Simulating Negotiation-based Cloud Markets

Benedikt Pittl, Werner Mach, and Erich Schikuta

Faculty of Computer Science, University of Vienna, Vienna, Austria
{firstname.lastname}@univie.ac.at

Abstract. Today, the so called supermarket approach is used for trading Cloud services on Cloud markets. Thereby, consumers purchase Cloud services at fixed prices without negotiation. More dynamic Cloud markets are emerging as e.g. the recent development of the Amazon EC2 spot market - with spot blocks and spot fleet management - shows. Hence, autonomous Bazaar-based negotiations are a promising approach for trading Cloud services on future Cloud markets. Thereby, market participants negotiate the characteristics of Cloud services which are described in Service Level Agreements (SLAs). Specifications such as the WS-Agreement Negotiation standard foster the development of such Bazaar-based Cloud markets.
In this paper we present a scientific simulation environment for the simulation of Bazaar-based Cloud markets which is conform to the WS-Agreement Negotiation standard. A three-stepped process is required for using the simulation environment: first consumers, intermediaries and providers have to be created, then strategies have to be assigned to them before the result of the simulation can be analyzed. The aim of the simulation environment is to support market participants during the evaluation of their negotiation strategies.

CR6

Keywords: Cloud Simulation · Cloud Market · Cloud SLAs.

1 Introduction

A Cloud market is the culmination point of stakeholders providing and requiring services. Recently, Gartner predicted a growth of 38.6% for the Infrastructure as a Service (IaaS) market in 2017 [12]. Infrastructure services such as virtual machines (VMs) are mainly traded on provider platforms whereby Amazon Web Services (AWS) with the EC2 platform is market leader [9]. Amazon EC2 supports four different marketspaces for trading virtual machines: (i) On the reservation marketspace consumers and providers have a long-term relationship with a fixed, predefined price. (ii) A marketspace exists where consumers can resell virtual machines with a long-term contract - which were purchased on the reservation market - to other consumers. (iii) Consumers on the on-demand marketspace pay per hour for a virtual machine whereby the prices are higher than the prices on the reservation marketspace. (iv) The spot marketspace is more dynamic: here consumers can bid for virtual machines. The higher the bid, the higher is the chance of getting the virtual machine. The recent development

CR6

of Amazons spot marketspace - with spot blocks and spot fleet management - shows that dynamic Cloud markets are gaining popularity. The notion of such a dynamic Cloud market is not a simple buyer-seller relationship, there are numerous other intermediaries involved in it. Papers of e.g. Weinmann [31, 32] consider intermediaries as important players on future Cloud markets - see also [3, 8, 20]. Strategies as well as a detailed analysis of the impact of such intermediaries are missing. We envision a whole network of market participants which negotiate autonomously with each other against end-user requirements resulting into binding SLAs and consequently to a temporary value network. During negotiation the participants exchange offers and counteroffers - such negotiations are called Bazaar-based negotiations - see e.g. [17, 24] for our previous work on this topic. Specifications such as the Web Service Agreement Negotiation specification (WS-Agreement) [30] support the development of such Bazaar-based Cloud markets. Due to the high number of different market participants as well as the infinite number of possible negotiation strategies simulation environments are an eligible approach to assess the success of negotiation strategies under changing market conditions. For simulating such markets we did not identify appropriate frameworks: (i) The framework wsag4j [33] allows to create WS-Agreement documents in Java but has no simulation capabilities. (ii) The simulation environment greenCloud [14] was developed by the University of Luxembourg. It focuses on the simulation of energy consumption of Cloud infrastructures. (iii) iCanCloud [21] is a Cloud simulation framework for analyzing trade-offs between costs and performance of a given set of applications executed on a certain hardware. (iv) Genius is a generic simulation environment focusing on negotiations without any Cloud specific simulation capabilities. (v) The CloudSim framework [5] was developed at the University of Melbourne and is widely used in the scientific community. It is able to simulate Cloud datacenters but no Cloud markets.

CR6

The previous ICCS conferences underpin a trend towards domain-specific simulation environments in the scientific community - see [1] for a detailed analysis. So e.g. in [7] the authors present a simulation approach for search engine services with a special aim on measuring the impact of different configurations on the performance. In [11] the authors focused on simulating financial portfolios for stress testing scenarios with suppes-bayes causal networks while in [19] the authors developed a simulation environment for evacuation scenarios at the Gdansk University of Technology. Unlike generic simulation environments such as e.g. Genius[1] domain-specific simulation environments are designed for simulating a narrow domain comprehensively[2].

