Cloud SLA Negotiation for Energy Saving - A Particle Swarm Optimization Approach

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

***Abstract*—The necessity of balancing the obtained performance with the energy consumed is an emerging ambition for cloud computing research. Performance in cloud computing is defined through Service Level Agreement contracts between the cloud provider and cloud customer, being a projection of the customer's perspective on the service offered by the cloud provider. Although more and more research efforts go into standardizing Service Level Agreement in cloud systems, the area is still at its early ages. This paper proposes a Service Level Agreement negotiation protocol based on particle swarm optimization techniques, for obtaining a balance between the energy consumed and performance offered in the cloud. The two parties of the defined negotiation protocol are the performance-oriented cloud customer and the energy-oriented cloud provider. The agreement resulted from the negotiation process satisfies the two major negotiation properties we aim for: closeness to Pareto optimality and high social welfare.**

***Index Terms*—negotiation, particle swarm optimization, SLA, QoS, cloud computing, energy awareness**

I. Introduction

The energy consumption of data centres directly depends on internal loads, and less on the data centre climate as it was previously believed. Nowadays data centres' total power consumption is around 31GW, an enormous amount if we consider that 1GW power can be used by almost 1 million homes [1]. The same study estimates that the world's data centres energy consumption has increased by 19% only in the year 2011. One of the main factors contributing to these high values in energy consumption is the service centre over provisioning of computing resources.

Several tools such as Green Cloud Scheduler [2] or CLUES [3] address the service centres energy efficiency issue by executing consolidation and dynamic management actions for on-demand energy-aware resource allocation. When a decrease in power usage is desired, one needs to consider the impact that this change has on performance. To be able to consider strategies for energy efficiency optimization, a service centre manager firstly has to address the end user satisfaction expressed as quality of the provided service.

Quality of Service (QoS) has been an important component in the definition of distributed systems, from the early ones to the new paradigms of grid and cloud computing [4]. The definition for cloud QoS has a double dimension [5]. It firstly concerns the agreement between the cloud client and the cloud provider regarding the resources allocated by the cloud manager for a specific task. The second part of the definition contains the relation between the cloud customer which generally is a service provider, and the client of the cloud customer. In this second part we talk about known QoS attributes like throughput or jitter, which are in their turn influenced by resource-level quality of service. For managing the QoS requirements and being able to orient the system towards energy efficiency, the management system of the cloud provider needs to negotiate with the cloud customer a Service Level Agreement (SLA) contract which enables energy awareness, while respecting the cloud customer's requirements. Knowing and managing the QoS requirements gives the provider the opportunity of having a better managed system from the point of view of energy efficiency. After having negotiated a trade-off in service level agreement, the management system will be able to distribute the tasks in the cloud in an energy-efficient manner, while fulfilling the agreed-upon SLA contract.

This paper proposes an approach for cloud Service Level Agreement negotiation, biased on promoting the decrease of resource allocation within the cloud customer's accepted limits, thus leading to lower energy consumption. For computing the negotiation offers we use a Particle Swarm Optimization (PSO) based mechanism, which evolves populations of possible negotiation offers for each party towards a new offer. The PSO-based negotiation process converges towards an approximate negotiation solution, considering both Pareto optimality and maximization of social welfare.

The remainder of the paper is organized as follows. The next section presents the related work. In Section 3 we present the negotiation problem in the context of cloud computing. Section 4 describes our Particle Swarm Optimization approach for Service Level Agreement negotiation while in Section 5 we present the results obtained. The last section, Section 6, presents our conclusions and future work.

II. Related Work

SLA specification, negotiation and enforcement is a relatively new concern in cloud computing. SLA management deals with three main problems: (i) SLA specification standards, (ii) the nature of Service Level Objectives (SLOs) in cloud systems and (iii) the negotiation protocols, including the algorithms used for computing the new offers to be sent to other parties.

The SLA contract specification addresses the need for accurate description of service level objectives together with

978-1-4673-2952-1/12/$31.00 ©2012 IEEE

the deadlines and the necessary constraints for the SLA negotiation stage. For meeting these specification needs, several web service agreement languages have been defined. The Web Service Level Agreement (WSLA) language [6] specifies agreements between a service provider and a customer, and the obligations of the involved parties. Additionally, it allows the specification of measurements in case the agreed-upon contracts are not met by one of the parties. The WS-Agreement specification [7] developed by the participants of the Grid Resource Allocation and Agreement Protocol Working Group of Open Grid Forum [8] proposes an XML-based web services protocol for reaching agreements between two parties. The structure of the negotiation protocol depends on the nature of the negotiation subjects and the type of negotiation parties, oscillating between WS-Agreement, WSLA or even predefined situation-dependent XML templates. In [9] the presented SLAng language describes the definition and agreements for complex QoS specifications. It defines facets differentiating between types of vertical SLAs, which regulate the support from the underlying infrastructure, and horizontal SLAs contracted between services from the same vertical level.

