A Genetic Model for Pricing in Cloud Computing Markets

Mario Macías
Barcelona Supercomputing Center/Technical University of Catalonia
Jordi Girona s/n
Barcelona, Spain
mario.macias@bsc.es

Jordi Guitart
Barcelona Supercomputing Center/Technical University of Catalonia
Jordi Girona s/n
Barcelona, Spain
jguitart@ac.upc.edu

ABSTRACT

Cloud Computing markets arise as an efficient way to allocate resources for the execution of tasks and services within a set of geographically dispersed providers from different organisations. Client applications and service providers meet in a market and negotiate for the sales of services by means of the signature of a Service Level Agreement. Depending on the status of the demand, the provider is able to offer higher or lower prices for maximising its profit. It is difficult to establish a profitable pricing function in competitive markets, because there are several factors that can influence in the prices. This paper deals with the problem of offering competitive prices in the negotiation of services in Cloud Computing markets. A Genetic Algorithms approach is proposed, in which a naive pricing function evolves to a pricing function that offers suitable prices in function of the system status. Its results are compared with other pricing strategies, demonstrating its validity.

Keywords

market-based cloud computing, genetic algorithms, pricing

1. INTRODUCTION

At recent years, the big mainframes paradigm in which users own their computing resources [1] is being progressively transiting to an utility-driven paradigm, in which users do not own resources and pay for the usage of remote resources [17]. Cloud Computing [5] is the most promising current implementation of Utility Computing in the business world, because it provides some key features over classic utility computing, such as elasticity to allow clients dynamically scale-up and scale-down the resources in execution time, or the possibility of customizing completely the software environment by acquiring administrator rights without putting in risk the whole system.

Since Clouds are heterogeneous, elastic and scalable, large systems are too complex to be managed centrally. Market-based resource management is proposed to deal with the complexity because the possibility of doing business will motivate Service Providers to offer their resources in the system and give a Quality of Service (QoS) according to their real capacity. In addition, market mechanisms obligate the users to adjust their reservations to their real space and time requirements. Another advantage is that it is relatively easy to implement in a decentralised architecture, whose participants enter in the Market looking for the satisfaction of their own necessities, and they do not need to know about the global status of the system to maximise their utility.

Brokers that represent Service Providers or Clients enter in a Cloud Computing market for selling or buying services or resources. When a Client finds its requirements in the market, a negotiation process is started to establish the terms of the contract. If both parties reach an agreement, the terms of the contract are specified in a Service Level Agreement (SLA) and the Client can use the resource. During the usage of the resources, the correct fulfilment of the terms of the SLA is watched by a neutral entity, and penalises the buyers or the sellers when they violate the SLA. Negotiating Brokers must be provided with business models and intelligent behavior, so they are able to take the best decisions for maximising the utility of Clients or Providers in the market.

From the wide bunch of economic knowledge and behavior of Market Brokers, this paper concretely deals with finding the more suitable offer prices in each market status: Cloud providers want to sell their services at high prices for maximising their benefit; however, clients have possibility of election, and will choose the cheapest provider for the same QoS. The freedom of election of the client depends on the status of the demand [12]: providers can raise their prices when the demand is high, and they must decrease prices when the demand is lower than the offer. The actual price that the client pays for the service is named **Exercise Price**.

Previous work from the authors demonstrated that providers can acquire high benefits by pricing their services in function of the demand [16]. This work assumes that markets are stable and always behave rationally, according to some pre-defined models. These assumptions can lead providers to underperform economically in some special scenarios, such as very low or very high offer/demand ratios. The proposed model considers some parameters such as demand, work load of the resources, or predictions about future load. However, there are some other parameters that

SAC'11 March 21-25, 2011, TaiChung, Taiwan.

can influence the prices, which can be difficult or impossible to include in the models because of their random nature.

For dealing with this uncertainty problem, this paper proposes Genetic Algorithms [14] as a model for analysing financial markets [7]. The basic idea of Genetic Algorithms is to have an extensive population of generic pricing models (chromosomes) whose parameters are stored as genes. At the initial moment, the genes are random, and some chromosomes are better than others (this is, their pricing models provide prices that are more beneficial for providers). The best chromosomes are selected in base to their proposed prices, and they are reproduced and mutated by simulating the natural evolution process. After some iterations of this process, the population of chromosomes will tend to provide prices that maximise the benefit of the provider. As in nature, if the environment changes, the population will self-evolve to become well adapted.

