet application-level QoS goals is needed [6]. Although there is a state of the art problem [6], concerns about application performance, infrequent but inevitable workload peaks [7], and security requirements [8] persuade the provisioning decision logic to adopt a conservative approach, such as hardware isolation among applications with minimum resources sharing.

Taking into account the above presented disadvantages we propose a dynamic resource provisioning solution based on negotiating QoS and energy tradeoffs. The QoS and energy tradeoff technique exploits the energy saving opportunities by lowering the performance request levels of the service center servers running workload tasks. The proposed provisioning solution uses a bilateral negotiation process between the workload tasks' QoS request, as a client, and the service center servers available computing resources, as a provider. The negotiation protocol is based on genetic algorithms. Both the provider and the client are represented by agents and their requests/offers are modeled as chromosomes, the chromosome genes giving the value of the computing resources subject of negotiation.

The state of the art literature provides several types of negotiation techniques, categorized by number of players, number of issues under negotiation or constraints existence. Fatima et al. [9] propose a heuristic-based alternating offers protocol for agent-based negotiation. Approaches for multi-issue negotiation are given for the following types of negotiation [10]: sequential, package-deal and simultaneous negotiation. They are compared in terms of necessary time for reaching an agreement, necessary time for computing equilibrium and the existence of a Pareto optimal equilibrium [11], [9]. Lai et al. [12] propose a multi-attribute negotiation protocol, in which self-interested agents need to reach "win-win" agreements. It is a heuristic-based alternating-offer protocol, in which agents model possible offers as an indifference curve which depends on the offer made by the opponent previously. In [13] the authors propose a complex multi-issue negotiation process which uses utility graphs. An agent-based strategy for complex bilateral negotiations over many issues with inter-dependent valuations, leading to Pareto-efficient outcomes, is described. Using this protocol, a relatively small number of steps are required for reaching a negotiation agreement due to the use of prior information about the utility space structure. A negotiation process based on genetic algorithms is the purpose of [14]. The subsystems representing the negotiating client and provider negotiate with each other with the help of a mediator component. The genetic algorithm enables the system to search for the best mutually efficient solution. Each negotiation offer is represented as a chromosome, in order to enable applying genetic operators like mutation and crossover. The chromosome representation comprises a threshold for accepting an offer as the first cell, while the remaining cells represent the options associated with each issue. The fitness expression includes the joint payoff, thus leading a cooperation game.

Analyzing the above presented negotiation solutions we reached the conclusion that genetic based, non-mediated, multi-issue, time constrained negotiation is the best suited solution for negotiating the QoS and energy tradeoffs. As opposed to the agent based negotiation approaches listed above, in our solution the agent goals are updated depending on the opponent's offer/request, leading to a faster convergence while copying real world negotiation model.

The rest of the paper is organized as follows: Section II presents the genetic inspired negotiation algorithm, Section III shows a test-case scenario for the QoS vs. energy efficiency negotiation solution while Section IV concludes the paper.

II. The Genetic Inspired Negotiation Algorithm

The negotiation process achieves a tradeoff between the workload QoS and the energy consumed by the service center with the goal of allocating computing resources for the incoming workload in an energy efficient manner. By workload QoS we refer to the workload tasks request for specific service center computing resource which has a direct impact on different factors like the task execution time or task execution costs. The negotiation process takes place between two parties represented in our algorithm by two intelligent agents: the service center incoming workload, also called client, and the service center servers, also called a provider. We consider that the service center workload is composed of virtualized tasks annotated with QoS requests for service center computing resources. Virtualization is used as an abstraction level, because it allows us to manage the server's running tasks uniformly, without worrying about application dependencies and low-level details.

The subject of the negotiation process (the negotiation issue) is the client's request for service center computing resources and also the provider's offer of computing resources. Both the request and the offer are defined in our genetic algorithm as chromosomes. A chromosome is formed by a set of genes. In our algorithm a gene represents a service center computing resource under negotiation. As a consequence, the chromosomes are modeled as a vector of computing resources, each vector element representing the value of a specific resource. Each chromosome considered computing resource represents a negotiation issue and determines the dimension of the solution search space. In the following we have considered that the most hungry energy computing resources in a service center are the processor (CPU), the memory (MEM) and the disk (HDD) (see Fig. 1).