Given the relatively new interest in QoS and SLAs in cloud infrastructures, we focus particularly on related systems like SOAs and grids. I. Bradnic et al. propose in [10] a meta-level for SLA negotiation, which at its turn negotiates the standard and the properties used in the actual negotiation protocol. This approach addresses especially the problems in grid and cloud systems, where the customer and the provider dynamically find each other in an on-demand basis. Another approach [11] proposes an agent-based infrastructure for scheduling, negotiating on different concerns between user agents and super scheduling agents, or super agents and local agents. The first one is a meta-negotiation level, where the user negotiates with the agents the high level QoS requirements, while the latter is a sub-level negotiation, in which the super scheduling agents decompose the meta-SLA into low level resource attributes. The authors of [12] use the request descriptor provided by the client and the machine descriptor given by the provider and compute the performance descriptor which contains the estimated execution time together with other resource-level parameters. During negotiation, a compute resource manager is used by the QoS manager for creating temporary reservations depending on the available grid resources.

In cloud computing, one of the few approaches toward SLA negotiation is VIESLAF [13] whose goal is to propose a general framework for cloud QoS which can generally define SLAs through the introduction of meta-negotiation and SLA-mappings for bridging the gap between inconsistent SLA templates. For the latter, a MAPE (Monitoring, Analysis, Planning and Execution) loop is used in order to solve conflicts between SLA-mappings. Ferretti et al. in [5] describe a QoS-oriented middleware architecture for configuration, management and optimization of cloud resources in a QoS-aware manner. This approach deals with the resource overprovisioning problem by dynamically allocating resources on an on-demand basis, without violating the SLAs. Compared with this solution, the Global Loop Component of the GAMES project [14] orients this dynamism to energy-awareness, while still agreeing with the negotiated SLAs resulted from the energy and QoS balancing stage. The current paper discusses this energy efficiency oriented SLA negotiation, presenting the algorithm used and the results obtained.

Negotiation protocols are key issues in distributed systems, considering the automation of quality of service management. Considering mechanisms and existent standards defined in business and economics domains, researchers have defined protocols for agent negotiation in distributed systems, giving heuristics and innovative techniques for measuring the quality of an offer and constructing a new offer in return. In [15], an approach for multi-attribute negotiation resulting in Pareto optimal solutions is presented. The proposed model decomposes the n-dimensional negotiation space into a series of base lines, each containing a simpler heuristic negotiation process. Sierra et al. propose in [16] a formal negotiation model for autonomous agents, which is being used to generate initial offers, evaluate the proposals and offer counter-proposals. This work is continued in [17] [18] [19] by proposing different constraints and deadlines, agent preferences over the issues and different issue packing approaches (independent issues, sequential negotiation or package deal procedure). Gatti et al. describe in [20] an approximate Pareto optimal solution for negotiation under continuous interdependent issues. The solution resulted from the cooperative negotiation process described above is Pareto optimal while most of the offers reside on the Pareto frontier. This is a great advantage of our solution, considering that throughout the negotiation process the offers made are Pareto optimal, so the only matter remained to be considered is the social welfare of the parts. As opposed to this approach, the one we will present in what follows doesn't use a mediator, being completely decentralized and using algorithms inspired from biology for evaluating the proposals and computing the counter offer proposals.

III. SLA Negotiation in Cloud

With emerging new technologies like cloud computing, the quality of the service offered cannot be disregarded. Similar to the evolution of Grid, cloud computing needs to have well defined standards with regard to the QoS, the SLA negotiation, management and monitoring. In this paper we are concerned with the SLA definition in cloud and its negotiation.

A. Negotiation and Service Level Agreement in Cloud Computing

In the cloud computing context, the negotiation process has two parties: the cloud provider and the cloud customer. The cloud provider hosts on virtual machines customer's services, in return receiving an amount of money from the cloud customer. The cloud customer needs on-demand resources in order to host its application. The acceptable or even optimum situation is found through a negotiation process which aims at finding the equilibrium between the two apparently opposable parts.

Therefore, SLA negotiation is necessary for managing two types of trade-offs: (i) the trade-off the cloud infrastructure makes from the point of view of the price obtained and energy saved, and (ii) the trade-off the customer makes from the point of view of the resources received (directly related to the performance obtained) and the cost of the resources.

For interchanging the offers between negotiation parties we need a SLA specification standard describing the negotiation proposals. We use WS-Agreement [7] for defining the proposals from one party to another, and an agent-based implementation for the negotiation process.

The quality of the negotiation solution is still subject to debate. Jennings et al. give in [21] a thorough description of automated negotiation. According to their classification, the quality of the negotiated solution can be defined in terms of maximizing social welfare, Pareto efficiency, guaranteed success, individual rationality and stability (Nash Equilibrium) [22]. However, negotiation principles [23] like equality, equity and need or even negotiation fairness, are considered concepts that cannot be defined mathematically, and thus they don't make the point of our current interest.

