To compete or cooperate? This is the question in communities of autonomous services

Ehsan Khosrowshahi Asl a, Jamal Bentahar a,⇑, Rabeb Mizouni b, Babak Khosravifar c, Hadi Otrok b

a Faculty of Engineering & Computer Science, Concordia University, 1515 Ste-Catherine Street West, EV7.640, Montreal, Quebec, Canada H3G 2W1
b College of Engineering, Khalifa University of Science, Technology, & Research, Abu Dhabi, United Arab Emirates
c Faculty of Science, McGill University, Dawson Hall, 853 Sherbrooke St. West Montreal, Quebec, Canada H3A 0C5

Abstract

Frameworks for aggregating similar services into structures called communities have been recently advocated. A common assumption in those frameworks is that residing services are coopetitive, i.e., competing over received requests, but also cooperating, for instance in terms of substituting each other. In this coopetition context, deciding to compete or cooperate at different moments in time is an open question yet to be addressed. The contribution of this paper is the answer to this challenging question by proposing a game-theoretic-based decision mechanism that services can use to effectively choose competition or cooperation strategies that maximize their payoffs. To achieve this objective, we investigate autonomous services’ characteristics and their expected utilities over different strategies. We propose a game-theoretic best response technique to measure the threshold that services can use in order to decide about the two strategies. We prove that the proposed decision mechanism is efficient and can be implemented in time linear in the length of the time period considered for the analysis and the number of services in the community. Moreover, we conduct extensive simulations to analyze various scenarios and confirm the obtained theoretical results. Those results show that our model outperforms existing competitive and random coopetitive strategies and the more services deviate from our game-theoretic-based coopetitive strategy the more they make less benefits.

© 2014 Elsevier Ltd. All rights reserved.

1. Introduction

Online autonomous agent-based services are software systems providing a standard way of interoperability among different applications. These services are developed on top of intelligent agents, which continuously and autonomously interact with each other to fulfill users’ requests. Organizing such services within virtual structures called communities has been proposed for the purpose of making their discovery easier and increasing the likelihood of their cooperation (Benatallah, Sheng, & Dumas, 2003; Maamar, Subramanian, Thiran, Benslimane, & Bentahar, 2009; Medjahed & Atif, 2007; Medjahed & Bouguettaya, 2005). The idea behind communities is to gather services having similar functionalities but distinct non-functional properties. In that context, communities are providing pockets of expertise and networks of services able to cooperate by substituting each other in case of execution problems (Bentahar et al., 2008; Maamar et al., 2009; M’hamdi & Bentahar, 2012).

1.1. Motivations

Abstracting and associating services1 with knowledge-empowered agents without changing these services implementation has already been proposed and widely analyzed (Garcia-Sanchez, Valencia-Gariaa, Martinez-Bejarb, & Fernandez-Breis, 2008; Lomuscio, Qu, & Solanki, 2012; Maamar, Moustefaoui, & Yahyaoui, 2005; Paschke & Boley, 2011; Pradhan & Lu, 2007). Such an agent-based abstraction benefits services from advanced interaction and decision-making techniques that those agents are able to manage (Jacyno, Bullock, Luck, & Payne, 2009; Bentahar, Khosravifar, Serhani, & Alishahi, 2012; Sanchez-Anguix, Valero, & Garcia-Fornes, 2011). That means services are no more considered as simply passive components but as intelligent entities that enjoy autonomy and selfishness, two significant properties in business settings where competition is

1 In this paper, the terms service, agent-based service, and service agent have the same meaning and are used interchangeably.
a key factor (Jurca & Faltings, 2005; Khosravifar, Bentahar, Clacens, Goffart, & Thiran, 2011). Within communities, those services, selfish and utility maximizers by nature, can follow two different strategies, namely cooperation and competition in order to increase their payoffs when they provide services to consumers (Maamar, Thiran, & Bentahar, 2011). In typical business settings, services are used to compete within communities as they provide the same functionalities and the number of users requests is finite. However, the same reason of providing similar functionalities can lead services to cooperate because they can replace each other in case of failure or unavailability, and services can do better in a coalition structure. Analyzing services competition and cooperation strategies within communities is still an open problem that motivates the research described in this paper.

1.2. Contributions

Analyzing services acting behaviors in such a context in terms of deciding which strategy to choose is an open and challenging problem. Our main contribution in this paper is to address this problem using the game-theoretic best response technique (Fudenberg & Tirole, 1991). Best response is the strategy that produces the most favorable outcome for a player, taking other players’ strategies as given. The framework we propose in this paper enables services to cope with strategic decision making in different scenarios. More precisely, in this paper we define and analyze the properties of different key factors and parameters, such as a new metric called growth factor aiming to measure the performance of services taking into account their reputation and QoS. These factors are then used to compute a threshold, called coopetition threshold, which helps service agents decide whether to compete or cooperate based on the expected payoff.