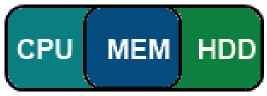

Figure 1. Chromosome representation

The request and offer chromosomes are represented using vectors of three elements as in relations 1 and 2 (the algorithm can be easily extrapolated to any value of n computing resources, with $n > 0$).

$$CHR_{request} = \{CPU_{req}, MEM_{req}, HDD_{req}\} \quad (1)$$

$$CHR_{offer} = \{CPU_{available}, MEM_{available}, HDD_{available}\} \quad (2)$$

For both the client and provider a goal chromosome is defined as their ideal solution in negotiation. The client goal chromosome represents the maximum amount of computing resources that needs to be allocated to a workload virtual task so that it is executed with a maximum performance:

$$CHR_{request-goal} = \{CPU_{max-req}, MEM_{max-req}, HDD_{max-req}\} \quad (3)$$

The provider goal chromosome represents the minimum amount of computing resources for accommodating the workload virtual task so that it is executed with a minimum energy consumption:

$$CHR_{offer-goal} = \{CPU_{min-offer}, MEM_{min-offer}, HDD_{min-offer}\} \quad (4)$$

The genetic inspired negotiation process has two main phases: (i) *an initial phase*, in which an initial chromosome population is generated for both the provider and the client and the values of their associated goal chromosomes are set and (ii) *a bargaining phase*, in which both the client and provider attached chromosomal populations are evolved using a cognitive process similar to the genetic evolution.

While computing a new counteroffer all the chromosomes of the client's population will converge towards the client's goal chromosome and all the chromosomes of the provider's population will converge towards the provider's goal chromosome.

A. The Negotiation Initial Phase

In this phase both the provider and client initial chromosomal populations are generated by creating chromosomes with gene values derived (Δ_{CPU}, Δ_{MEM} and Δ_{HDD} in relation 5 and 6 represent the derivation factors) from their goal chromosome values (see Fig. 2).

$$InitP_{request} = \{ \cup CHR_{request} \mid$$
$$CHR_{request}.CPU \in (CPU_{max-req} - \Delta_{CPU}, CPU_{max-req})$$
$$CHR_{request}.MEM \in (MEM_{max-req} - \Delta_{MEM}, MEM_{max-req})$$
$$CHR_{request}.HDD \in (HDD_{max-req} - \Delta_{HDD}, HDD_{max-req})\} \quad (5)$$

$$InitP_{offer} = \{ \cup CHR_{offer} \mid$$
$$CHR_{offer}.CPU \in (CPU_{min-offer} - \Delta_{CPU}, CPU_{min-offer})$$
$$CHR_{offer}.MEM \in (MEM_{min-offer} - \Delta_{MEM}, MEM_{min-offer})$$
$$CHR_{offer}.HDD \in (HDD_{min-offer} - \Delta_{HDD}, HDD_{min-offer})\} \quad (6)$$

The aim is to create the initial requests ($InitP_{request}$) and offers ($InitP_{offer}$) as close as possible to the parties goals and to force counteroffers as close as possible to those goals.

B. The Bargaining Phase

The bargaining phase which takes place between the client and provider has two main steps: (i) the request and offer exchange and (ii) the chromosomal population evolution.

In the *bargaining phase first step* the provider and the client agents will alternatively exchange requests and offers until an agreement is reached (see Fig. 2). At the beginning, the client chooses from its chromosomal population the goal chromosome (the maximum amount of computing resources that need to be allocated) and sends it to the provider agent as the initial request. After the provider agent receives the initial request, its chromosomal population is evolved using a genetic evolution algorithm (see bargaining phase second step). From the provider evolved population the best chromosome individuals are selected (the closest to the goal chromosome) and sent to the client as counteroffers. At the client side the same evolution process is used to generate a new adjusted request chromosome by taking into account the received offers.

An agreement can be reached both at the client and provider sides when one of the received chromosomes representing an offer or a request is close enough to the population goal. To evaluate the degree of closeness between two chromosomes, they are represented in a three-dimensional space (see Fig. 3) having as axes the chromosome's genes (computing resources) and the Euclidian distance is employed:

$$Dist(CHR_x, CHR_y) = \sqrt[2]{\begin{array}{l}(CHR_x.CPU - CHR_y.CPU)^2 + \\ (CHR_x.MEM - CHR_y.MEM)^2 + \\ (CHR_x.HDD - CHR_y.HDD)^2\end{array}} \quad (7)$$