B. Service Level Objectives in Cloud

Generally, in service oriented architectures, Service Level Objectives (SLOs) are specific measurable characteristics contained within SLA [24]. In cloud computing, SLOs can be defined on two levels [5]: (i) between the cloud provider and the cloud customer and (ii) between cloud customer and its services users. The first level focuses on low level resource allocation. The second level focuses on the agreement between the cloud customer, which is in turn a service provider, and the client of the service he offers, defining higher level SLOs. Considering the fact that the second type of SLOs can be translated into low level requirements or resource-level QoS metrics as demonstrated in [25], for simplicity we will consider solely the low level SLOs.

In our approach, the SLOs have a maximum and minimum level defined by both the customer and the provider. In the case of resource allocation, the provider has as goal the minimum level of used resources for encouraging energy efficiency, while the cloud customer needs a maximum level of resources for performance.

Listing 1: Customer acceptable range SLA definition for memory

```
<wsag:ServiceDescriptionTerm
  wsag:Name="memoryHigh
  wsag:ServiceName="ComputeJob1">
    <job:memorySize>200</job:memorySize>
</wsag:ServiceDescriptionTerm>
<wsag:ServiceDescriptionTerm
  wsag:Name="memoryLow"
  wsag:ServiceName="ComputeJob1">
    <job:memorySize>100</job:memorySize>
</wsag:ServiceDescriptionTerm>
```

When speaking about the price paid for the resources to be allocated, the trend is vice-versa: the provider tends to request a maximum amount, while the customer will try to pay a minimum amount. An example of this specification for the cloud customer case is visible in Listing 1 which presents a WS-Agreement description for the acceptable range of the memory negotiation issue.

Listing 2 shows an example of an offer structure, in which the value for the term Memory is 120, in other words the cloud customer offers to agree on 120MB of RAM.

Listing 2: Customer SLA offer for memory

```
<wsag:AgreementOffer>
  <wsag:Terms>
     <wsag:ServiceDescriptionTerm
     wsag:Name="memory"
     wsag:ServiceName="ComputeJob1">
     <job:memorySize>120</job:memorySize>
     ...
     </wsag:ServiceDescriptionTerm>
  </wsag:Terms>
</wsag:AgreementOffer>
```

IV. PSO Negotiation

This section presents the Particle Swarm Optimization (PSO) based negotiation mechanism. At the beginning of the negotiation process, a hybrid variant of a PSO algorithm is employed by each negotiating party in order to define its search space. During the negotiation process, each agent updates the position and velocity of its own proposals according to the offers received. We firstly present the main concepts of the particle swarm inspired heuristics. Next we map the negotiation terms to the PSO world, and explain why this mapping has been used. In the last part of this section we detail the negotiation mechanism.

A. Particle Swarm Optimization background

Particle swarm optimization is a heuristic search technique introduced in the mid 1990 by Kennedy and Eberhart [26] as an attempt to mimic the graceful, yet unpredictable motion of a flock of birds. PSO is generally used for multi-objective optimization, each particle (bird) being mapped to a possible solution for the problem to be solved. The particles are evolving towards a better solution, being moved trough the search space considering mathematical formulas for updating the particle's position and velocity. Changes to the particle's distance and velocity are determined by the particle's own experience as well as the experience of other individuals from the swarm.

In the first stage, the swarm is initialized and each particle is associated with a randomly generated position and velocity, as well as with a communication channel with other particles. Each particle communicates with the other particles based on a network topology that defines the structure of the swarm.

The position of a particle is altered by adding a velocity vector v_t to its current position (see Relation (1)).

$$x_t = x_{t-1} + v_t \tag{1}$$

The velocity vector illustrates the information exchanged between the particles and is updated using Relation (2).

$$v_t = W * v_{t-1} + c_1 * r_1(pBest - x_t) + c_2 * r_2(leader - x_t) \tag{2}$$

Algorithm 1 PSO Algorithm

```
1: InitializeSwarm()
2: leader := DetermineLeader()
3: repeat
4:   for all particle ∈ swarm do
5:     UpdateParticleVelocity(particle)
6:     if fitness(particle) ≥ fitness(pBest) then
7:       pBest := particles
8:     end if
9:     if fitness(particle) ≥ fitness(leader) then
10:      leader := particle
11:    end if
12:  end for
13: until terminationCriterion
```

The W, c_1, c_2 parameters are empirically chosen considering both the existent studies and the behavior of the algorithm, while r_1 and r_2 are randomly generated values. Each particle has associated a fitness function which gives its closeness to the swarm goal. Over time, particles are accelerated towards the particles within their communication neighbourhood which have a better fitness function.

B. PSO - Negotiation Mapping

The PSO algorithm is used by both negotiation parties, i.e. the provider and the customer, for computing the offers to be exchanged during the negotiation process.