Due to the lack of a simulation environment which is able to simulate the envisioned Bazaar-based Cloud market we developed our own simulation environment based on CloudSim as this framework (i) is well known by the community and, (ii) offers Cloud specific simulation capabilities. The subject of negotiation

[1] Genius is a negotiation simulation tool - see http://ii.tudelft.nl/genius/

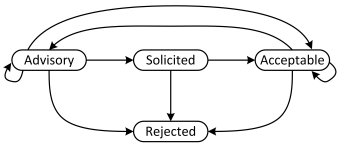
[2] The distinction between domain specific and generic simulation environment is fuzzy and a discussion about this is out of the scope of this paper.

of our simulation environment are virtual machines as an example of a Cloud service.

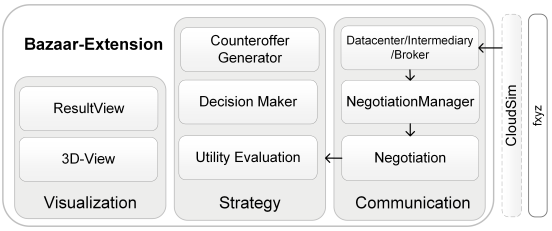
The remainder of the paper is structured as follows: In the following section we present foundations of Cloud markets. The architecture of the simulation environment is summarized in section 3 while an overview of the implemented simulation environment is given in section 4. Section 5 contains a summary of a use case which we executed with the simulation environment. The paper closes with the conclusion in section 6.

2 Background

Amazon's EC2 on-demand marketspace[3] is an example of a platform which applies the supermarket approach. Here, consumers and providers trade services without negotiating price and service characteristics. More dynamic market mechanisms are currently emerging - see e.g. Amazon's EC2 spot marketspace[4]. The scientific community suggests e.g. auction-based approaches [4, 28] or bilateral negotiation-based approaches [10, 23] for future Cloud markets. Latter are based on the alternating exchange of offers which leads to negotiation trees - hence they are called Bazaar-based negotiations. The WS-Agreement Negotiation standard [30] is maintained by the Open Grid Forum and aims on specifying such negotiations. It is an extension to the WS-Agreement standard [2] and describes a XML based structure of offers as well as their possible states. In total the WS-Agreement Negotiation standard defines four states of offers. These four states and their transitions are illustrated in figure 1a.



(a) Negotiation offer states (message types) of the WS-Agreement Negotiation standard



(b) High level architecture of the main components

Fig. 1: Offer states and architecture of the simulation environment

An offer in the advisory state requires further negotiation as it is e.g. not completely specified. The solicited state is used for offers which are completely specified. The negotiation party which receives such an offer has to either accept the offer so that the state of the offer becomes acceptable or reject it which leads

[3] https://aws.amazon.com/de/en/ec2/pricing/on-demand/
[4] https://aws.amazon.com/en/ec2/spot/

to the state rejected. Acceptable offers might result into agreements. Agreements are offers to which consumers and providers agree. Offers in the acceptable state of the WS-Agreement Negotiation standard are not binding: *The ACCEPTABLE state indicates that a negotiation participant is willing to accept a negotiation offer as is.* But it is also described that *there is no guarantee that a subsequent agreement is created.* So in [16] we extended the specification by introducing a binding state. The rejected state is used for offers which are rejected. To improve the readability of the paper we call offers in the acceptable state acceptable messages, offers which are in the rejected state are termed reject messages and offers which are in the solicited state are called solicited messages. We use the term offers to either refer to all offers or to refer to offers in the advisory state - it should be clear from the context.

While the WS-Agreement Negotiation standard describes XML based offers and different states of offers, a concrete negotiation strategy is not specified. We surveyed existing bilateral service negotiation strategies in [25] but we have not found any WS-Agreement Negotiation compliant negotiation strategy. However, the need for WS-Agreement Negotiation compliant strategies was emphasized in [22, 27]. Descriptions of negotiation frameworks such as in [13, 18] elaborate on the importance of the WS-Agreement Negotiation standard without introducing compliant strategies. The negotiation strategy introduced in [34] mentions the WS-Agreement standard but does not use the WS-Agreement Negotiation standard. Instead, the strategy was developed to comply with the FIPA standard. In [29] a bilateral negotiation strategy was introduced. Thereby, the WS-Agreement Negotiation standard is mentioned but not considered for the introduced negotiation strategy. In [26] foundations of the simulation environment were introduced without a concrete negotiation strategy and use case.