A preliminary short version of this paper is presented in Khosravifar et al. (2012) where the need of analyzing coopetition strategies of communities of autonomous services has been identified and the general architecture has been introduced. However, this paper extends (Khosravifar et al., 2012) by (1) considering more parameters (20 instead of only 5) including cooperation and competition fees and probabilities, which makes the system comprehensive; (2) using game theory to analyze the strategies and characterize the payoffs, which makes the mathematical and theoretical apparatus richer; (3) introducing and formalizing the concept of coopetition threshold, which allows services to decide about their strategies; (4) analyzing the computational complexity of the decision procedure, which shows the feasibility and efficiency of the approach; and (5) extending the simulation results and analysis by considering more scenarios.

In a nutshell, we propose a mechanism within which service agents in the community could choose either to compete for an announced task, or cooperate with other competing services in the same community to accomplish some subtasks of the announced task. We explore details behind the strategic decision making procedures and enable service agents to apply different techniques to constrain high efficiency and obtain the maximum utility. We investigate services’ expected payoffs and the involved probabilities that are used to choose over the two interacting strategies. As intelligent entities, service agents require a reasoning technique that enhances their abilities over best acting strategies and the attitude they could exhibit to yield maximum utility.

In this area of flight booking and having parameters extracted from a real dataset (Al-Masri & Mahmoud, 2007). This dataset represents 2507 real services that exist on the web. It includes the QoS values of 9 parameters including availability, throughput and reliability. These QoS values were determined by monitoring the services over a 6 day period. We equip some of those services with our proposed strategic decision making procedure and compare the performance of the equipped services against other ordinary services. We provide detailed discussions over the implemented environment and verify the effectiveness of the proposed mechanism.

1.3. Paper organization

The rest of the paper is organized as follows. Section 2 presents some preliminaries about the system design needed to understand the proposed model. Section 3 introduces the theoretical model along with services’ competitive and cooperative strategies. Section 4 outlines the experimental results and a detailed comparison of the equipped services with our strategic reasoning techniques with the ordinary services. Discussion of relevant related work is presented in Section 5. Finally, Section 6 concludes the paper.

2. The proposed framework

In this section, we first present the architecture of the proposed model. We explore the characteristics of intelligent service agents and their network. We link this architecture to the implemented system where we investigate the services’ coopetitive attitudes. We compute the involved system parameters and explain the services’ interactive strategy profiles by highlighting their coopetitive choices.

2.1. The architecture

The proposed system consists of three types of autonomous entities with different goals (Khosravifar et al., 2012):

(1) Services that reside inside a community which aggregates a number of functionally similar or complementary services as a group (more details about communities of services are given in Khosravifar, Bentahar, Moazin, & Thiran, 2010a; Maamar et al., 2009). Within the same community, each service might have a network consisting of some other services that might get involved in a cooperative work (e.g., composition and substitution). As services are also competing, particularly when they provide similar functionalities, each one of them aims to maximize its individual income (i.e., the payoff) by adopting a given strategy.

(2) Master Service is the manager and representative of the community of services. Among other functionalities, the master agent is responsible for allocating the tasks to services within the community. After the task being accomplished, and based on the delivered quality of service, the master rewards or penalizes the associated service agent by updating its reputation. The master is equipped with a task allocation mechanism aiming to increase service users satisfaction and eventually the community’s market share in the whole system.

(3) Users generate tasks with specified QoS. In our proposed system, tasks are continuously being generated and user satisfaction is abstracted since we focus on services’ interactive strategies.

The implemented environment includes the QWS dataset by Ey hab Al-Masri and Qusay H. Mahmoud freely available at: www.uoguelph.ca/qmahmoud/qws.

2 Requests and tasks are used in this paper interchangeably.
Fig. 1 illustrates the architecture of a typical community aggregating a number of services with different interactive strategies. Some of them compete for the task where they directly deal with the master. Some others cooperate in the associated task where they only deal with the competing service as the task leader and do not directly interact with the master (the master deals only with the service that has bid for the task, which is responsible of choosing its collaborative network). In both sets, some service agents are for certain moments out of any collaboration network. We highlight details of the interactive strategies in the rest of this section. In the proposed system, the master sorts the competing services (i.e., the bidders) based on some parameters (such as reputation) that we explain in the rest of this section. The winner to which the master allocates the task is the bidder service agent that is ranked first. There is a chance that some tasks could not be assigned to any service. These tasks are accumulated in the task pool to be allocated in the next task allocation round. Upon allocation of the task, the service is responsible for offering the required QoS that is stated in the task being generated by a consumer. Afterwards, the master rewards or penalizes the competing service by upgrading or degrading its reputation according to the offered QoS compared with the required one. This comparison influences the sorting mechanism used by the master to allocate the tasks in further task allocation rounds.