A particle in PSO corresponds to a SLA proposal, containing information for all the issues under consideration (in the current case, resource value, price, response time, etc.). In PSO, the swarm can be seen as a set of all the birds in the flock. In SLA negotiation, the swarm is the set of all the possible offers considered by a party at a given time(see Table I). For a better mapping on the real life phenomena we define the concept of *wind* in the PSO algorithm. The wind is a force which affects particles' movement in swarm, leading them away from the swarm goal. Wind is generated by the counter-offer of the opponent, and carries away the particles (possible offers) from the goal.

The cognitive process of computing new proposals and evaluating possible offers is associated to PSO swarm evolution. Just like in real life, offers are considered, evaluated and used for exploring new solutions which will lead us to the best solution.

In the negotiation process, each party has attached a SLA goal, which is the optimal SLA agreement. However, in most cases the contract desired by a party can't be fulfilled by the other negotiating party, otherwise the negotiation process wouldn't be necessary. This is why the counter-offer in our negotiation protocol takes the average of the considered offers, not the one closest to the goal. In this way, the alternating offers reflect each party's swarm evolution, and therefore the cognitive process during the negotiation process.

PSO	Negotiation
particle	possible offer
swarm	collection of possible offers
wind	effect of counter-offer (compromise)
swarm evolution	computing new proposal (evolution of thought)
swarm average particle	counter-offer

TABLE I: PSO-Negotiation mapping

C. Negotiation Model

This section presents in detail the non-mediated bilateral multi-issue, time-dependent negotiation model with private agent preferences on which this paper is focused. Each negotiation party is represented by an agent and hence we will have two agents: customer agent and provider agent.

The first phase of the negotiation process is the pre-negotiation. It represents the operational and strategic process of preparation and planning for negotiation. In particular, this phase can be broken down into three distinct activities [27]: (i) structuring personal information, (ii) analysis of the opponents, (iii) definition of the protocol and selection of the initial strategy.

During the *structuring personal information* activity, each agent defines a goal to reach, prioritizes negotiation issues by assigning a weight to each one and defines a maximum accepted compromise for each negotiation issue. The client starts by defining the issue to negotiate and the accepted interval of each issue (see Relation (3)).

$$Issue\_Specification_{Res_i} = \{value \in \Re \mid resValue_i \geq minValue_i \wedge resValue_i \leq maxValue_i\} \quad (3)$$

The SLA will firstly contain the name for each party and its role, the complete set of issues that concern the negotiating parties and the negotiation characteristics like expected negotiation time, maximum number of offers that each side is willing to exchange (see Relation (4)).

$$SLA\_Specification = \cup_i\ issueSpecification_i \cup_i\ negotiationCharacteristics_i \cup_i\ partySpecification_i \quad (4)$$

Each agent also defines its fitness function, which gives the degree of satisfaction with the current offer (see Relation (5)), being equal to the distance (Euclidian, Chebyshev, etc.) from the current offer to the party's goal.

$$fitness_{offer_i} = distance(offer_i, goal) \quad (5)$$

After structuring its information, in the *definition of the protocol and selection for the initial strategy* activity each agent populates its swarm with particles close to the defined

goal. The global best particle of the swarm is initialized with the goal of each agent and remains the same throughout the entire process of the negotiation. Considering the fact that the model we propose deals with unknown strategies for the opponent, the *analysis of the opponents* activity dimension is not necessary.

The negotiation process is triggered by the cloud customer sending its own goal as the first offer. Afterwards, the negotiation process consists of the exchange of offers and information between negotiation parties in order to move towards an agreement. For determining the counter-offer several steps are followed (see Algorithm 2).

Algorithm 2 PSO Negotiation Model

1: $InitializeNegotiation()$
2: **while** $fitness(currentOffer) \leq threshold$ **do**
3: $compromise := ComputeCompromise(currentOffer)$
4: $UpdateSwarm(swarm, currentOffer, compromise)$
$\{$ *The steps* 3 $\rightarrow$ 14 *from PSO Algorithm* $\}$
5: $counterOffer := ComputeAverage(swarm)$
6: $SendOffer(counterOffer)$
7: $currentOffer := WaitForNewOffer()$
8: **end while**
9: $SendToOpponent(offerAccepted)$

Figure 1 shows the first steps of the negotiation process. When a new offer is received, the party checks whether the offer can be accepted (Line 2). The fitness threshold for accepting an offer varies with time, therefore at the beginning of the negotiation process the threshold will be at its highest level, accepting only high-fitness offers, while at the end of the negotiation process the agent will scale its threshold towards the minimum specified acceptable fitness. In relation (6) the $ScaleThreshold_t$ function helps with asymptotically decreasing the agent's threshold with time towards the $ThresholdLow$ value.

$$Threshold_t = ThresholdHigh - (ThresholdHigh - ThresholdLow) * \frac{ScaleThreshold_t}{ScaleThreshold_t + n} \quad (6)$$

The $ThresholdLow$ value is defined by each negotiation party, while the $ThresholdHigh$ is 1, the maximum possible for the normalized fitness. If the fitness of the offer is bellow the computed threshold, the agent samples for new offers from its search space and selects a counter-offer to be sent.