Our analysis shows that the scientific community introduced bilateral SLA negotiation strategies. However, a systematic analysis of these strategies under changing market conditions is missing as well as a simulation environment which allows to asses and compare them.

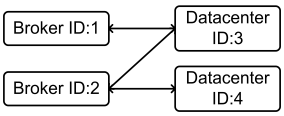
3 Architecture Overview

Our simulation environment implements the concepts of the WS-Agreement standard described in section 2. A high level architecture of the simulation environment is depicted in figure 1b. The simulation environment is based on CloudSim and uses fxyz for the creation of the 3D-View. There are three different types of participants: datacenter (representing providers), intermediary and broker (representing consumers). They inherit the structure and behavior of the CloudSim entity. Each entity has a negotiation manager which acts as a gateway: it forwards received messages to the corresponding negotiations. The negotiation component is a container which stores the negotiation history. Further, it has a reference to the used negotiation strategy. The components are detailed in the following paragraphs.

Negotiation Messages. Bazaar-based negotiations are characterized by the alternating exchange of offers between market participants. These offers are stored in messages - also termed events. In CloudSim, a event has three important fields which are summarized in figure 2a:

- The content of a message is stored in the field MsgContent which is of type *Object*.
- The type of a message is represented by the field MsgType. It is an integer and also termed tag. For example, the integer 2 represents the type of message *Register_Resource*. It is used by datacenters to register at the CIS[5].
- Each entity has an id which is used by CloudSim to deliver messages. This id is stored in the *To* field.

(a) Message structure of CloudSim (simplified)



(b) Example negotiation scenario

Fig. 2: Message structure and negotiation example

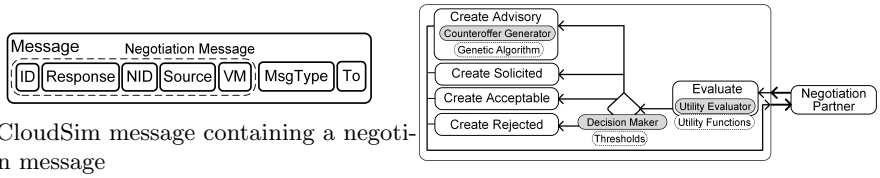
For the simulation environment we created tags representing the offer states defined in the WS-Agreement Negotiation specification.

Precondition for running negotiations on CloudSim is the exchange of offers. Therefore, the message content field cloud be used. A sender has to add its offer to the field $MsgContent$, set the type of message and set the destination using the *to* field. As CloudSim adds the entity id of the sender to CloudSim messages, the receiver is able to identify the sending entity - also termed source. Hence, entities could also negotiate in parallel as figure 2b shows. Here, broker 2 negotiates with two datacenters in parallel. Entity 3 uses the added source field to distinguish between the two brokers. However, if a entity requires virtual machines e.g. for two different systems then it has two negotiations in parallel with another entity such as a datacenter. In such a case e.g. broker 3 is able to distinguish between the different entities using the source field. However, it is unable to distinguish between different negotiations with the same entity. This issue underpins the need of a negotiation id. So instead of adding offers directly to the message content field we suggest to use an intermediary object of type $NegotiationMessage$ which is added to the field $MsgContent$. The most important fields are shown in figure 3a.

- Each negotiation message has an unique id (UDDI)
- Each negotiation message stores a reference to the preceding negotiation message (if existing)

[5] Cloud Information System - an internal CloudSim component

– The source field represent the id of the entity which created the negotiation message It is part of the negotiation message to simplify it's processing.
– In the field VM the offered virtual machine is stored



(a) CloudSim message containing a negotiation message

(b) High level strategy

Fig. 3: Negotiation message and summary of the high level strategy

Negotiation Manager. The negotiation manager is responsible for two tasks: forwarding negotiation messages and creating new negotiations. An example is depicted in figure 4. The CloudSim framework uses the entity id of the destination (field to) for forwarding messages. In the illustrated example, CloudSim forwards the message to the entity with the id 1. A negotiation entity - an entity which we introduced with our simulation environment - passes its messages to its negotiation manager. The negotiation manager checks, if the received message is a negotiation message (as described before). In such a case, the negotiation manager access the negotiation message stored in the received message and forwards it to the corresponding negotiation. If the negotiation does not exist, then the negotiation manager has to create one. In cases in which the received message is not a negotiation message then the negotiation manager ignores it.