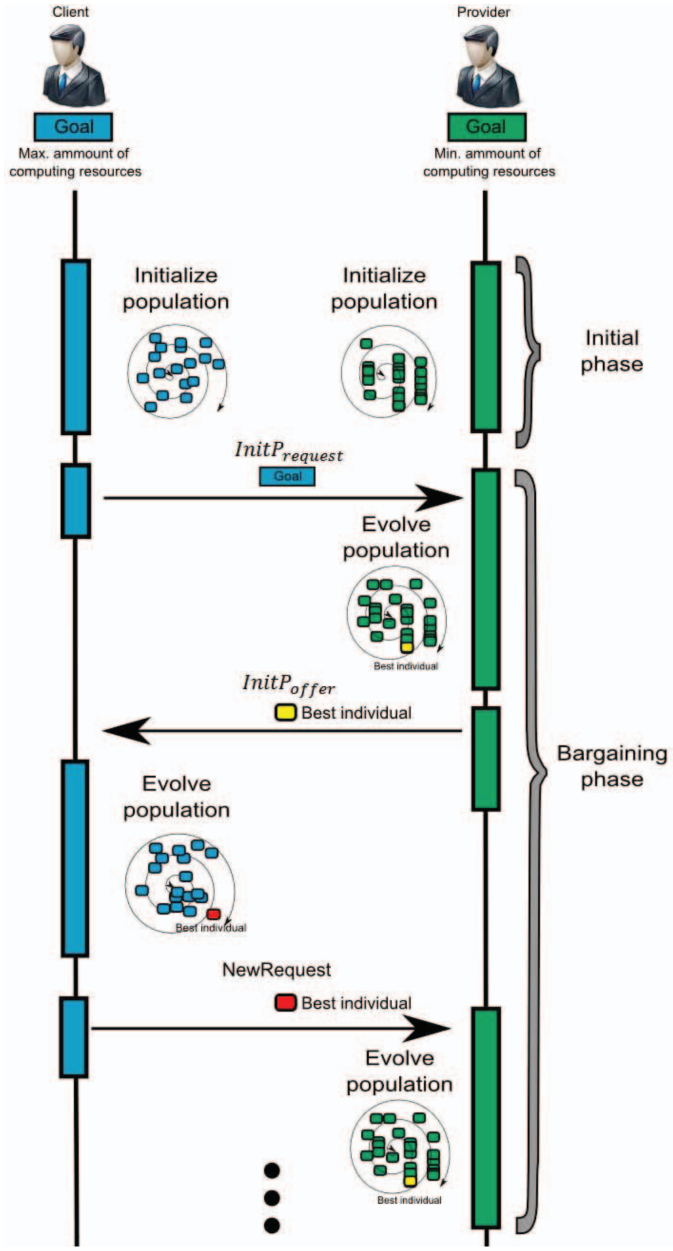

Figure 2. The negociasion process intial and bargaining phases

If the distance between one of the received chromosomes and the corresponding goal chromosome is below a predefined threshold (T) an agreement is reached (terminationCondition in Fig. 4 algorithm). Otherwise the bargaining process is continued. The threshold is heuristically determined and depends on the bargaining process elapsed time (inversely proportional to the time spent negotiating) and the maximum time allocated for negotiating (a cut off constant value).

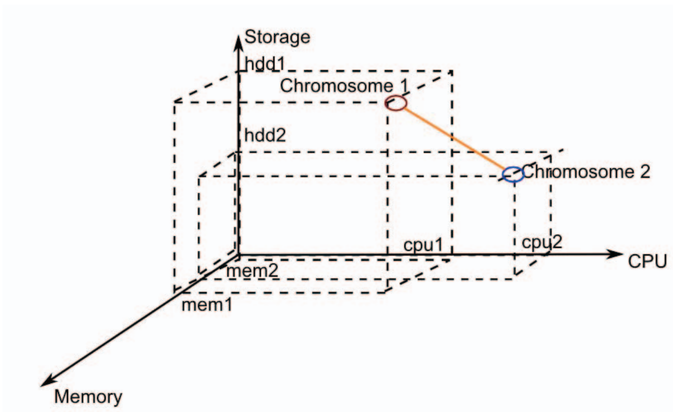

Figure 3. Chromosomal decision 3D space example

In the *bargaining phase second step* the chromosome population evolution process takes place both at the provider and client sides following the genetic evolution algorithms with few adjustments. A new offspring evolved chromosomal individual is generated from two chromosomes parents. One parent chromosome is selected from the set of best individuals of the current population and represents chromosomes closest to the goal (SELECT-BEST-FIT-INDIVIDUALS procedure in Fig. 4 algorithm). The other parent is selected from the chromosomes received as request or offer. To assure the increase of the population diversity, the new chromosomes are generated using all the following methods: heuristic crossover, uniform crossovers and mutation (see Fig. 4 algorithm).