Firstly a selection operator is defined in order to determine the individuals in the swarm altered by the opposite agent's offer. An offer from the opposite agent acts as wind for a particle in the swarm, and modifies its velocity and distance according to a compromise factor (Line 3). The number of individuals affected by the opposite agent's counter offer (see Relation (7)) varies with time: fewer particles (offers) are selected in the beginning of the negotiation process and their number increases with time. In this way, the agent is more stubborn in the beginning of the negotiation process, as fewer

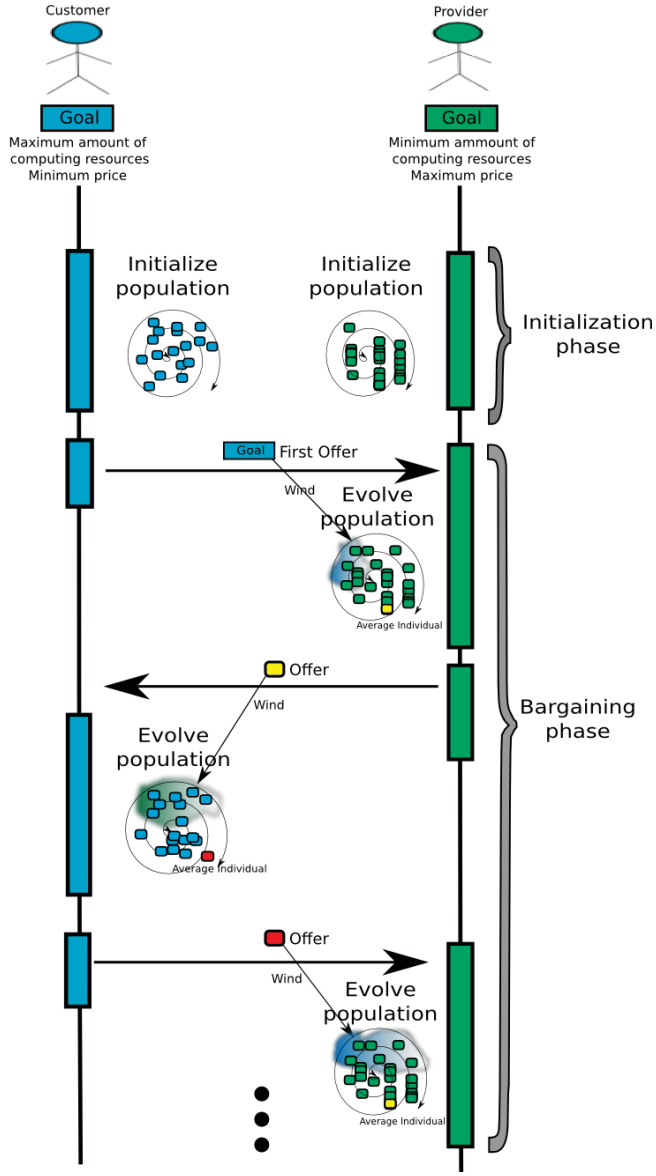

Fig. 1: Steps in the Negotiation Process

particles in the swarm are affected by the opposite agent's offer which becomes increasingly more important with time.

$$nbAffectedIndividuals_t = swarmSize * \delta_t, \quad \delta_t = f(\frac{availableTime_t}{totalTime_t}) \quad (7)$$

After selecting the particles which will be affected by the wind force (opposite agent's offer), the new properties of position and velocity are computed using the relations (8) and (9) (Line 4).

$$v_{t+1} = \lambda * (oppositeOffer - x_t) \quad (8)$$

$$x_{t+1} = x_t + v_t \quad (9)$$

The compromise factor λ is asymptotic with time, as the time passes the agent being more inclined to having high compromise factor in order to reach an agreement.

The counter-offer will be the average of particles existent in the swarm (Line 5), reflecting the overall needs of the current party. The counter-offer acts as wind and affects part of the opponent swarm, leading it further away from the swarm goal. After the counter-offer is being computed, the current agent sends it to the other agent (Line 6) and waits for an acceptance/refusal or a new proposal from the second party (Line 7). After receiving an offer which is greater than the defined threshold, the agent sends to the opponent an acceptance message (Line 9).

V. Case Study and Results

This section presents a case study on which we applied the above presented algorithm, together with the results showing the ability of the negotiation process to converge to a close to optimum result, and reach a balance in QoS requirements and the energy consumed, the latter being proportional to the allocated resources.

For implementing the negotiation framework we have used the Java language, together with the Java Agent DEvelopment (JADE) Framework [28]. The negotiation framework has an agent-based architecture with two main agents: the provider agent and the customer agent. These two agents interchange offers specified using the WS-Agreement language, describing the offered resource levels and the price for them.