This paper is a step forward in the definition of pricing strategies of Cloud Providers. Genetic Algorithms are used because they are simple to implement, and enough dynamic for modifying themselves (in comparison to previous models from the authors, whose pricing results were dynamic, but the models were static). This dynamicity will allow the model to self-adapt to changes in the market, and keep providers offering beneficial prices in the long term. This paper proposes a new Genetic Pricing Model that considers the relative simplicity (compared to real financial markets) of Cloud Computing Markets and evaluates it experimentally, by comparing it with pricing models used in previous works from the authors.

The rest of the paper is organised as follows: after the related work of Section 2, Section 3 describes in detail the model and algorithms used in the experiments. Section 4 starts with a concrete description of the experimentation framework and the values used for the parameters of the model, and ends with a visual description of the obtained results, compared with several pricing models. At the end, the conclusions of the work and the future trends in the related research are enumerated.

2. RELATED WORK

Computer models have been demonstrated more efficient than humans when making decisions [13] in many market scenarios, especially when a high volume of data must be considered.

Previous work from the authors [16] introduces policies for pricing in Cloud Computing markets. We demonstrated that providers that adapt their prices in function of the competence, time slot and SLA terms can achieve better Business Objectives, such as Revenue Maximisation. However, the proposed model is still too rigid, and assumes that other participants in the market always behave rationally. In addition to the flexibility in pricing, this paper also adds flexibility to the pricing functions, by allowing their self-modification for a better adaptation to changing market environments.

Genetic Algorithms are a widely used tool for the analysis of financial markets due to their simplicity and capacity of adaptation to chaotic environments [6, 7]. However, the major usage of Genetic Algorithms is the forecasting. Forecasting is a valuable tool for the sales of futures in Cloud Computing, for example batch jobs whose sales can be negotiated some days before their execution. However, values of future predictions are not so useful when selling Web Services, whose negotiation and execution must be performed in real time according to some existing Utility Computing markets [3]. This paper intends to adapt pricing models to Web Services sales in Cloud Computing.

Cliff [9] explored a continuous space of auction mechanisms via Genetic Algorithms, with artificial trading agents operating in evolved markets. His work does not rely on the modelling of market agents but in the market itself, by using Genetic Algorithms to tune market dynamics. It is important to emphasize that agents and market are more stable against market shocks, by evolving to suitable behaviors.

Fayek et al. [11] propose the usage of Genetic Algorithms for evaluating the validity of a set of offers by calculating their utility. Its application of Genetic Algorithms when modelling the behavior of agents is worth considering. This paper intends to be a step forward, by adding pricing models to the behavior of the agents.

Chidambaran et al. [8] study the effectiveness of Genetic Algorithms in Option Pricings, but our scenario is not strictly an Option Sales Scenario [10]. Their solution is based in the Black-Scholes algorithm [2], in which the input is not necessarily available in Cloud Computing markets (e.g. stock prices, risk-free rates, volatility, etc). The Genetic Model proposed in this paper works with any available set of parameters that could influence in the price of Service.

3. APPLYING GENETIC MODELS TO PRICING

Finding a good pricing model through Genetic Algorithms implies solving the next three issues:

Define a chromosome. In this paper, the *chromosome* is a naive function, whose parameters are some relevant data that could influence in the price, as described in Section 4.1. The relations and weights of these parameters are determined by the *genes* of the chromosome, which are at least partially different between the chromosomes. This function is called **pricing function**, because its evaluation corresponds with the price that a provider will ask for the sale of a Cloud service. The result of the pricing function is named **output** of the chromosome.

Evaluating the chromosomes. The chromosomes in a population must be evaluated. That means that their output must be compared to a **reference value** that is given by a teaching entity or by the actual value when trying to do predictions. In this paper, the reference value is the Exercise Price.

Selection and reproduction of chromosomes. The chromosomes with lowest results in the evaluation are discarded from the population. Pairs of the best adapted chromosomes are selected to be reproduced by mixing their genomes, so the population is replenished.

The rest of this section describes how the three enumerated issues have been faced up.

3.1 Definition of chromosomes

Let $\overrightarrow{P} = \{p_1, ...p_n\}$ be a set of n parameters that contain some relevant information that **could** influence in the price of a requested task (for example, the amount of demand, the load of the system, the hour of day, the amount of resources, etc.). It must be emphasized that some of these parameters could influence, but actually do not necessarily do. Section 4.1 describes deeply the parameters used in the experiments of this paper.