Algorithm 1. Population_Evolution_Client

Input: P_{offer}= $\{\cup CHR_{offer}\}$ – supplier chromosomal population
Output: evolved chromosomal population
begin
 P_{req}= GEN-INITIAL-POPULATION()
 while (!terminationCondition) **do**
  bestIndReq = SELECT-BEST-FIT-INDIVIDUALS(P_{req})
  bestIndOffer = SELECT-BEST-FIT-INDIVIDUALS(P_{offer})
  HEURISTIC-CROSSOVER(bestIndReq, bestIndOffer)
  UNIFORM-CROSSOVER(bestIndReq, bestIndOffer)
  MUTATION(P_{req})
  UPDATE-POPULATION()
 end while
end

Figure 4. Client agent's population evolution algorithm

When using *heuristic crossover* two different evolved chromosomes are generated (see Fig. 5 for an example): (i) an identical replica of the best parent and (ii) a best parent derived chromosome with a random chromosomal value:

$$CHR_{offspring} = CHR_{bestparent} + CHR_{random} \quad (8)$$

In relation 8 the random chromosomal values are determined using the two parents chromosomes and a random factor α as:

$$CHR_{random} = \alpha(CHR_{bestparent} - CHR_{worstparent}) \quad (9)$$

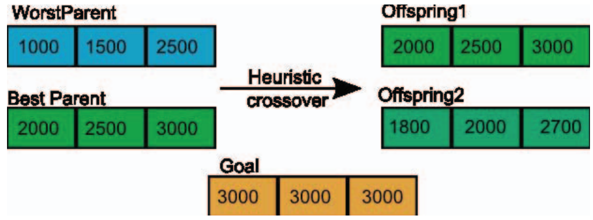

Figure 5. New chromosome generation using heuristic crossover

Using *uniform crossover* the new evolved chromosomes are generated by exchanging the genetic content of the two parents chromosomes at randomly chosen position (see Fig. 6 for an example).

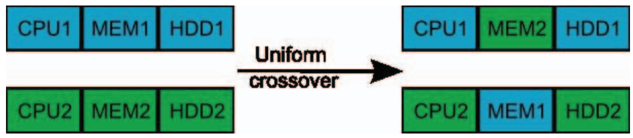

Figure 6. New chromosome generation using uniform crossover

For *mutations* new random values are added to chromosome parent genes.

The new offspring chromosomes are evaluated using as fitness function the Euclidean distance to the goal chromosome. For fastening the approach towards an agreement, the goal is updated depending on the received offer or request:

$$CHR^{t}_{offer-goal} = CHR^{t}_{offer-goal} + \delta * CHR^{t}_{request} \quad (10)$$
$$CHR^{t}_{request-goal} = CHR^{t}_{request-goal} - \delta * CHR^{t}_{offer} \quad (11)$$

In relations 10 and 11 δ is a compromise factor characteristic to each party agent denoting how much the agent is willing to sacrifice for the sake of reaching an agreement. This approach follows the real life situations, where a person's goal diminishes as continuing with the negotiation.

III. Case Study and Results

In this chapter we have tested the above presented negotiation algorithm ability to tradeoff between the workload request for computing resources and the service center energy consumption. For testing purposes we have simulated a service center with five servers, each server having the same hardware configuration: CPU - Intel(R) i7 870 2.93 Ghz, MEM - 6GB DDR3 and HDD - 750GB. The workload that the simulated service center needs to accommodate is generated randomly and it consists of sequential virtual tasks. Each virtual task is described by its request for service center server hardware resources expressed as values of CPU, MEM and HDD.

Being a multi-issue negotiation, each negotiation party must have attached admissible intervals for each of the issue (computing resource) under consideration. To ease the testing process we have considered that all the test case workload virtual tasks have the same admissible intervals for the requested computing resources to be allocated (CPU between 500Mhz and 1000Mhz, MEM between 300 Mb and 1000 Mb and HDD between 20 and 50 Gb). The service center servers admissible intervals of computing resources to be allocated are given by the amount of server available computing resources (see Table 1 for the considered test case situation).