Resource name	Acceptable Range	Weight
Memory	200-900 (MB)	0.5
CPU	100-300 (MHz)	0.3
Price	50-100 (Euro)	0.2

TABLE II: Customer Issues Description

Table II gives the acceptable ranges from the customer's point of view for the issues under negotiation. We can see here that the customer desires alot of memory, and less CPU, having a memory intensive task. The provider will prefer lower values for resource provisioning with a higher price (see Table III).

Resource name	Acceptable Range	Weight
Memory	100-700 (MB)	0.4
CPU	100-250 (MHz)	0.4
Price	70-120 (Euro)	0.2

TABLE III: Provider Issues Description

Other than the acceptable ranges for the issues, each party defines its lower possible threshold for the fitness function, which in this case are 0.4 for the customer and 0.5 for the provider. Depending on the values for this limits, the algorithm will converge faster or slower towards a solution.

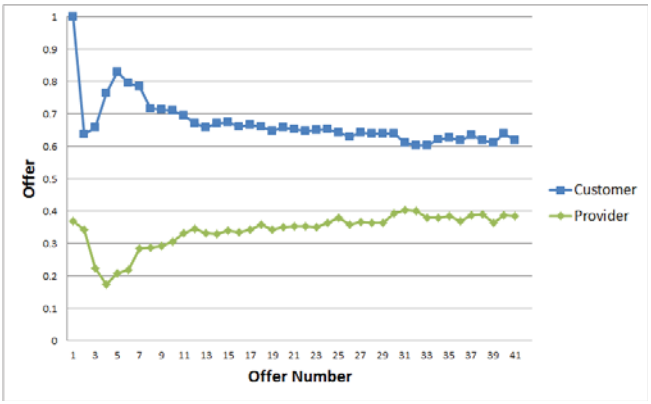

Fig. 2: Offer evolution

Both the agents use the swarm-inspired protocol for negotiation, with empirical-derived parameters for the presented algorithm. For this case study, considering the best PSO practices [29] and our own empirical tests, we have set the parameters from the Relation (2) to $W = 0.5, c_1 = 1.5$ and $c_2 = 1.5$. For computing the fitness of each particle, we use an Euclidian distance on the normalized resources.

Figure 2 shows the compromise made by each of the negotiating parties. For quantifying the offer we use a weighted average among the issues under negotiation, from resources to price (see Relation (10)). Therefore, from these offers we can see that while the cost of the provider increases, the fitness of the customer decreases. The fast downfall in the customer's offers is due to the growing awareness in provider's desires. At first, the customer sends its goal as an offer, and after finding out the desires of the provider, there comes an attenuation period in which each agent tries to impose its own requirements. After that, the two agents go through a cooperation period in which they compromise and reach a common agreement. As shown in Figue 2, eventhough the preferences of the agents don't meet, the fitness thereshold which has reached a lower level allows the customer agent to accept the received offer. The negotiated solution is of 521.20 MB, 220.02 MHz and 84.5 Euros.

$$Offer_t = \frac{\sum issueWeight_i * issueValue_i}{\sum issueWeight_i} \quad (10)$$

For deciding on the quality of the negotiation solution, we plot the Pareto frontier chart, which tells us which are the Pareto optimal points. Generally, a solution is Pareto optimal when there is no possible allocation that could make someone better off without making someone else worse off. In bilateral negotiation, a Pareto optimal offer is the combination of issues that cannot be improved simultaneously for both parties.

In Figure 3 the blue dots represent the Pareto frontier. The Pareto frontier is composed of Pareto optimal solutions, in which we encounter the minimum cost both for the cloud customer and the cloud provider. The cost function is the reverse of the fitness function: while we always intend to maximize the fitness we always need to minimize the cost.