Let $\overrightarrow{G} = \{g_1, ...g_m\}$ be a set of $m = 2n^2+2n+1$ genes that vary across different chromosomes and indicate the weights and mathematical relations between the parameters. Equation 1 shows the pricing function expressed in each chromosome by $\overrightarrow{P}$ and $\overrightarrow{G}$.

$$Pricing(\overrightarrow{P}, \overrightarrow{G}) = \frac{\sum_{i=0}^{n} g_i \prod_{j=0}^{n} p_j^{g_{i+j+1}}}{\sum_{i=n^2}^{n^2+n} g_{i+n} \prod_{j=0}^{n} p_j^{g_{i+j+1}}} + g_m \qquad (1)$$

Assuming that the optimum pricing function is unknown because it can change as the market evolves, Equation 1 describes a simple and generic function that is able to evolve to specific approximation functions by assigning a proper value set for $\overrightarrow{G}$. For example, Equation 1 can be transformed into a linear function such as ${p_0}^3 + 3p_2 + 6$, a division of linear functions such as $\frac{{p_1}^2+0.5p_4}{{p_0}^{1/3}+3{p_2}^{-1}} + 0.3$, or other types of nonlinear functions such as $(p_0p_1)^4 + 2{p_2}^6 + 4p_2p_3 + 3.2$

3.2 Evaluation of chromosomes

The reference value ($RefVal$) is the lowest price that the buyer has chosen to pay in the last market competition, after the sale is performed. That evaluation requires of the existence of a Market Information System [4] that makes visible some pricing information to the market participants.

The scoring of a chromosome at time t is $|Pricing_t(\overrightarrow{P}, \overrightarrow{G}) - RefVal_t|$. The closest to 0 is the score the best price has proposed the chromosome at instant t. However, this score is not enough to select or discard chromosomes from a population, since it does not have any temporal perspective: the chromosome that is proposing the best prices during the last negotiations could be discarded by only returning one inexact price at a given moment. To deal with this issue, the score at time t is weighted by a memory factor $M \in [0, 1]$ with the past scores as shown in Equation 2.

$$Score_t = (1-M) \cdot |Pricing_t(\overrightarrow{P}, \overrightarrow{G}) - RefVal_t| + M \cdot Score_{t-1} \qquad (2)$$

The higher the memory rate M is, the higher importance is given to past price offers. The lower M is, the higher importance is given to the last offer.

3.3 Selection and reproduction of chromosomes

After all the chromosomes are evaluated, the population is sorted in function of the score of the chromosomes. A fixed percentage of the last chromosomes in the sorted population is discarded. At last, the missing population is restored with descendants of the most effective chromosomes, which will inherit most characteristics of their parents with small variations due to possible mutations. The chromosomes that will be crossed for having offspring are chosen successively from the most effective to the less effective ones, until the population is restored again.

When two chromosomes are having offspring, a crossover index between 0 and the length of the genome is chosen randomly, and the genomes of the two parents are divided

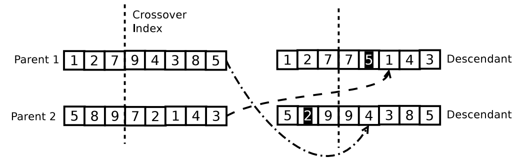

Figure 1: Process of crossing two chromosomes and mix their genome in their offspring. Genes with black background represent random mutations

in this index. The first division of the genome of parent 1 and the last division of the genome of parent 2 are copied in the genome of descendant 1. The first division of the genome of parent 2 and the last division of the genome of parent 1 are copied in the genome of descendant 2 (see Figure 1).

During the process of crossing and copying genomes, some random mutations can occur, with very low probability: a gene is multiplied by a random number with a Normal distribution, whose mean value and standard deviation are 1.