2.2. System parameters

In this part, we demonstrate the involved parameters and their corresponding formulations and explanations. Our proposed decision making model makes use of five main system parameters, namely Task QoS, Service QoS, Budget, Reputation and Growth Factor. Other variables needed to formalize our game-theoretic-based model are derived from those five parameters. In the following, we describe these parameters in more detail.

Task QoS \((T_{QoS}^r)\) is the required QoS metric for a specific task \(r\). Users define tasks with specific QoS requirements such as response time, availability, and successability (or accuracy) (Lim, Thiran, Maammar, & Bentahar, 2012). We aggregate and normalize these metrics to a value between 0 and 1.

Service QoS \((QoS^w)\) is the QoS provided by the service \(w\) after performing the task \(r\). Again, the metrics that contribute in computing this QoS are aggregated and normalized to a value between 0 and 1. The offered quality might or might not meet the required task quality \(T_{QoS}^r\). In the latter case, the service user would be disappointed and a negative satisfaction feedback is expected. In our proposed system, both cases are considered when calculating the services’ reputation.

Budget \((B^w)\) is the amount of money the service agent \(w\) has in its disposal during the window time \(t\) (i.e., \([0,t]\)), which helps pay for the community membership fees \((c)\) and is one of the parameters that the service agent considers when deciding about getting involved in a competition or not. This parameter has been used in other service computing settings such as Khosravifar et al. (2012) and Lim et al. (2012).

Reputation \((Rep^w)\) is a significant factor in any online community (Fouss, Achbany, & Saerens, 2010). Without a reputation enabling mechanism, users cannot differentiate among services, specially the ones which offer the same type of service. Reputation mechanisms usually aggregate users’ experiences (Khosravifar, Bentahar, Moazin, & Thiran, 2010b), and in our case it strongly depends on QoS that each service provides. Users define tasks, each one with specific quality \(T_{QoS}^r\), so that after performing a certain number of tasks, each one with QoS \(QoS^w\), during a window time \(t\), the reputation of \(w\) gets evaluated by the master agent. \(Rep^w\) refers to the reputation of \(w\) during that window time \(t\).

In Eq. (1), we compute the reward that the master computes considering the task’s QoS \(T_{QoS}^r\) compared with the service offered quality \(QoS^w\). In case the offered quality meets user expectations, the reward value would be positive. In this system, we consider a small value as default rewards \(\eta\) which the master considers together with the proportional level of satisfaction as a weighted value (by \(v\)). In this case, the higher the offered quality, the more weighted reward. In case the offered quality did not meet the user expectations, the reward would be negative. A default penalty value \(\rho\) (where \(\rho > \eta\)) together with the weighted proportional difference are therefore considered. The idea is to harshly penalize the services rather than rewarding them. To this end, rational service agents should carefully analyze their capabilities once the available tasks are announced.

Eq. (2) computes the obtained reward by \(w\) during the window time \(t\) considering the set tasks of tasks performed by \(w\) during the window time \(t\). In our proposal, service agents have the goal of increasing their budget, which is directly related to their reputation. Thus, they have to decide strategically how to maximize this value.

\[
reward_w = \begin{cases} 
\eta + v \left( \frac{QoS^w}{T_{QoS}^r} - \frac{QoS^w}{QoS^w} \right) & \text{if } T_{QoS}^r \leq QoS^w; \\
-\left( \rho + v \left( \frac{T_{QoS}^r}{QoS^w} - \frac{QoS^w}{QoS^w} \right) \right) & \text{otherwise.}
\end{cases}
\]

The assigned reputation value is updated by the computed reward value. The computed reputation of services is bounded by the minimum and maximum reputation values 0 and 1. Let \(\Gamma = Rep^w + reward^w\). The updated reputation value is then computed as follows:

\[
Rep^{w+1} = \begin{cases} 
\Gamma & \text{if } 0 \leq \Gamma \leq 1; \\
0 & \text{if } \Gamma < 0; \\
1 & \text{if } \Gamma > 1.
\end{cases}
\]

For new services with no previous reputation value, we use the bootstrapping trust technique proposed in Yahyaoui and Zhioua.
This technique consists in giving the new services a chance and observe their behaviors for a period of testing time. The observation sequence is modeled as a hidden Markov model that is used to detect the behavior of the service by comparing the observation behavior against pre-defined trust patterns. Based on the matching result, an initial value is assigned to the service. Using this initial reputation value, services quickly converge to their actual and stable values using the update function.