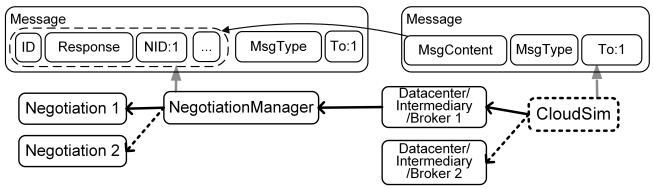


Fig. 4: Forwarding negotiation messages to the corresponding negotiation

Negotiation. The negotiation component acts as a container which stores the negotiation id as well as the negotiation history. Further, it has a reference to the used negotiation strategy - see next paragraph. The negotiation strategy creates offers which it forwards to the negotiation component, which forwards

the offers to the negotiation manager. The negotiation manager adds the source to the negotiation message and forwards the message to the CloudSim framework which delivers the offer.

Strategy. The simulation environment simulates Cloud markets. Each market participant uses a negotiation strategy. The strategy is responsible for deciding how to repose to received offers. The user of the introduced simulation environment is responsible for creating them and assigning the strategies to the market participants. For test purposes, we implemented initial negotiation strategies which follow the high-level strategy process illustrated in figure 3b. The figure contains dark boxes as well as dashed boxes. Former are components of the strategy which are shown in figure 1b. Latter will be discussed in the following use case section. The process starts with the collection of received offers. Then, these offers are ranked. Thereby, utility functions such as described in [23] are used[6]. The ranking of the offers is the precondition for making decisions - in figure 3b the *decision maker* decides if an offer is rejected or if a counteroffer (an advisory message) is created. In all the other cases offers in the states *acceptable* and *solicited* will be created. CR2

4 Simulation Environment

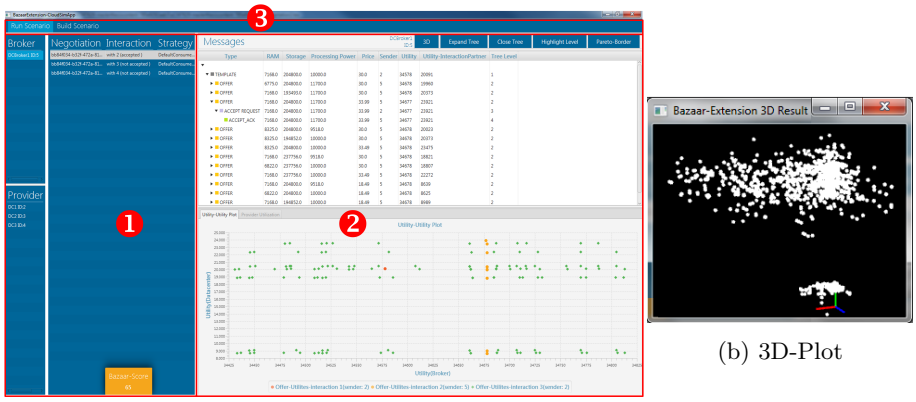
CloudSim supports the simulation of technical algorithms such as allocation algorithms which map physical resources to virtual resources (time-shared, space-shared) and VM placement algorithms (which determine which host runs which virtual machine). The negotiation process is usually executed before VMs are placed on datacenters. Free capacities - which is determined by the used technical algorithms - might be considered by negotiations strategies during negotiations . CR3
For our simulation environment we developed a result view. Figure 5a depicts the structure of it. The numbers in the figure represent three sections. The main section is section 3 which composes the menu bar as well as the other two sections. Section 1 shows the participants of the simulated market. By selecting a negotiation of a market participant its negotiation details are loaded into section 2 which encompasses of two visualizations:

- The exchanged offers of a negotiation are visualized in a tree list. Each offer contains a description of a virtual machine - it's characteristics are shown in in the rows of the tree list.
- The utility-utility plot visualizes the tree list. As figure 7a shows, the ordinate shows the utility of the offers for the selected market participant (from section 1) while the abscissa shows the utility of the offers of the negotiation partner. The points in the plot represent the exchanged messages which contain the offered VMs. The utility evaluator is responsible for assigning utility values to offers. The different colors indicate in which iteration an offer was

[6] Utility values represent the satisfaction experienced by an individual from a good - for more information see [23]

exchanged. So e.g. the negotiation starts with an initial offer represented by a red point. The negotiation partner responds to this offer with counteroffers which are visualized by the green points. It is possible to calculate the Pareto-border which visualizes the efficiency of the exchanged offers.

The messages exchanged during negotiation can be further visualized using a 3D-Plot which is shown in figure 5b. Thereby, the white dots represent the offers while the axis represent the characteristics of the VMs contained in the offers. So the red axis represents the RAM, the blue axis represents the storage while the green axis represents the processing power.