4. EVALUATION OF THE MODEL

Four Cloud providers are competing in a services market whose demands are variable across the day (few demand in the early morning, peaks of demand in the evening). Each of the four providers has a different pricing strategy:

Fixed Pricing. Offered prices are the 5% between the minimum price that the provider can offer in order to not lose money (Reservation Price of the Seller, RP_{seller}) and the maximum price that the client can pay in order to get benefit by buying the service (RP_{buyer}). Instead of 5%, any other percentage could be chosen, but previous work demonstrated that 5% gets good results in most of the demand scenarios [16]. Although the seller knows its own RP, the buyer does not communicate its RP to the provider, so it only can be estimated in function to the historic prices and other market data. Equation 3 shows the used pricing formula.

$$Price_{Fixed} = RP_{seller} + (RP_{buyer} - RP_{seller}) \cdot 0.05 \qquad (3)$$

Random Pricing. Prices are offered randomly, in an uniform distribution, between RP_{buyer} and RP_{seller}. This is not a real pricing model, but it is included in the experiments to be compared with the genetic pricing models and show that they do not behave randomly, as sometimes apparently do.

Utility Maximisation Price. Previous work [16] demonstrated the advantages of using an utility function whose maximisation leads to a beneficial offer price (see Equation 4). $u_p(S)$ is a sub-utility function that tends to 0 when the proposed price is near the Reservation Price of the Seller and tends to 1 when the proposed price is near the Reservation Price of the Buyer. $a(t)$ is the aggressiveness factor, that tends to 0 when the resources are idle and to 1 when the resources are in their maximum workload capacity. Previous work [16] contains a detailed description of Equation 4, and explains its idiosyncrasy.

$$u_{rv}(S) = 0.5 + \frac{\sin\left(\frac{\pi}{2}\left(2u_p(S) + \left(1 - a(t)^{15}\right)\right)\right)}{2} \qquad (4)$$

Genetic Pricing. Applies the genetic pricing algorithm explained in Section 3 with the parameters and constants described in Section 4.1. The offer price is the output of the first chromosome in the list, which is ordered by the calculated scores as explained in Sections 3.2 and 3.3. When deciding the size of the population of chromosomes and the mutations rate, it must be considered the advantages and inconveniences of the choice. Providers with a large number of chromosomes and a small mutations rate are pretty stable, but they are less capable to adapt quickly to changes in the environment. On the other hand, providers with less chromosomes and more mutations converge quicker to a good solution, but they are less stable, and small changes in the environment could make them *bouncing* to bad price offers.

4.1 Simulation environment

The simulation does not simulate a complete market environment, but mainly how providers behave under different demand scenarios. In the simulation, clients send requests for buying cloud resources to a simple market. These requests contains information about the number of resources that the client is willing to use, the Quality of Service and the time slot in which the client will execute the tasks. The market forwards the requests to the providers, which will return their proposed price according to the information contained in the requests. Finally, the client buys the resources of the provider that returns the lowest price.

Each request includes information about the number of CPUs required for the deployment of the task and the range of QoS, which can be Gold, Silver, or Bronze. The provider will make bigger efforts for fulfilling SLAs whose QoS range is Gold. In case of system failure, Bronze SLAs will be violated firstly. If this is not enough, Silver SLAs will be violated then [15]. In compensation, the Reservation Price in Gold tasks is 25% higher than in Silver tasks and 66% than in Bronze tasks.

The frequency of requests is variable: from 2 tasks/hour (off-peak hours) to a maximum (peak hour) that is changed across the multiple simulations. The value of this maximum varies from 2 to 32 tasks per hour. Each task can require randomly from 1 to 4 CPUs, and only providers that have free resources can accept an incoming task and offer a price. Each provider has 16 CPUs.

The set of parameters, chosen by their influence in the final price, is $\overrightarrow{P} = \{Q, C, a(t)\}$, where Q is the QoS category (Bronze = 1, Silver = 2 and Gold = 3), C is the number of CPUs, and $a(t)$ is the aggressiveness factor of Equation 4. The memory rate M (Equation 2) is 0.9. This value has been chosen because it allows chromosomes to ascend in the ordered population, and avoids that a chromosome falls down if it reports only a bad offer price. Some previous tests revealed that M does not have to be exactly 0.9: it also could have similar values such as 0.8 or 0.95. Small values, such as 0.5, make the system too unstable and the provider cannot converge to a good solution.

Regarding the flexibility of the genetic algorithm, two types of genetic providers have been tested: a flexible one, with 200 chromosomes and a mutation rate of 6%, and a rigid provider with 500 chromosomes and a mutation rate of 1%. *Flexible provider* means that it can converge quickly to a good solution, but it is unstable and it *forgets* past experiences. Since each chromosome has 25 genes ($2n^2 + 2n + 1$ when $n = 3$, according to the number of elements of $\overrightarrow{P}$), 200 chromosomes in a same provider is enough diverse and it introduces a small probability of redundancy. The quick change of population is strengthened by setting the mutations rate to 6%: in average, each new chromosome will have 1.5 mutations.

