

LOUD SERVICE NEGOTIATION IN INTERNET OF THINGS ENVIRONMENT: A MIXED APPROACH

Satya Veni Matta¹, Dr.Y.Venkateswarulu², Dr U Ravi Babu³

¹Mtech Student, CSE, Giet Engineering College, Rajahmundry, A,P, India
Professor and HOD, Dept of CSE, Giet Engineering College, Rajahmundry, A,P, India
Professor, Dept of CSE, MREC(A)

ABSTRACT

Internet of Things (IoT) is allowed to communicate the connected objects via the Internet. IoT can benefit from the boundless capabilities and resources of cloud computing. Due to the cloud market becomes more competitive and open, Quality of Service (QoS) will be more important. However, cloud providers and cloud consumers have different, and sometimes opposite, preferences. To balance effectiveness and accomplishment rate, the present paper propose a mixed approach for cloud service negotiation, which is based on the “game of chicken”. To evaluate the effectiveness of this approach, experiments are conducted on extensive simulations. Results show that a mixed negotiation approach can achieve a higher utility than a concession approach.

INTRODUCTION

Internet of Things (IoT) allows objects like computers, sensors, mobile phones, etc. to communicate via the Internet. IoT is expected to be a worldwide network of interconnected objects [1]. It is characterized by limited capacities and constrained devices, and its development depends on new technologies including cloud computing. IoT can benefit from the unlimited

capabilities and resources of cloud computing. Also, when coupled with IoT, cloud computing can in turn deal with real world things in a more distributed and dynamic manner.

Cloud services are easier to access and use, cost-efficient, and environmentally sustainable. As they eliminate large upfront expenses in hardware and expensive labor costs for maintenance, cloud services are beneficial to small- and medium-sized enterprises. Cloud services are Internet-based IT services. Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) are three representative examples [2], [3]. However, cloud providers and cloud consumers have different and sometimes opposite preferences. For example, a cloud consumer usually prefers a high reliability, whereas a cloud provider may only guarantee a less than maximum reliability in order to reduce costs and maximize profits. If such a conflict occurs, a Service Level Agreement (SLA) cannot be reached without negotiation.

To create a proposal, a negotiation agent can adopt two strategies i.e. concession and tradeoff. However, if information is incomplete, it may cause miscalculations, and so underperform the concession one in

terms of success rate. To balance utility and success rate, in this paper propose a mixed approach for cloud service negotiation, which is based on the “game of chicken.” In other words, if a party’s counterpart uses a concession strategy, it is best to adopt a tradeoff one; if a party’s counterpart uses a tradeoff strategy, it is best to adopt a concession one; and if a party is uncertain about the strategy of its counterpart, it is best to mix concession and tradeoff. In fact, those are the three Nash equilibrium of a negotiation game with two pure strategies. A mixed negotiation approach based on the “game of chicken,” which can balance utility and success rate. In particular, if a party has no knowledge of which strategy that its counterpart will play, it is best to mix concession and tradeoff in negotiation to evaluate the effectiveness of the mixed negotiation approach. We first test the impact of different parameters on negotiation results and then conduct Monte Carlo simulations. Results show that the mixed negotiation approach can achieve a higher utility than a concession approach, while incurring fewer failures than a tradeoff approach, which demonstrates its effectiveness.

The rest of the paper is structured as follows. Section II describes multi-attribute bilateral negotiations where concession and tradeoff strategies are detailed and proposes a mixed approach for cloud service negotiation, which is based on the “game of chicken.” Section III results and discussions analyzes results. Conclusion are given in section IV

Section II: Proposed Approach

To balance usefulness and accomplishment rate, the present propose a mixed negotiation approach for cloud service negotiation, which is based on the “game of chicken.”