(a) Screenshot of the GUI of the simulation environment based on JavaFX

(b) 3D-Plot

Fig. 5: Screenshots of the simulation environment

CR1

Evaluations such as scalability test are out of the scope of this paper. Scalability tests for CloudSim are published in [6].

5 Use Case

In this section we summarize a consumer-provider negotiation scenario. Following the high level strategy depicted in figure 3b three components are necessary for a negotiation strategy. The dashed boxes show how we implemented these components for the use case. For the use case we reverted to utility functions developed in [23]. They are depicited in table 1. The min/max values are part of the utility functions introduced in [23]. Usually, neither consumers nor providers will publish these values and so in our simulation environment market participants can not see these values of other market participants. U is a typical example of a utility function used by a consumer where the utility value increases with additional VM resources. For the decision maker predefined utility values were used

CR5

as threshold. A genetic algorithm is used for creating counteroffers and considers the valuation of negotiation partners in the fitness function as suggested in [15]. By assuming that the utility functions of the negotiation partners are unknown the creator of a counteroffer has to estimate the utility functions used by the negotiation partners. This assumption is typical for bilateral negotiations - see e.g. [10]. The utility function $\hat{U}$ in table 1 represents an estimation of a typical utility function which is used by a provider and which represents the profit contribution. For example 0.001 are the estimated costs for one MB RAM. The most important parameters are summarized in table 1. With a focus on demonstrating the described genetic algorithm we assumed these parameters . In the paper at hand we focus on the genetic algorithm due to the strict page limit.

CR9

CR5

We describe VM characteristics using a vector (x_1, x_2, x_3, x_4). The first element (x_1) represents the storage (GB), the second element the processing power (MIPS), the third element RAM (GB) and the last element the price ($). A genetic algorithm has a Population, a Fitness Function as well as Crossover and Mutation operations which are summarized in the following. The individuals generated by the genetic algorithm represent potential counteroffers.

Population. The population of the genetic algorithm consists of vectors representing VMs. These VMs are the individuals. There are two basic options for creating the initial population.

1. The received offer is ignored for population generation. So the initial population is created randomly.
2. Usually, a received offer has high utility for its sender. Hence the received offer is used for creating the population and consequently counteroffers.

Individuals resulting from a random created initial population using option 1. may have no utility for the negotiation partner as described in the next paragraph. Hence we decided to create the population by using option 2. based on the received counteroffer: Depending on the population size, different variations of the received offer are created which form the initial population. An individual is created by modifying one of the four characteristics of a VM. For example, an individual differs in price from received offer whereas another individual differs in storage from the received offer. Characteristics are increased as well as decreased. Due to crossover and mutation operations the offers created by the genetic algorithm have less similarity with the received offer. But the result is more similar to the received offer than it would be by using a random initial population.

Fitness Function. The used fitness functions for counteroffer creation have a twofold goal: they represent the utility for the sender as well as the receiver. This is reflected by the fitness function which has two components. The first component represents the utility function used by the sender and the second component is an estimated utility function which represents the utility function of the negotiation partner. The estimated utility function may be generated using genetic programming techniques. Techniques for creating estimated utility

functions are part of our further research. In the paper at hand the estimated utility function $\hat{U}$ was defined by us to illustrate the mechanism of the negotiation strategy. Individuals having a high fitness value usually have a high utility for both consumer and provider. This increases the probability that both, sender and receiver will accept the offer which decrease the time of negotiation. The estimated utility function representing the utility of the negotiation partner is imprecise. Hence, an offer with a high fitness value may be not acceptable for the negotiation partner because the high fitness value may result from the imprecise estimated fitness function. Therefore, we limited the initial population creation by using option 2. to keep the counteroffer closer to the received offer. The received offer is used as guideline for counteroffer creation. This reduces the risk of creating offers with high fitness values and low utility for the receiver. The structure of the used fitness functions are shown in equation 1. The fitness function $F_{consumer}$ is used by consumers. It considers its utility function as well as an estimated utility function of the provider $\hat{U}_{provider}$. Similarly, the fitness function used by the provider considers its utility function and an estimated utility function of the consumer $\hat{U}_{consumer}$. The estimated utility functions have to be weighted with weight w for an adequate share. The higher the weight, the stronger is the consideration of the negotiation partner. Therefore, w is called *consideration factor*. For the scenarios we have pre-defined the size of w. In our simulation environment w could also be calculated dynamically.