**Proposition 1.** \( \text{Rep}^t_w \) can be computed in time \( O(|t|) \), i.e., in time linear in the size of the window \( t \).

**Proof.** The function \( \text{Rep}^t_w \) is recursive on \( t \), but the algorithm works by storing the last calculated reputation value in a variable, so it will not be recalculated again at each iteration. However, the calculation of \( \text{reward}^t_w \) is needed. Since the function \( \text{reward}^t_w \) can be computed in time linear in the number of tasks (see Eqs. (1) and (2)), which in turn is linear in the size of the window time, the result follows. □

**Growth Factor** \( \{G'_w(t)\} \) is a parameter which declares services' performance based on their recent strategies and activities. Growth factor is relative to services' reputation \( \text{Rep}^t_w \), QoS during the window time \( t \) \( \text{QoS}^t_w \), and budget \( B^t_w \). This factor is the main variable a typical service uses to decide about which strategy to adopt. The details about the decision making process are described in Section 3. We use Eq. (4) to compute the growth factor \( G'_w(t) \) of the service \( w \) during the window time \( t \) as the average of the three aforementioned parameters, where \( n_t \) is the total number of offered tasks to the whole community during the window time \( t \), \( \mu_w(\text{Rep}^t_w) \) is the mean received service fee, and \( \epsilon \) is the cost of community membership.

\[
G'_w(t) = \frac{\text{Rep}^t_w + \text{QoS}^t_w + \frac{\mu_w(\text{Rep}^t_w)}{n_t} - \epsilon}{3}
\]

\[
\mu_w \in \{\text{CM}, \text{CO}\}, \quad \text{QoS}^t_w = \begin{cases} \sum_{t = 0}^{\text{task}_w(t)} \frac{\text{QoS}^t_w}{\text{task}_w(t)} & \text{if task}_w(t) \neq \emptyset; \vspace{1mm} \\ 0 & \text{otherwise.} \end{cases}
\]

This equation is designed so that it satisfies the following desirable properties:

1. The growth factor function should be monotonically increasing in the offered quality of service \( \text{QoS}^t_w \).
2. The growth factor function should be monotonically increasing in the service's reputation \( \text{Rep}^t_w \).
3. The growth factor function should be monotonically increasing in the budget \( B^t_w \) if the maximum possible profit is positive and monotonically decreasing in \( B^t_w \) if the maximum possible profit is negative. This property reflects the idea that the budget contributes in the increase of the growth factor as far as there is a chance to make profit. In fact, the contribution of the budget \( B^t_w \) in the calculation of the growth factor should be proportional to the maximum possible profit \( n_t \mu_w - \epsilon \) where the service \( w \) receives all the offered tasks during the window time.

It is easy to show that Eq. (4) satisfies the three aforementioned properties by calculating the partial derivatives \( \partial G'_w / \partial \mu_w \) of this function in (1) \( \text{QoS}^t_w / \text{task}_w(t) = \frac{1}{\text{task}_w(t)} \); (2) \( \text{Rep}^t_w / \text{task}_w(t) = \frac{1}{\text{task}_w(t)} \); and (3) \( B^t_w (\text{task}_w(t) - 1) / \text{task}_w(t) \).

Thus, the sign of the two first partial derivatives is positive and the sign of \( \partial G'_w / \partial \mu_w \) depends on the sign of the maximum profit \( n_t \mu_w - \epsilon \), so we are done. The mean service fee depends on the strategy adopted by the service because a competitive service receives higher fees \( \mu_w(\text{CM}) \) compared to a cooperative one \( \mu_w(\text{CO}) (\mu_w(\text{CM}) > \mu_w(\text{CO})) \). The motivation behind this is that a competitive service for a given task is the leader for that task while other cooperative services are performing specific subtasks as asked by the leader.

**Proposition 2.** \( G'_w(t) \) can be computed in time linear in the size of the window \( t \).

**Proof.** As shown in the second part of Eq. (4), the function \( \text{QoS}^t_w \) can be computed in time linear in the number of tasks, which in turn is linear in the size of the window time. Since \( B^t_w \) is constant, the result follows from Proposition 1. □

The above explained parameters and other variables derived from those parameters are listed and self explained in Table 1. Those parameters and variables, which will be used in the rest of the paper, are needed to formalize our decision making model.