Two-Player Negotiation Game

In a negotiation game, a selfish agent’s utility remains the same with a tradeoff strategy, whereas its utility is decreased with a concession one. As the agent attempts to maximize its utility, it seems that it should stick to the tradeoff strategy instead of the concession one. If the agent and its counterpart both adopt the tradeoff strategy, unfortunately, it is very likely that a failure happens, whereupon both receive the worst utility. It thus becomes a dilemma. This indicates that how to play concession and tradeoff strategies is of utmost importance. However, to the best of our knowledge, no previous work deals with this problem. In fact, we first identify the problem and model it with the “game of chicken,” which goes as follows [2]. Two boys, say Alan and Bob, want to prove their manhood. They drive toward each other at breakneck speed. The one who swerves loses face and becomes a “chicken,” whereas the other who stays, of course, proves his manhood and becomes a hero to his friends. If both swerve, nothing is proved. If neither swerves, they crash into each other with potentially disastrous results.

A possible payoff matrix of the game of chicken is shown in Table II, where a number only has a relative significance, namely, the greater the number, the higher the payoff. A Nash equilibrium is “a situation in which each player in a game chooses the strategy that yields the highest payoff, given the strategies chosen by the other players” [4]. The “game of chicken” has two pure strategy Nash equilibrium. One is for Alan to swerve and for Bob to stay, whereas the other is for Alan to stay and for Bob to swerve. In fact, if Alan swerves, Bob is better off staying (payoff 1) than swerving (payoff 0). Conversely, if Alan stays, Bob is better off

swerving (payoff -1) than staying payoff -10). So, those are the two pure strategies Nash equilibrium. Below, we give a formal description for Nash equilibrium [5].

Definition 1 (Nash Equilibrium): A Nash equilibrium is a strategy profile

$s^* = (s^*_1, s^*_{-1})$ such that each player, $i(i=1,2,3,\dots,n)$, has no incentive to deviate from its current strategy, s^*_i , given the strategy profile, s^*_{-i} , of the other players.

A general payoff matrix of a two-player negotiation game with concession and tradeoff strategies is shown in Table III, where $a_1, a_2, b_1, b_2, c_1, c_2, d_1, d_2$ belongs to \mathbb{R} and $a_1 \geq a_2 > b_1 \geq b_2 > c_1 \geq c_2 > d_1 \geq d_2$. It should be noted that, here, the game is asymmetric, in that the two players are distinguishable from each other, and is more applicable, in that it generalizes the “game of chicken.”

Proposed Algorithmic Description:

Algorithm 1 implements a mixed negotiation approach. It works as follows. First, in line 1, agent i sends V —its initial proposal—to agent j , and waits for a response. If j does not accept V and j 's counter proposal is not acceptable to i , then i adopts a mixed approach in the while loop of lines 2–15 to create a new proposal; otherwise, true is returned in line 16. Here, a party's acceptance criterion is that the utility received from a proposal is no less than that of its reserved proposal, and the values received from the proposal do not go beyond its reserved values.

Next, in line 4, uses function random to generate a random number between 0 and 1 for variable r . In lines 5–10, if $r < 1-p$, which implies that a concession strategy is triggered, i uses function concession to create a new proposal, where $P_r \{ r < 1-p \} = 1-p$. In

line 6, is increased by one, each time the condition is triggered. It should be mentioned that concession is a function that implements a concession strategy of a multi-attribute negotiation. Refer to [30] for its algorithmic description. If $r \geq 1-p$, which implies that a tradeoff strategy is triggered, uses function tradeoff to create a new proposal, where $P_r \{ r \geq 1-p \} = p$. In line 9, k_2 is increased by 1, each time the condition is triggered. The Mixed approach algorithm is shown below

Algorithm: Mixed Approach ($V, W, F, \lambda_1, \lambda_2, p$)

Input: array V with raw values of n attributes

array W with weights of n attributes

array F with flags of n attributes

A flag indicates whether an attribute is higher-is-better

parameters λ_1 and λ_2 ($0 < \lambda_1, \lambda_2 < 1$) which indicate the rate of concession and the rate of tradeoff at a time, respectively;

parameter p ($0 < p < 1$) which indicates the probability of playing tradeoff, or p -for short