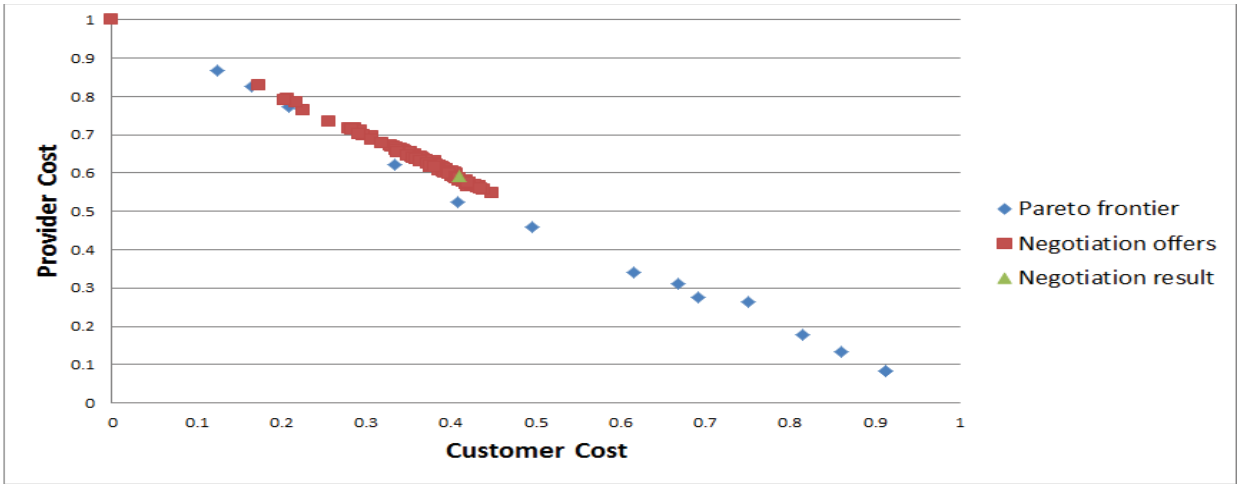

Fig. 3: Comparison with Pareto frontier

Here we can observe that the interchanged offers are constantly close to the Pareto frontier. This means that whenever the two agents decide to stop the negotiation process, the resulted solution will be one close to Pareto-optimality. Moreover, the negotiated solution has high social welfare, maximizing the sum of the two parties' fitness functions.

The "guaranteed success" property of negotiation protocols is another important property, next to the Pareto optimality and social welfare maximization, since this property ensures that eventually an agreement is reached. For our negotiation protocol, reaching a solution depends on the configuration and the behaviour of the opposable parties.

If both agents use the PSO-based negotiation algorithm, they will eventually reach an agreement regardless of the specified compromise function. This is a result of the fact that the particles are affected by the counter-offers, and the two party's swarms are basically converging towards one another, until they reach a solution. For fastening the convergence towards an agreement, the acceptance threshold is also being altered with time, such that the smaller the remaining time, the lower the threshold value.

If one of the negotiation parties uses a negotiation model different than the PSO-based one, we need to be sure that the agent which uses PSO-based negotiation doesn't progressively lower its expectations while receiving nothing in return. This is why the swarm behaviour doesn't depend only on the elapsed time since the start of the negotiation process, but also on the compromise made by the other agent. When it would benefit the agent using the PSO negotiation model (i.e. the other agent is not willing to compromise) we might end up with no solution. Otherwise, the algorithm will surely converge to at a faster or lower rate, depending on the values of the parameters specified for each negotiation party.

VI. Conclusion

In this paper we approach the problem of cloud resources overprovisioning by proposing a PSO-based mechanism for negotiating a trade-off between the cloud customer requirements and the provider requirements which include energy-awareness necessities. The negotiation framework specifies the offers using a service level agreement standardized language, WS-Agreement [7], proposing a general solution from the higher layer of SLA specification to the lower one of SLA negotiation. The bio-inspired model maps nature elements of wind and particle to the negotiation world, defining counter-offer computation and offer evaluation mechanisms. The test case results shows that using our negotiation mechanism the customer and provider converge towards a SLA contract which is Pareto-optimal, and adds welfare for the both negotiation parties.

Acknowledgment

This work has been supported by the European Commission within the GAMES project [14] funded under EU FP7.

References

[1] (2011) The EnergyStar website. [Online]. Available: http://www.energystar.gov

[2] T. U. o. C.-N. Distributed Systems Research Laboratory. (2012) Green cloud scheduler (gcs). [Online]. Available: http://www.coned.utcluj.ro/GreenCloudScheduler

[3] U. P. d. V. Grid y Computacion de Altas Prestaciones (GryCAP) group. (2012) Cluster energy saving (clues). [Online]. Available: http://www.grycap.upv.es/clues

[4] A. Leff, J. T. Rayfield, and D. M. Dias, "Service-level agreements and commercial grids," *IEEE Internet Computing*, vol. 7, no. 4, pp. 44–50, Jul. 2003. [Online]. Available: http://dx.doi.org/10.1109/MIC.2003.1215659

[5] S. Ferretti, V. Ghini, F. Panzieri, M. Pellegrini, and E. Turrini, "Qos-aware clouds," in *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, ser. CLOUD '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 321–328. [Online]. Available: http://dx.doi.org/10.1109/CLOUD.2010.17

[6] H. Ludwig, A. Keller, A. Dan, R. P. King, and R. Franck, "Web service level agreement (wsla) language specification," *Language*, pp. 1–110, 2003. [Online]. Available: http://www.research.ibm.com/wsla/WSLASpecV1-20030128.pdf

[7] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu, "Web Services Agreement Specification (WS-Agreement)." [Online]. Available: https://forge.gridforum.org/projects/graap-wg/

[8] Open grid forum. [Online]. Available: http://www.ogf.org/

[9] D. D. Lamanna, J. Skene, and W. Emmerich, "Slang: A language for defining service level agreements," in *Proceedings of the The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems*, ser. FTDCS '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 100–. [Online]. Available: http://dl.acm.org/citation.cfm?id=795675.797134