2.3. Service interactive strategies

The main goal of each individual service agent is to increase its income (payoff). This income can be earned from tasks (or requests) done by this service. In our model, services can decide to compete to get a task from the master agent or cooperate with other services within a given collaborative network (the way a collaborative network is set by a leader is based on the cooperative services reputation and their QoS parameters that should coincide with the required QoS). Therefore we define two types of service strategies. First, when a service has higher level of confidence based on its growth factor, it can compete to get a task from the master and adopts the competitive strategy. Second, when the service agent has a lower level of confidence that it does not feel capable to compete, it waits for some other services to cooperate with to perform some tasks, and thus it adopts the cooperative strategy. Services estimate the outcome of all the strategies and choose one of them accordingly. This decision is not static but can change over time so service agents can switch from one strategy to the other, and this dynamic attitude is referred to as coopetition. The underlying decision making process is presented in the next section.

3. Theoretical results

3.1. Service decision making procedure

In this section, we explore in details the interaction strategies and the outcome of each strategy in terms of services’ utilities. The main part of services’ decision making procedure falls into their growth factor analysis. In fact, the comparison of the growth factor to a particular threshold is the main reason that influences the service’s decision to follow either competitive or cooperative behavior. Service agents initially compute this value and compare it with their computed threshold. Generally the main challenge is the threshold computation and we cope with this issue in the rest of this section. We additionally use the obtained results in the implemented environment and analyze their effectiveness on services’ strategic decision making procedures.

Fig. 2 shows the decision making process that is followed by a typical service. In case the service agent is ready to compete, there is a chance that it bids for a task if it has the required capabilities to accomplish that task, or stays silent and returns to the cooperative status. But in case the service agent is willing to cooperate, it has to wait for a cooperation opportunity that could be triggered by

\[ \text{Through the paper, requests or tasks are supposed to be decomposable.} \]
another service agent that competed and obtained the task, so both services will be part of the same collaborative network. In the decision making process presented in Fig. 2, we assume that the competing service might get the task (denoted as Bid/obtainedTask) or not in case of being rejected by the master agent, or do not even bid for the task (denoted as Silent/rejectedTask). For simplicity reasons and without loss of generality, we group the two cases of Bids and obtainedTask together as well as Silent and rejectedTask. The rational behind this aggregation is the fact that our main concentration is services’ status (competitive or cooperative) over different decision making rounds, which could be caused by internal factors (the services) or by the external factor (the master agent).

Consider a service w that is willing to compete for the period of time t (that means the computed growth factor is more than the analyzed threshold). This service can estimate the expected payoff associated to this decision, called competition payoff. Eq. (5) computes this expected payoff for the competing service w (\(\pi_{w,CM}^t\)) considering the Bid/obtainedTask probability of \(p_{w,CM}^t\) and Silent/rejectedTask probability of 1 – \(p_{w,CM}^t\).

\[
\pi_{w,CM}^t = p_{w,CM}^t (\mu_{w,CM}^t E_w^t - \text{COF}_w^t E_w^t - \epsilon) + (1 - p_{w,CM}^t)(-\epsilon) \tag{5}
\]

In Eq. (5), \(E_w^t\) is the number of tasks that w expects during the window time t, and \(\mu_{w,CM}^t\) is the mean service fee that is assigned by the master agent to w. This means that a competing service directly obtains this fee from the master agent. Moreover, the competing service w expects a cooperation fee (\(\text{COF}_w^t\)) that it gives to its collaborators in case w needs to cooperate with other services (cooperative service agents in its collaboration network). In any case, the competing or cooperating service agent pays a fixed amount of membership (\(\epsilon\)) to the master agent, which plays the role of the community’s coordinator. This fee would be taken into account when a service decides to leave to a cheaper community or act alone. But to concentrate on the main concerns of this paper, we skip these small details.

From Eq. (5), the following proposition is straightforward.

**Proposition 3.** The complexity of computing the competition payoff \(\pi_{w,CM}^t\) is linear in the competition probability \(p_{w,CM}^t\), the expected number of tasks \(E_w^t\), and the cooperation fee \(\text{COF}_w^t\).