Output: true if succeed and false otherwise

Step 1: agent sends V to agent and waits for a response

Step 2: while agent j does not accept V and j 's counter proposal is not

Step 3: acceptable to agent i

Step 4: $r = \text{random}(0,1)$

Step 5: if r is less than $1 - p$ then

Step 6: $k_1 = k_1 + 1$

Step 7: $V = \text{concession}(V, W, F, k_1, \lambda_1)$

Step 8: otherwise

Step 9: $k_2 = k_2 + 1$

Step 10: $V = \text{tradeoff}(V, W, F, k_2, \lambda_2)$

Step 11: $k = k_2 + k_1$

Step 12: if V is out of bounds then

Step 13: return FALSE

Step 14: otherwise

Step 15: agent i sends V to agent j and waits for a response

Step 16: return TRUE

It should also be mentioned that tradeoff is a function that implements a tradeoff strategy of a multi-attribute negotiation. Refer to [6] for its algorithmic description. In line 11, k counts the total number of negotiation rounds.

Finally, in lines 12–15, if V is out of bounds, false is returned; otherwise, agent sends, whose values are adjusted, to agent j as a new proposal, and waits for a response again. The process repeats until either success or failure occurs. In this process, utility of the current proposal can remain the same (moves along its current indifference curve) or be reduced (moves down to its next indifference curve). It can be proved that Algorithm 1 converges and terminates in a

finite number of rounds. Refer to [6] for the proof.

It should be noted that the mixed approach we adopt in negotiation exhibits a certain degree of intelligence. Just as Turing [7] pointed out in Computing Machinery and Intelligence, “Intelligent behavior presumably consists in a departure from the completely disciplined behavior involved in computation, but a rather slight one, which does not give rise to random behavior, or to pointless repetitive loops.”

EVALUATION AND ANALYSIS

We conduct extensive simulations to evaluate the mixed approach for cloud service negotiation. First, we describe the experimental setup. Next, we describe the parameter setup. Finally, we report and analyze simulation results.

Experimental Setup

All simulations are conducted on a Lenovo Think Centre desktop with a 2.80-GHz Intel Pentium Dual-Core CPU and a 2.96-GB RAM, running Microsoft Windows 7 Professional Operating System. The simulations are implemented with Java under NetBeans IDE 7.2.1 with JDK 7u13. An alternating-offers protocol is adopted as the negotiation protocol, and a mixed negotiation strategy is compared with concession and tradeoff strategies. The negotiation process works as follows. First, without loss of generality, a SP sends its initial proposal to a SC. Next, if the proposal is accepted by the SC, negotiation ends successfully; otherwise, the SC uses either mixed, tradeoff, or concession negotiation approach to create a counter proposal. After that, the SC sends back the counter proposal to the SP, and the negotiation process repeats. The

process ends once a proposal or a counter proposal is accepted, and it fails if no proposal is acceptable to both parties. Java multithreading, which allows multiple tasks in a program to be executed concurrently, is the ideal technique to simulate the negotiation process. A thread is the flow of execution, from beginning to end, of a task. We model the behaviors of the SP and the SC as two threads. In particular, we use thread synchronization techniques to coordinate their behaviors, and a shared object to exchange their proposals and counter proposals. In our software prototype, there is a QoS matrix to be negotiated, where the SP and the SC can specify their QoS requirements, i.e., their preferred values, reserved values, and weights over quality dimensions AVAIL, REL, RESP, SECY, and ELAS. In a real negotiation, those values would be kept private.

TABLE III

TWO-PLAYER NEGOTIATION GAME

		Player 2	
		Concession	Tradeoff
Player 1	Concession	b_1, b_2	c_1, a_1
	Tradeoff	a_2, c_2	d_1, d_2

In our simulations, we attempt to resolve QoS conflicts in the motivating example. In other words, we use Table I as the QoS matrix to be negotiated. Also, there is a parameter variance $\delta (0 < \delta < 1)$ that can be used to generate a random number within a certain interval of a value, such that the impact of a specific data set on negotiation results can be reduced, if not completely removed.