[10] I. Brandic, S. Venugopal, M. Mattess, and R. Buyya, "Towards a meta-negotiation architecture for SLA-aware grid services," in *Workshop on Service-Oriented Engineering and Optimizations 2008*, Dec. 2008. [Online]. Available: http://www.hpl.hp.com/india/senopt08/papers/senopt08106.pdf

[11] D. Ouelhadj, J. Garibaldi, J. Maclaren, R. Sakellariou, and K. Krishnakumar, "A multi-agent infrastructure and a service level agreement negotiation protocol for robust scheduling in grid computing," in *In Advances in Grid Computing - EGC 2005, volume 3470 of Lecture Notes in Computer Science*. Springer Verlag, 2005, pp. 651–660.

[12] S. E. Middleton, M. Surridge, S. Benkner, and G. Engelbrecht, "Quality of service negotiation for commercial medical grid services," *Journal of Grid Computing*, vol. 5, no. 4, pp. 429–447, 2007. [Online]. Available: http://eprints.ecs.soton.ac.uk/17598/

[13] I. Brandic, D. Music, and S. Dustdar, "VieSLAF framework: Facilitating negotiations in clouds by applying service mediation and negotiation bootstrapping," vol. 11, no. 2, pp. 189–204, Jun. 2010.

[14] (2009-2012) Games project (green active management of energy efficiency in service centres). [Online]. Available: http://www.green-datacenters.eu/

[15] G. Lai, K. Sycara, and C. Li, "A pareto optimal model for automated multi-attribute negotiations," in *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, ser. AAMAS '07. New York, NY, USA: ACM, 2007, pp. 246:1–246:3. [Online]. Available: http://doi.acm.org/10.1145/1329125.1329423

[16] C. Sierra, P. Faratin, and N. R. Jennings, "A service-oriented negotiation model between autonomous agents," in *Proceedings of the 8th European Workshop on Modelling Autonomous Agents in a Multi-Agent World: Multi-Agent Rationality*. London, UK, UK: Springer-Verlag, 1997, pp. 17–35. [Online]. Available: http://dl.acm.org/citation.cfm?id=646909.710674

[17] S. S. Fatima, "Approximate and online multi-issue negotiation," in *In Proceedings of the 6th International Joint Conference on Autonomous Agents and Multi-agent Systems*, 2007. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.63.9477

[18] S. S. Fatima, M. Wooldridge, and N. R. Jennings, "Multi-issue negotiation with deadlines," *J. Artif. Int. Res.*, vol. 27, no. 1, pp. 381–417, Nov. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1622572.1622583

[19] S. Fatima, M. Wooldridge, and N. R. Jennings, "Optimal agendas for multi-issue negotiation," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, ser. AAMAS '03. New York, NY, USA: ACM, 2003, pp. 129–136. [Online]. Available: http://doi.acm.org/10.1145/860575.860597

[20] N. Gatti and F. Amigoni, "An approximate pareto optimal cooperative negotiation model for multiple," in *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, ser. IAT '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 565–571. [Online]. Available: http://dx.doi.org/10.1109/IAT.2005.40

[21] N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parsons, M. Wooldridge, and C. Sierra, "Automated negotiation: prospects, methods and challenges," *Intern. J. of Group Decision and Negotiation*, vol. 10, no. 2, pp. 199–215, 2001.

[22] J. Nash, *The Annals of Mathematics*, vol. 54, no. 2, pp. 286–295, 1951.

[23] H. Raiffa, *The art and science of negotiation*. Howard University Press, 1982.

[24] G. L. P. Bianco and P. Merson, "Service level agreements in service-oriented architecture environments," *Software Engineering Institute,Carnegie Mellon University, Pittsburgh, Pennsylvania*, 2008.

[25] Y. Chen, S. Iyer, X. Liu, D. Milojicic, and A. Sahai, "Sla decomposition: Translating service level objectives to system level thresholds," in *Proceedings of the Fourth International Conference on Autonomic Computing*, ser. ICAC '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 3–. [Online]. Available: http://dx.doi.org/10.1109/ICAC.2007.36

[26] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4. IEEE, Nov. 1995, pp. 1942–1948 vol.4. [Online]. Available: http://dx.doi.org/10.1109/ICNN.1995.488968

[27] F. Lopes, M. Wooldridge, and A. Q. Novais, "Negotiation among autonomous computational agents: principles, analysis and challenges," *Artif. Intell. Rev.*, vol. 29, no. 1, pp. 1–44, Mar. 2008. [Online]. Available: http://dx.doi.org/10.1007/s10462-009-9107-8

[28] Java agent development framework (jade). [Online]. Available: http://www.jade.tilab.com/

[29] R. Hassan, B. Cohanim, O. De Weck, and G. Venter, "A comparison of particle swarm optimization and the genetic algorithm," in *46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2005, pp. 1–13.