CR9

CR8

CR9

CR9

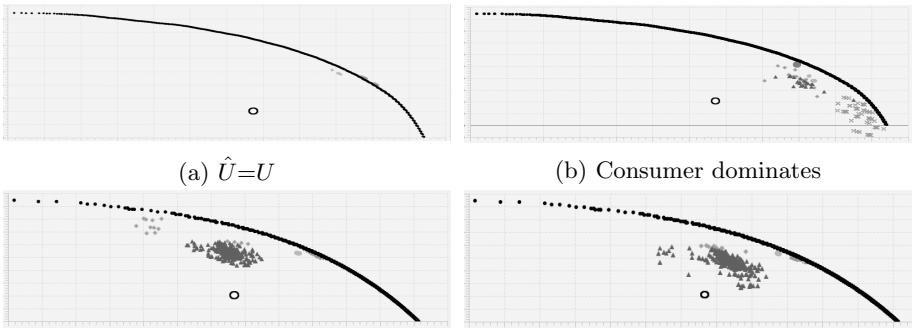
$$F_{consumer} = U_{consumer} + \hat{U}_{provider} \cdot w, F_{provider} = U_{provider} + \hat{U}_{consumer} \cdot w \quad (1)$$

Crossover and Mutation. The creation of a new generation consists of two steps. (i) Elitism is used for creating a part of the new generation by putting the best individuals regarding fitness of the old generation to the new generation. (ii) The other individuals are generated using crossover and mutation operations. A Roulette Wheel Selection is used for parents selection needed during the crossover operation. Thus, the parent selection probability is proportional to the fitness of an individual:

$$P_i = \frac{F_i}{\sum_{n=0}^{p} F_n} \quad (2)$$

P_i is the selection probability of an individual i, F_i is the fitness of the individual i and p is the population size. After the selection of two parents an individual is created by taking randomly two characteristics of the first parent and the other characteristics of the other parent. The new generated individual is mutated with a certain probability by modifying one of its characteristics.

The best offers created by the algorithm are used as counteroffers. In figure 6 some negotiation examples using different consideration factors are shown. In all graphs the ordinate represents the utility of the provider and the abscissa represents the utility of the consumer. The initial offer is represented by a white point with a black border and the message exchanged between consumer and provider are visualized by grey points. In all figures the black points forming a border represent an approximation of the Pareto-border. Messages on that



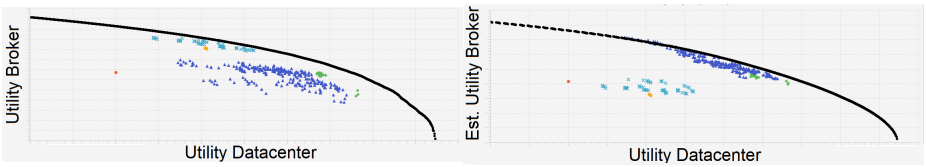
(a) $\hat{U}$=U (b) Consumer dominates

(c) Providers dominates (d) Balanced negotiation

Fig. 6: Screenshot of the result view of the simulation environment

border are Pareto-optimal. In figure 6c the consumer uses a high consideration factor while in figure 6b the provider uses a high consideration factor. In the negotiations 6a and 6d consumer as well as provider use a moderate consideration factor. Consumer and provider use real utility functions ($\hat{U} = U$) in the negotiation depicted in figure 6a. So almost all points are on the Pareto-border. CR9

The simulation environment supports two utility-utility plots. In the one depicted in figure 7a the ordinate represents the utility of the consumer while the abscissa represents the utility of the provider. The red dot represents the first offer with which the negotiation started. As already described, the colors of the other dots represent the negotiation round in which they were sent to the negotiation partner. The consumer created counteroffers in response to the CR7 first offer to which the provider responded with messages represented by green points. After the counteroffers were received, the provider responded to them with counteroffers visualized as blue triangles. A lot of offers have a great distance to the Pareto-border. The distance occurs because (i) the genetic algorithm calculates approximations and (ii) the estimated utility function $\hat{U}$ is imprecise. CR9



(a) Utility-utility plot - ordinate represents $U_{consumer}$ (b) Utility-utility plot - ordinate represents $\hat{U}_{consumer}$

Fig. 7: Two utility-utility plots

Figure 7b visualizes the perspective of the provider where the ordinate represents the estimated utility ($\hat{U}_{consumer}$) of the consumer. The offers created CR9

by the provider are approximately Pareto-optimal - from the perspective of the provider (with the estimated utility function).