The arrival of requests for service w during the time unit t (denoted here by \(m_w(t)\)) can be modeled as a nonhomogeneous Poisson process (Khosravifar, Bentahar, & Moazin, 2010; Ruggeri & Sivaganesan, 2005), which means as a Poisson process with dynamic arrival rate \(\lambda_w(x)\) where x belongs to the time unit t. The arrival rate is thus a function of time and typically varies significantly from moment to moment. In nonhomogeneous Poisson process, \(m_w(t)\) is expressed as follows:

\[m_w(t) = \int_0^t \lambda_w(x) \, dx\]

And the probability of having exactly n requests during the window t is computed as follows:

\[p(m_w(t) = n) = \frac{(m_w(t))^n}{n! \, \text{e}^{m_w(t)}}\]

Let \(\text{Max}_{w}^t\) be the maximum number of requests that w can receive during t. The number of expected requests \(E_w^t\) is given by the parameter \(\lambda_w(t)\) as follows:

\[E_w^t = \lambda_w(t) = \sum_{n=1}^{\text{Max}_{w}^t} n \, p(m_w(t) = n) \tag{6}\]

The parameter \(\lambda_w(t)\) is usually estimated from data samples using the least squares, iterative least squares, or maximum likelihood (Massey, Parker, & Whitt, 1996).

**Proposition 4.** The complexity of computing the expected number of requests \(E_w^t\) is linear in the size of the window time t.

**Proof.** For a fixed function \(\lambda_w(x)\), \(m_w(t)\) can be computed in \(O(1)\). Thus, from Eq. (6), it follows that \(E_w^t\) can be computed in time linear in \(\text{Max}_{w}^t\). As \(\text{Max}_{w}^t\) is linear in t, the result follows. \(\square\)

Similar to the competitive service case, if a service w declares cooperative status, its expected cooperation payoff (\(\pi_{w,CO}^t\)) is computed in Eq. (7). In this equation, \(\mu_{w,CO}^t\) is the probability of getting involved in a cooperative task with other services and 1 – \(\mu_{w,CO}^t\) is the probability of failure to find such a cooperation opportunity. These two probabilities are set when \(w\) decides to compete. We recall that \(\mu_{w,CO}^t\) denotes the mean cooperation fee that is directly obtained from the leader (i.e., the competitive) service of the underlying collaborative network. Compared to \(\mu_{w,CM}^t\), \(\mu_{w,CO}^t\) is relatively smaller since the competitive service generally dedicates a portion of its obtained income to pay other cooperative services.

\[
\pi_{w,CO}^t = \mu_{w,CO}^t (\mu_{w,CM}^t E_w^t - \epsilon) + (1 - \mu_{w,CO}^t)(-\epsilon) \tag{7}
\]

From Eq. (7), the following proposition holds.

**Proposition 5.** The complexity of computing the cooperation payoff \(\pi_{w,CO}^t\) is linear in the cooperation probability \(\mu_{w,CO}^t\) and the expected number of tasks \(E_w^t\).

To analyze the expected payoffs obtained from different strategies, services need to compute the estimated probabilities that distinguish subcases in each behavioral status (\(p_{w,CM}^t\) for competitive and \(p_{w,CO}^t\) for cooperative). To estimate these probabilities, we should notice that they are functions of services’ reputation values (\(\text{Rep}_w^t\)). Furthermore, \(\mu_{w,CM}^t\) is also function of the difference...
between the offered QoS (QoS\textsubscript{w}) and the mean requested one considering the set of all tasks task\textsuperscript{k} (T\textsubscript{QoS} (see Eq. (8))); and p\textsubscript{w,CO} is function of the reputation of other services in the community because the leader is supposed to be selective when it comes to choose the collaborators. To this end, we first discuss the desirable properties of an estimation function of each of these probabilities, and show that the proposed ones satisfy those properties.

\[
T\textsubscript{QoS}^t = \begin{cases} 
\sum_{\text{task}^t \neq \emptyset} T\textsubscript{QoS} & \text{if } \text{task}^t \neq \emptyset; \\
0 & \text{otherwise}.
\end{cases}
\]  

\textbf{Proposition 6.} \(T\textsubscript{QoS}^t\) can be computed in time linear in the size of the window \(t\).

\textbf{Proof.} From Eq. (8), \(T\textsubscript{QoS}^t\) can be computed in time linear in |\{task\textsuperscript{k}\}|, which in turn is linear in the window size \(|t|\). □

The desired properties of \(p\textsubscript{w,CM}^t\) are as follows:

\textbf{Property 1.} \(p\textsubscript{w,CM}^t\) is continuous with regard to \(\text{Rep}^t\textsubscript{w}\), \(\text{QoS}^t\textsubscript{w}\), and \(T\textsubscript{QoS}^t\).

\textbf{Property 2.} \(p\textsubscript{w,CM}^t\) is monotonically increasing in \(\text{Rep}^t\textsubscript{w}\) and \(\text{QoS}^t\textsubscript{w}\) while \(\text{QoS}^t\textsubscript{w} - T\textsubscript{QoS}^t\) is positive.

\textbf{Property 3.} \(p\textsubscript{w,CM}^t\) is null if \(\text{QoS}^t\textsubscript{w} - T\textsubscript{QoS}^t\) is negative.