TABLE I. SERVICE CENTER AVAILABLE COMPUTING RESOURCES

	CPU	MEM	HDD
Server 1	Offers: 280 Mhz ↗	Offers: 410 MB ↗	Offers: 4 GB ↗
Server 2	Offers: 320 Mhz ↗	Offers: 370 MB ↗	Offers: 9 GB ↗
Server 3	Offers: 240 Mhz ↗	Offers: 300 MB ↗	Offers: 1 GB ↗
Server 4	Offers: 210 Mhz ↗	Offers: 310 MB ↗	Offers: 5 GB ↗
Server 5	Offers: 230 Mhz ↗	Offers: 400 MB ↗	Offers: 5 GB ↗

The Client Agent (associated with the workload tasks requiring computing resources) and the Provider Agent (associated with the service center servers offering resources) are implemented as JADE (Java Agent Development Framework) [15] agents. Their behavior implements the bargaining process of exchanging the requests and offers. The Client Agent starts the negotiation process by issuing a request which in the first step is its goal, and waits for a response from the Provider Agent with a counteroffer. Both agents implement and execute the evolution algorithm population diversification processes and are able to choose the best fit solution for itself, this way increasing the algorithm convergence.

The negotiation process evolution together with the client request‘s and provider‘s offer convergence towards an agreement for the above presented scenario is shown in Fig. 7. We have considered as compromise factor $\delta = 0.05$ for the Client Agent and $\delta = 0.1$ for the Provider Agent. It can be noticed that an agreement is reached after approximately 10 turns of alternating requests and offers.

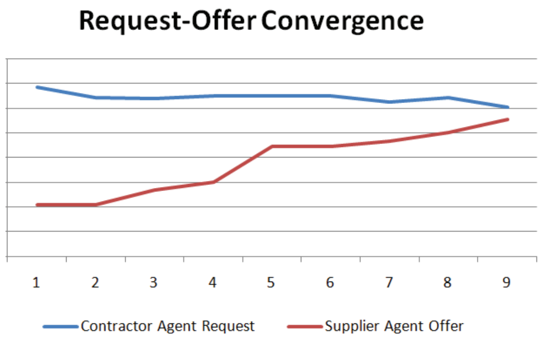

Figure 7. Client request and provider offer convergence

Fig. 8-10 present the evolution of both client request and provider offer during the negotiation process for accommodating the test case workload task in an energy efficient manner.

For HDD, the Client Agent associated to the workload virtual tasks issue a request for 50 Gb and at the end of negotiation it accepts an offer of around 44 Gb decreasing its request (see Fig. 8). The provider associated to the service center servers starts by offering between 1 Gb (for minimum energy consumption) and ends by allocating 44 Gb.

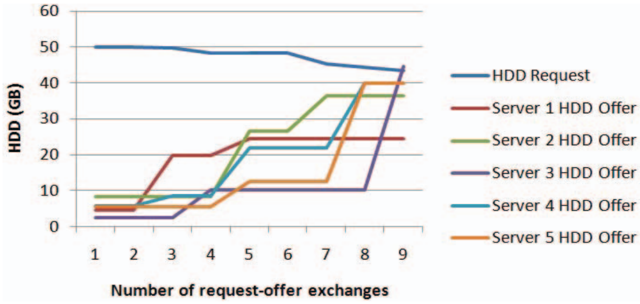

Figure 8. Negotiating the provisioning of HDD computing resource

The same type of compromises from both agents can be observed when analyzing the negotiation process results for CPU (Fig. 9) and MEM (Fig. 10).

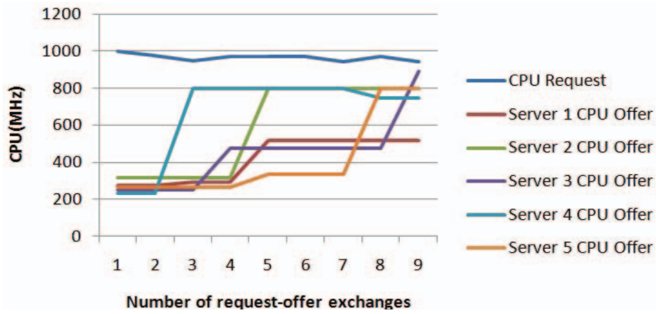

Figure 9. Negotiating the provisioning of CPU computing resource

In the above figures it can be seen that the Provider Agent converges faster towards the desired solution of the Client Agent due to the fact that its compromise value (the δ parameter) is greater than the Client Agent compromise value. The value of the compromise parameter is used to establish the direction of the QoS and energy consumption tradeoff negotiation. If the Client Agent compromise value is greater than the Provider Agent, the emphasis is on virtual tasks performance. Otherwise the emphasis is on reducing the energy consumption.