Table 1: Genetic algorithm setup summary

Parameters	Values	Parameters	Values
Population Size	96	Mutation Probability	5%
Received VM	(200,10000,7,30)	Elitism	best 5%
Fitness Function			
$U_x = \begin{cases} log(x) & x \geq Min_x \\ -\infty, & x < Min_x \end{cases}$		x $\in \{RAM, Storage, Proc.Power\}$	
$U_{Price} = \begin{cases} log(Max_{Price} - Price + 1) & Price \leq Max_{Price} \\ -\infty, & Price > Max_{Price} \end{cases}$			
U=$1 \cdot U_{Price} + 1 \cdot U_{RAM} + 1 \cdot U_{Storage} + 1 \cdot U_{Proc.Power} + 100000$			
$\hat{U} = Price - RAM \cdot 0.001 - Storage \cdot 0.0005 - Proc.Power \cdot 0.001$			
$w = 25$			

With a focus on the architecture as well as on the functional capabilities of the simulation environment we neglected non-functional characteristics such as performance which we see as part of our further research. An analysis of other possible negotiation strategies was done in [25]. This survey shows that Bayess theory is heavily used for negotiation strategies. We plan to implement and compare them with the introduced strategy.

CR4

6 Conclusion and Further Research

In this paper we presented a simulation environment based on CloudSim for the simulation of Bazaar-based Cloud markets. The simulation environment is compliant to the WS-Agreement negotiation specification and in the paper we describe its architecture as well as a summary of a negotiation strategy based on a genetic algorithm. Using the simulation environment brokers, intermediaries and datacenters are created, then negotiation strategies are assigned to them before the negotiation results can be analyzed.

In our further research we will develop further components based on the simulation environment: For example taxes on Cloud Markets have not been considered yet by the scientific community as well as smart contract technology which can be used for the created SLAs. Further, novel negotiation strategies based on deep learning techniques are part of our future research.

CR4

References

1. Abuhay, T.M., Kovalchuk, S.V., Bochenina, K.O., Kampis, G., Krzhizhanovskaya, V.V., Lees, M.H.: Analysis of computational science papers from ICCS 2001-2016 using topic modeling and graph theory. In: International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland. pp. 7–17 (2017)

2. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M.: Web services agreement specification (WS-Agreement). In: Open Grid Forum. vol. 128 (2007)
3. Böhm, M., Koleva, G., Leimeister, S., Riedl, C., Krcmar, H.: Towards a generic value network for cloud computing. In: Economics of Grids, Clouds, Systems, and Services, 7th International Workshop, GECON 2010, Ischia, Italy, August 31, 2010. Proceedings. pp. 129–140 (2010)
4. Bonacquisto, P., Modica, G.D., Petralia, G., Tomarchio, O.: A Strategy to Optimize Resource Allocation in Auction-Based Cloud Markets. In: Services Computing (SCC), 2014 IEEE International Conference on. pp. 339–346. IEEE (2014)
5. Buyya, R., Ranjan, R., Calheiros, R.N.: Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities. In: High Performance Computing & Simulation, 2009. HPCS'09. International Conference on. pp. 1–11. IEEE (2009)
6. Calheiros, R.N., Ranjan, R., Beloglazov, A., Rose, C.A.F.D., Buyya, R.: Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw., Pract. Exper. **41**(1), 23–50 (2011)
7. Carrión, J., Puntes, D.F., Luque, E.: Simulating a search engine service focusing on network performance. In: International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland. pp. 79–88 (2017)
8. Chhabra, S., Dixit, V.S.: Cloud computing: State of the art and security issues. ACM SIGSOFT Software Engineering Notes **40**(2), 1–11 (2015)
9. Coles, C.: Overview of cloud market in 2017 and beyond (2017), `https://www.skyhighnetworks.com/cloud-security-blog/microsoft-azure-closes-iaas-adoption-gap`, Accessed 14-10-2017
10. Dastjerdi, A.V., Buyya, R.: An autonomous reliability-aware negotiation strategy for cloud computing environments. In: Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on. pp. 284–291. IEEE (2012)
11. Gao, G., Mishra, B., Ramazzotti, D.: Efficient simulation of financial stress testing scenarios with suppes-bayes causal networks. In: International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland. pp. 272–284 (2017)
12. Gartner: Gartner says worldwide public cloud services market to grow 18 percent in 2017. In: Gartner. Gartner (2017), `https://www.gartner.com/newsroom/id/3616417`, Accessed 14-10-2017
13. Hudert, S., Ludwig, H., Wirtz, G.: Negotiating SLAs-An approach for a generic negotiation framework for WS-Agreement. Journal of Grid Computing **7**(2), 225–246 (2009)
14. Kliazovich, D., Bouvry, P., Khan, S.U.: Greencloud: a packet-level simulator of energy-aware cloud computing data centers. The Journal of Supercomputing **62**(3), 1263–1283 (2012)
15. Ludwig, S.A., Schoene, T.: Matchmaking in multi-attribute auctions using a genetic algorithm and a particle swarm approach. In: New Trends in Agent-Based Complex Automated Negotiations, pp. 81–98. Springer (2012)
16. Mach, W.: A simulation environment for ws-agreement negotiation compliant strategies. In: Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services, iiWAS 2017, Salzburg (2017)
17. Mach, W., Pittl, B., Schikuta, E.: A Forecasting and Decision Model for Successful Service Negotiation. In: Services Computing (SCC), 2014 IEEE International Conference on. pp. 733–740. IEEE (2014)