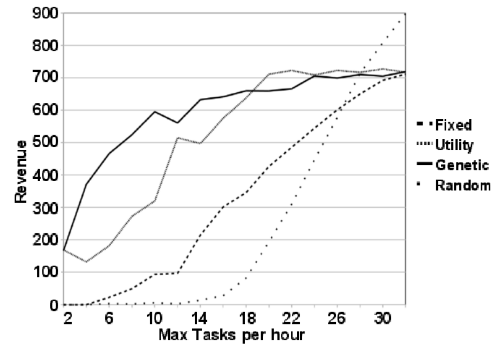

Figure 2: Comparison of revenues between four types of pricing. A provider with a flexible genome (200 chromosomes and 6% of mutations) is used.

The rigid provider increases its number of chromosomes by 150% to add possibility of redundancy and, with a mutation rate of 1%, only a mutation will occur for each 4 descendants. As the experiments show, those values will make the population of chromosomes more stable and uniform, and the provider will converge slowly to offer competitive prices, but it is more stable against noises.

For each chromosome evaluation and selection in both rigid and flexible providers, the lowest 50% of the ordered population is discarded and replenished with the descendants of the other 50% of population. When populations are large enough, this replacement proportion value could be also 40%-60%, 60%-40%, or any other equilibrated rate that guarantees that the best chromosomes during the last iterations are kept.

The chosen constant values of the experiments are not important from a qualitative point of view, because the goal of this research is to observe how variations can affect positively or negatively on results. Because the experimental environment is simulated, the goal is to show how, for example, adding rigidness to the providers leads to more stability in the results, but less capacity of adaptation. Future work will try to find the best constant values for real market environments and evaluate their quantitative data.

Several simulation sets, with same environments but different maximum tasks per hour, have been repeated and the comparisons of revenues in providers have been commented. 5 weeks of sales in a competing market have been simulated, but the first week is not counted for the statistics, because it is considered a prudential training period for the genetic providers.

Results are evaluated in terms of revenue: the client sends its task to the provider that offers the best price, and the provider earns the amount of money that is agreed between the two parts.

4.2 Comparing genetic and utility-based dynamic pricings

Figure 2 shows the revenues of the four providers described

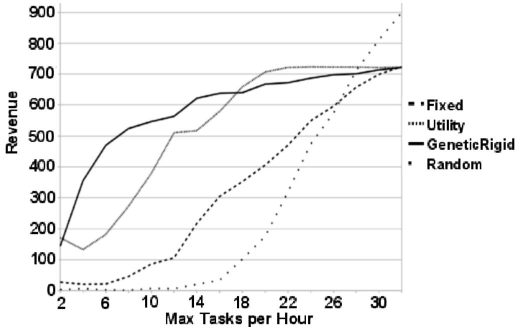

Figure 3: Comparison of revenues between four types of pricing. A provider with a rigid genome (500 chromosomes and 1% of mutations) is used.

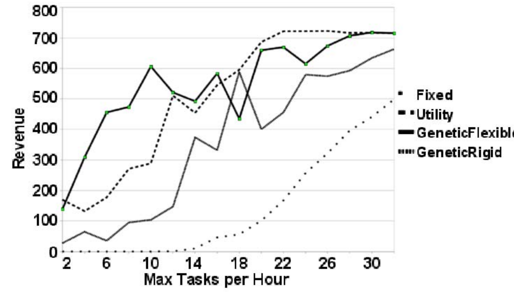

Figure 4: Comparison of revenues when genetic providers with both rigid and flexible genomes are competing.

in Section 4. Random-pricing provider is the most inefficient of all the providers, excepting when the market is extremely overloaded and any price below the Reservation Price of the Buyer is accepted by the clients. The revenue of fixed-pricing provider is increased linearly with the number of maximum tasks per hour: at more tasks with fixed price, the same proportion of revenue. The random nature of genetic algorithms introduces some noises in the results, such as the small perturbation in the revenue of providers when the maximum tasks are 12 per hour.