\textbf{Property 4.} The increase slope of \(p\textsubscript{w,CM}^t\) is higher when the reputation \(\text{Rep}^t\textsubscript{w}\) increases in the interval [0,0.5] than when it increases in the interval [0.5,1].

Property 1 simply says that the probability of success competition \(p\textsubscript{w,CM}^t\) can be always estimated as far as \(\text{Rep}^t\textsubscript{w}\), \(\text{QoS}^t\textsubscript{w}\), and \(T\textsubscript{QoS}^t\) are available. Property 2 says that the reputation and QoS are two key factors that influence the value of \(p\textsubscript{w,CM}^t\) in the sense of positive correlation. Property 3 indicates that the probability \(p\textsubscript{w,CM}^t\) is null if the offered QoS is less than the expectation. Property 4 promotes the increase of the reputation for new comers and imposes higher increase rate at the beginning of the reputation curve because it is always hard to build the reputation, but once it is built, its maintenance is less challenging.

The desired properties of \(p\textsubscript{w,CO}^t\) are as follows:

\textbf{Property 5.} \(p\textsubscript{w,CO}^t\) is continuous with regard to \(\text{Rep}^t\textsubscript{w}\) and the reputation of other services in the community.

\textbf{Property 6.} \(p\textsubscript{w,CO}^t\) is monotonically increasing in \(\text{Rep}^t\textsubscript{w}\) and monotonically decreasing in the community average reputation.

\textbf{Property 7.} The increase slope of \(p\textsubscript{w,CO}^t\) is higher when the reputation \(\text{Rep}^t\textsubscript{w}\) increases in the interval [0,0.5] than when it increases in the interval [0.5,1].

Property 5 is similar to Property 1. Property 6 says that \(w\) has more chance to get involved in a cooperation if it has high reputation compared to the other members. This chance decreases if other services have higher reputation. Property 7 is similar to Property 4.

Eqs. (9) and (10) respectively compute the estimated success probability in cases where service \(w\) is competing and cooperating. These values are computed considering service’s reputation value (\(\text{Rep}^t\textsubscript{w}\) computed by the master), service’s offered QoS (QoS\textsubscript{w}), the task required QoS (T\textsubscript{QoS}), which is the mean required QoS computed from previous tasks, the maximum offered QoS (QoS\textsubscript{w}), which is provided by another competitive service \(k\), and the cooperative factor \(CL\textsubscript{w}\) of service \(w\) during the window time \(t\), which is computed as the portion of service’s current reputation on the average reputation of the community \(C\).

\[
p\textsubscript{w,CM}^t = \begin{cases} 
\sin(\text{Rep}^t\textsubscript{w} \cdot \frac{|C|}{2}) \cdot \frac{\text{QoS}^t\textsubscript{w} - T\textsubscript{QoS}^t}{\text{Max}_{k}(\text{QoS}^t\textsubscript{w} - T\textsubscript{QoS}^t)} & \text{if } \text{QoS}^t\textsubscript{w} \geq T\textsubscript{QoS}^t; \\
0 & \text{otherwise}.
\end{cases}
\]

\[
p\textsubscript{w,CO}^t = \sin(\text{Rep}^t\textsubscript{w} \cdot \frac{|C|}{2}) \cdot CL\textsubscript{w}^t.
\]

\textbf{Theorem 1.} Eq. (9) satisfies Properties 1–4.

\textbf{Proof.} It is easy to show the continuity of Eq. (9), which satisfies Property 1. The partial derivative \(\frac{\partial p\textsubscript{w,CM}^t}{\partial \text{Rep}^t\textsubscript{w}} = \frac{\sin(\text{Rep}^t\textsubscript{w} \cdot \frac{|C|}{2}) \cdot \text{QoS}^t\textsubscript{w} - T\textsubscript{QoS}^t}{\text{Max}_{k}(\text{QoS}^t\textsubscript{w} - T\textsubscript{QoS}^t)}\) is positive as the function \(\cos(\text{derivative of sin})\) is positive on \([0,\frac{|C|}{2}], \text{Rep}^t\textsubscript{w} \in [0,1]\), and \(\text{QoS}^t\textsubscript{w} \geq T\textsubscript{QoS}^t\). The partial derivative \(\frac{\partial p\textsubscript{w,CM}^t}{\partial T\textsubscript{QoS}^t}\) with regard to \(\text{QoS}^t\textsubscript{w}\) is \(\frac{\sin(\text{Rep}^t\textsubscript{w} \cdot \frac{|C|}{2})}{\text{Max}_{k}(\text{QoS}^t\textsubscript{w} - T\textsubscript{QoS}^t)}\) is also positive since \(\text{QoS}^t\textsubscript{w} - T\textsubscript{QoS}^t > 0\) and \(\text{Rep}^t\textsubscript{w} \in [0,\frac{|C|}{2}]\) and sin is positive on \([0,\frac{|C|}{2}]\), which proves the satisfaction of Property 2. Property 3 is straightforward. Finally, the increase slope of the function sin on \([0,\frac{|C|}{2}]\) proves Property 4. □

\textbf{Theorem 2.} Eq. (10) satisfies Properties 5 to 7.