As to negotiation strategies, the SP and the SC can choose a concession, a tradeoff, or a mixed approach. So, in total, there exist nine combinations, i.e., CC, CT, CM, TC, TT, TM, MC, MT, MM, respectively, where

stands for a concession approach, a tradeoff one, and a mixed one. Also, there are parameters the rate of concession α , the rate of tradeoff β , and the probability of playing tradeoff. As to negotiation results, success or failure can happen. In the case that success occurs, QoS conflicts are resolved, and the new values agreed to by the SP and the SC are output. In the case that failure happens, relevant information about the failure is output. Also, there is parameter tolerance ϵ , within which a solution whose values go beyond a party's reserved values, but its utility is no less than its reserved utility is still acceptable. So, a rigid cutoff value is avoided, and the chance of success increased. In fact, it is the acceptance criterion that we adopt in our simulations. To fully understand the impact of different parameters on negotiation results, we conduct a series of simulations where the SP and the SC both adopt a concession, a tradeoff, and a mixed approach, respectively. Refer to [30] for those parts. We keep, without further details, in our simulations. Unless specified otherwise, we keep the gap in preferred values as 0.11, 0.20, 0.60, 0.05, and 0.06, and the gap in reserved values as 0.19, 0.10, 0.20, 0.15, and 0.14 for AVAIL, REL, RESP, SECY, and ELAS, respectively, in our simulations. Also, we set the maximum negotiation round, as most negotiations can finish in no more than 20 rounds. In our simulations, agents' preferences are kept private so we cannot apply Theorem 2 here. However, it does give us some hint on how to choose. As a general rule, if competition is high, a small value is preferred; otherwise, a large value is preferred.

CONCLUSIONS:

A tradeoff approach can outperform a concession one in terms of utility, but may incur more failures if information is

incomplete. To balance utility and success rate, we propose a mixed approach for cloud service negotiation, which is based on the “game of chicken.” In particular, if a party is uncertain about the strategy of its counterpart, it is best to mix concession and tradeoff strategies. In fact, it is a mixed strategy Nash equilibrium of a negotiation game with two pure strategies, which provides the theoretical basis for our approach. To demonstrate the effectiveness of the mixed approach, we conduct extensive simulations. Results show that when a party has no knowledge of the strategy of its counterpart, a mixed approach outperforms a concession one in terms of utility, and it outperforms a tradeoff one in terms of success rate. It should be noted that the mixed approach works under incomplete information, and so is applicable for real negotiations, where information is generally not complete. In conclusion, when one is uncertain about the strategy of its counterpart, a mixed negotiation approach, which exhibits a certain degree of intelligence, can achieve a higher utility than a concession one, while incurring fewer failures than a tradeoff one. It thus becomes a promising approach for cloud servicenegotiation.

REFERENCES:

- [1] Q. Li et al., “Applications integration in a hybrid cloud computing environment: Modelling and platform,” *Enterpr. Inf. Syst.*, vol. 7, no. 3, pp. 237–271, 2013
- [2] M. Armbrust et al., “A view of cloud computing,” *Commun. ACM*, vol. 40, no. 4, pp. 50–58, 2010.
- [3] L. M. Vaquero et al., “A break in the clouds: Towards a cloud definition,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 50–55, 2009
- [4] D. Besanko and R. R. Braeutigam, *Microeconomics*, 3rd ed. Hoboken, NJ, USA: Wiley, 2008.
- [5] K. Leyton-Brown and Y. Shoham, *Essentials of Game Theory: A Concise, Multidisciplinary Introduction*. San Rafael, CA, USA: Morgan & Claypool, 2008.
- [6] X. Zheng, “QoS Representation, Negotiation and Assurance in Cloud Services,” Ph.D dissertation, School of Computing Queen’s Univ., Kingston, ON, Canada, Feb. 2014
- [7] A. M. Turing, “Computing machinery and intelligence,” *Mind*, vol. 59, no. 236, pp. 2023–2033, 1950.