18. Mach, W., Schikuta, E.: A generic negotiation and re-negotiation framework for consumer-provider contracting of web services. In: Proceedings of the 14th International Conference on Information Integration and Web-based Applications & Services. pp. 348–351. ACM (2012)
19. Malinowski, A., Czarnul, P., Czurylo, K., Maciejewski, M., Skowron, P.: Multi-agent large-scale parallel crowd simulation. In: International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland. pp. 917–926 (2017)
20. Mitchell, J.: What's the best way to purchase cloud services? IEEE Cloud Computing **2**(3), 12–15 (2015)
21. Núñez, A., Vázquez-Poletti, J.L., Caminero, A.C., Castañé, G.G., Carretero, J., Llorente, I.M.: icancloud: A flexible and scalable cloud infrastructure simulator. Journal of Grid Computing **10**(1), 185–209 (2012)
22. Pichot, A., Waeldrich, O., Ziegler, W., Wieder, P.: Dynamic SLA Negotiation Based on WS-Agreement. In: WEBIST (1). pp. 38–45 (2008)
23. Pittl, B., Mach, W., Schikuta, E.: A Negotiation-Based Resource Allocation Model in IaaS-Markets. In: 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC). pp. 55–64. IEEE (2015)
24. Pittl, B., Mach, W., Schikuta, E.: Bazaar-extension: A cloudsim extension for simulating negotiation based resource allocations. In: IEEE International Conference on Services Computing, SCC 2016, San Francisco, CA, USA, June 27 - July 2, 2016. pp. 427–434 (2016)
25. Pittl, B., Mach, W., Schikuta, E.: A classification of autonomous bilateral cloud SLA negotiation strategies. In: Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services, iiWAS 2016, Singapore, November 28-30, 2016. pp. 379–388 (2016)
26. Pittl, B., Mach, W., Schikuta, E.: An implementation of the ws-agreement negotiation standard in cloudsim. In: 20th IEEE International Enterprise Distributed Object Computing Workshop, EDOC Workshops 2016, Vienna, Austria, September 5-9, 2016. pp. 1–4 (2016)
27. Rumpl, A., Waeldrich, O., Ziegler, W.: Extending WS-Agreement with multi-round negotiation capability. In: Grids and Service-Oriented Architectures for Service Level Agreements, pp. 89–103. Springer (2010)
28. Samimi, P., Teimouri, Y., Mukhtar, M.: A combinatorial double auction resource allocation model in cloud computing. Information Sciences (2014)
29. Silaghi, G.C., Erban, L.D., Litan, C.M.: A time-constrained SLA negotiation strategy in competitive computational grids. Future Generation Computer Systems **28**(8), 1303–1315 (2012)
30. Waeldrich, O., Battr, D., Brazier, F., Clark, K., Oey, M., Papaspyrou, A., Wieder, P., Ziegler, W.: Ws-agreement negotiation version 1.0. In: Open Grid Forum (2011)
31. Weinman, J.: Cloud pricing and markets. IEEE Cloud Computing **2**(1), 10–13 (2015)
32. Weinman, J.: Migrating to-or away from-the public cloud. IEEE Cloud Computing **3**(2), 6–10 (2016). https://doi.org/10.1109/MCC.2016.45
33. WSAG4J: Wsag4j, `http://wsag4j.sourceforge.net/site/index.html`
34. Yan, J., Kowalczyk, R., Lin, J., Chhetri, M.B., Goh, S.K., Zhang, J.: Autonomous service level agreement negotiation for service composition provision. Future Generation Computer Systems **23**(6), 748–759 (2007)