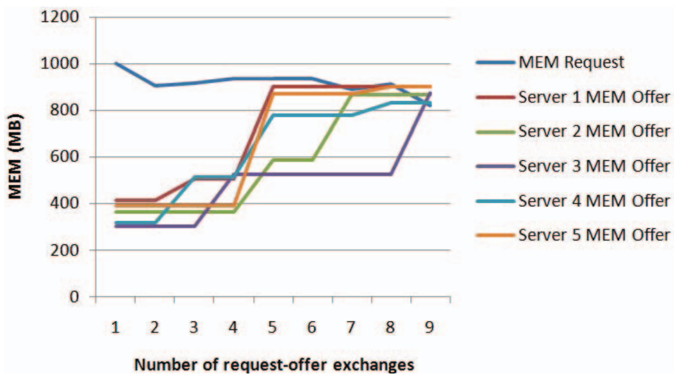

Figure 10. Negotiating the provisioning of MEM computing resource

IV. CONCLUSIONS

In this paper the problem of under utilization of service center computing resources is approached by proposing a genetic inspired algorithm for negotiating the tradeoffs between the workload QoS requests for computing resources and the service center energy consumption. The algorithm is based on genetic concepts like chromosomes, genes and uses methods inspired from the genetic evolution process. A negotiation compromise factor is defined and used to determine the negotiation emphasis: on performance or energy consumption optimization. The test case results show that using our negotiation process the client request and provider offer convergence and an agreement is obtained in a reasonable time frame. This is due to the population diversification processes implemented by our negotiation solution which enables each party to choose the best fit solution for itself by continuously updating their goals.

ACKNOWLEDGMENT

This work has been supported by the European Commission within the GAMES project [16] funded under EU FP7.

REFERENCES

[1] J. Scaramella, M. Eastwood, “Solutions for the Datacenter’s Thermal Challenges”, White Paper, 2007.

[2] Y. Ajiro and A. Tanaka, “Improving Packing Algorithms for Server Consolidation”, Proceedings of the Computer Measurement Group's, 2007.

[3] C. Kesselman, “Grid Resource Abstraction, Virtualization, and Provisioning for Time-Targeted Applications”, 8th IEEE International Symposium on Grid, 2008.

[4] A. Verma, P. Ahuja, A. Neogi, “pMapper: Power and Migration Cost Aware Application Placement in Virtualized Systems”, Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, 2008.

[5] P. Padala, K. Shin, A. Arbor, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, P. Alto, K. Salem, “Adaptive Control of Virtualized Resources in Utility Computing Environments”, Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems, 2007.

[6] D. Jiang, G. Pierre, C. Chi, “Autonomous resource provisioning for multi-service web applications”, Proceedings of the 19th international conference on World wide web, 471-480, 2010.

[7] Q. Zhu, G. Agrawal, “Resource provisioning with budget constraints for adaptive applications in cloud environments”, Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, 2010.

[8] Y. Demchenko, A. Wan, M. Cristea, C. de Laat, “Authorisation infrastructure for on-demand network resource provisioning”, Proceedings of the 2008 9th IEEE/ACM International Conference on Grid Computing, 2008.

[9] S. Fatima, M. Wooldridge, Nicholas Jennings, "Multi-Issue Negotiation with Deadlines", Journal of Artificial Intelligence Research, Volume 27, pages 381-417, 2006

[10] H. Raiffa, “The art and science of negotiation”, Cambridge, Massachusetts: Harvard University Press, 1982

[11] T. Kaihara, S. Fujii, N. Yoshimura, "A study on agent behaviour conducting Pareto optimality in virtual market", Proceedings. 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation, 1375 – 1380, vol.3, 2003.

[12] G. Lai, K. Sycara, C. Li, “A Decentralized Model for Multi-Issue Negotiation”, Proceedings of the 8th international conference on Electronic commerce: The new e-commerce: innovations for conquering current barriers, obstacles and limitations to conducting successful business on the internet, pages 3 - 10 ,2006

[13] V. Robu, K. Somefun, J. La Poutre, “Modeling Complex Multi-Issue Negotiations Using Utility Graphs”, Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, pages 280 - 287, 2005

[14] R. Dzeng, Y. Lin, “Searching for Better Negotiation Agreement Based on Genetic Algorithm”, Proceedings of Computer-Aided Civil and Infrastructure Engineering, 2005

[15] F. Bellifemine, G. Caire, T. Trucco, and G. Rimassa, “Jade programmers guide”, free Library, 2007.

[16] GAMES FP7 Research Project, http://www.green-datacenters.eu/