Although utility-maximisation provider is a good solution compared with fixed pricing, Figure 2 shows that the genetic provider gets the highest revenue in most of the scenarios. When the maximum number of tasks is high, both solutions are similar. Genetic pricing showed its effectiveness mainly in equilibrium markets, which is the status that markets tend to. Both right and left extremes of the graph (respectively demand and offer excess) are unrealistic scenarios.

4.3 Comparing genetic providers by flexibility

Figure 3 shows how rigid genomes do not introduce so much perturbation as flexible genomes, but it does not mean that they are more suitable in terms of revenue maximisation. To check which flexibility grade is more suitable in Cloud computing markets, the same experiment is repeated with a rigid and a flexible genetic provider competing in the same market. Figure 4 shows the results of the experiment, and some relevant information can be extracted from it:

- Two genetic providers add instability to the results. It is because the genetic algorithm proposed in this market **imitates** the best pricing in each moment. Fixed and utility-based pricings are predictable, if the genetic provider takes their pricing attempts as input it will be much more stable than if it takes the output of another genetic (and unpredictable) provider.

- Within this instability scenario, a flexible genetic provider earns more money than the rigid one, since it can converge quicker to best solutions.

To illustrate this last statement, the accuracy of pricing and speed of convergence of both flexible and rigid genetic providers are measured. Figure 5 shows the difference of the offered prices and the Exercise Price, and speed of convergence of both rigid (upper graph) and flexible (lower graph) genetic providers. If the difference is 0, it means that the price offered by the genetic provider is actually the Exercise Price.

Both figures show the influence of noises in the genetic providers, which made them spontaneously evolve to offer prices far from the Exercise Price. However, a provider with a flexible genome is more stable against noises. The left part of the graph in Figure 5 also shows that the rigid genetic provider takes much more time in getting trained to be competitive in its prices.

5. CONCLUSIONS AND FUTURE WORK

This paper has shown the effectiveness and capacity of adaptation of genetic algorithms for pricing in Cloud Computing Markets. In a competitive environment, where providers cannot know which strategy other providers will follow, genetic providers earn up to the 100% more than utility-based dynamic pricing providers, and up to 1000% more than a typical fixed pricing provider.

The proposed genetic algorithm is easy to implement and it is flexible enough to be used with a huge set of parameters $\overrightarrow{P}$, even when there is not evidence that some of the parameters have a real influence in the price: the evolutionary selection process will discard all the invalid parameters, so the proposed model can be used to make decisions in complex, even chaotic, environments.

In unstable/unpredictable markets, the experiments clearly showed that a provider with a flexible genome is more stable against noises and rough changes, and evolve to competitive pricings quicker than a provider with a rigid genome.

This work is a first proof of concept of genetic pricing for Cloud Computing Markets whose results have been validated by market simulations. A future line of work is testing the proposed model in real Cloud computing market environments. Another important line of work is creating a *meta-genome* that is able to dynamically tune up some data about the chromosomes and the population, such as the number of chromosomes, the mutation rate, the memory rate of the scoring process, etc. At last, new ways of representing the generic pricing function must be explored, such as the defining more complex the relations between the parameters of the function of the chromosome, such as logarithms, sinus, derivatives, etc.

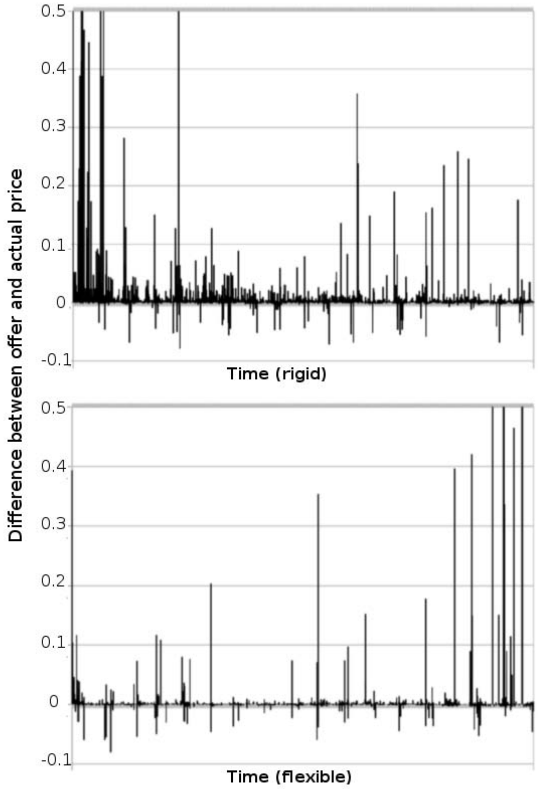

Figure 5: Difference between offer price and Exercise Price, and speed of convergence, of a provider with a rigid genetic algorithm (upper graph) and a provider with a flexible genetic algorithm (lower graph)

Acknowledgment

This work is supported by the Ministry of Science and Technology of Spain and the European Union (FEDER funds) under contract TIN2007-60625, and by the Generalitat de Catalunya under contract 2009-SGR-980.

6. REFERENCES

[1] J. Basney and M. Livny. Deploying a High Throughput Computing Cluster. In R. Buyya, editor, *High Performance Cluster Computing: Architectures and Systems, Volume 1.* Prentice Hall, 1999.

[2] F. Black and M. S. Scholes. The Pricing of Options and Corporate Liabilities. *Journal of Political Economy*, 81(3):637–654, May-June 1973.

[3] N. Borissov, B. Blau, and D. Neumann. Semi-automated Provisioning and Usage of Configurable Web Services. In *Proceedings of the 16th European Conference on Information Systems (ECIS)*, pages 1941–1952, Galway, Ireland, June 9–11 2008.

[4] R. Brunner, F. Freitag, and L. Navarro. Towards the Development of a Decentralized Market Information System: Requirements and Architecture. In *PDCoF'08: First Workshop on Parallel and Distributed Computing in Finance (Computational Finance)*, page 1–7, Miami, FL, USA, April 18 2008.

[5] R. Buyya, C. S. Yeo, and S. Venugopal. Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities. pages 5–13. IEEE Computer Society, September 25-27 2008.

[6] S.-H. Chen, editor. *Genetic Algorithms and Genetic Programming in Computational Finance*. Kluwer Academic Publishers, Norwell, MA, USA, 2002.

[7] S.-H. Cheng, editor. *Evolutionary Computation in Economics and Finance*, volume 100 of *Studies in Fuzziness and Soft Computing*. Springer-Verlag, 2002.

[8] N. K. Chidambaran, C.-W. J. Lee, and J. R. Trigueros. An Adaptive Evolutionary Approach to Option Pricing via Genetic Programming. Finance Department Working Paper Series, New York University, Leonard N. Stern School of Business-, 1998.

[9] D. Cliff. Evolution of Market Mechanism through a Continuous Space of Auction-types. volume 2, pages 2029–2034, Honolulu, HI, USA, May 12–17 2002. IEEE Computer Society.

[10] J. Cordier and M. Gross. *The Complete Guide to Option Selling: How Selling Options can Lead to Stellar Returns in Bull and Bear Markets.* McGraw-Hill, 2004.

[11] M. B. E. Fayek, I. A. Talkhan, and K. S. El-Masry. GAMA (Genetic Algorithm driven Multi-Agents)for e-Commerce Integrative Negotiation. In *GECCO'09: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pages 1845–1846, Montreal, Québec, Canada, July 8–12 2009. ACM.

[12] J. Guitart, M. Macías, O. Rana, P. Wieder, R. Yahyapour, and W. Ziegler. *Market-Oriented Grid and Utility Computing*, chapter SLA-based Resource Management and Allocation. Number 12. Wiley, 2009.

[13] R. D. James, J. E. Hanson, J. O. Kephart, and G. Tesauro. Agent-Human Interactions in the Continuous Double Auction. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pages 1169–1176, Seattle, Washington, USA, August 4–10 2001.

[14] J. R. Koza. *Genetic Programming: on the Programming of Computers by means of Natural Selection.* MIT Press, Cambridge, MA, USA, 1992.

[15] M. Macías and J. Guitart. Influence of Reputation In Revenue of Grid Service Providers. In *Proceedings of the 2nd International Workshop on High Performance Grid Middleware (HiPerGRID 2008)*, pages 9–16, Bucharest, Romania, November 21 2008.

[16] M. Macías and J. Guitart. Using Resource-level Information into Nonadditive Negotiation Models for Cloud Market Environments. In *12th IEEE/IFIP Network Operations and Management Symposium (NOMS'10)*, pages 325–332, Osaka, Japan, April 19–23 2010.

[17] M. A. Rappa. The Utility Business Model and the Future of Computing Services. *IBM Systems Journal*, 43(1):32–42, 2004.