\textbf{Proof.} We can easily show the continuity of Eq. (10) from which Property 5 follows. Property 6 can be shown by calculating the partial derivative \(\frac{\partial p\textsubscript{w,CO}^t}{\partial \text{Rep}^t\textsubscript{w}}\) first with regard to \(\text{Rep}^t\textsubscript{w}\) and second with regard to the community \(C\) average reputation \(\sum_{k \in C} \text{Rep}^t\textsubscript{w}/|C|\), where \(|C|\) is the cardinality of the considered community \(C\). The first partial derivative \(\frac{\partial p\textsubscript{w,CO}^t}{\partial \text{Rep}^t\textsubscript{w}}\) is positive and the second \(\frac{\sum_{k \in C} \text{Rep}^t\textsubscript{w}/|C|}{\sum_{k \in C} \text{Rep}^t\textsubscript{w}/|C|}\) is negative, which proves the satisfaction of Property 6. The proof of satisfaction of Property 7 is similar to the one of Property 4. □

\textbf{Property 7.} \(p\textsubscript{w,CM}^t\) can be computed in time linear in the window size \(|t|\).

\textbf{Proof.} The result follows directly from (1) Eq. (9); (2) Property 1 (Complexity of \(\text{Rep}^t\textsubscript{w}\) is linear in the window size \(|t|\)); (3) second part of Eq. (4) (the function \(\text{QoS}^t\textsubscript{w}\) can be computed in time linear in the number of tasks, which in turn is linear in the size of the window time); (4) Property 6 (Complexity of \(T\textsubscript{QoS}^t\) is linear in the size of the window \(t\)); and (5) the fact that those functions are computed independently one from the other. □

\textbf{Property 8.} \(p\textsubscript{w,CO}^t\) can be computed in time \(O(|t|,|C|)\), which means linear in both the size of the window \(t\) and the size of the community \(C\).

\textbf{Proof.} From Proposition 1, \(\text{Rep}^t\textsubscript{w}\) can be computed in \(O(|t|)\). Consequently, the function \(\sum_{k \in C} \text{Rep}^t\textsubscript{w}\) can be computed in \(O(|t|,|C|)\), so it does the computation of the function \(CL\textsubscript{w}^t\) as \(p\textsubscript{w,CO}^t\) will be computed just once and stored in a variable. The same variable will be used to compute \(\sin(\text{Rep}^t\textsubscript{w} \cdot \frac{|C|}{2})\). Thus, from Eq. (10), the result follows. □
The coopetition threshold

In this part, we compute the coopetition threshold that a typical service agent could use to adopt reasonable interacting strategies and we empirically verify the effectiveness of the obtained results in the next section. In fact, to decide which strategy to adopt, we let the service agent w compare its growth factor \( G_w \) with the coopetition threshold \( t^*_w \) that we compute in Eq. (11). Based on this probability, we calculate the total utility \( U_w \) in Eq. (12).

\[
P_w = \begin{cases} \frac{C_0}{C_1} & \text{if } G_w \leq t^*_w; \\ 1 & \text{otherwise.} \end{cases} 
\]

\[
U_w = P_w^l (\pi^l_{w,CM}) + (1 - P_w)(\pi^l_{w,CO}) 
\]

The key factor in the computation of the probability \( P_w \) and the associated utility is the threshold value. To compute the threshold, we use the game-theoretic best response technique. A typical service agent w will follow the best response strategy to maximize its expected aggregated payoff. The idea is to equalize the expected payoffs of the two acting strategies: compete and cooperate. The objective behind equalizing payoffs is to explore conditions under which service agent w could react with best response to further decision making procedures. We use the obtained conditions to compute the threshold \( t^*_w \) during the window time \( t \). By equalizing \( \pi^l_{w,CM} \) and \( \pi^l_{w,CO} \), we obtain:

\[
\pi^l_{w,CM} = \pi^l_{w,CO} \rightarrow 
\]

\[
P_w^l (\pi^l_{w,CM}) + (1 - P_w^l)(\pi^l_{w,CO}) = P_w^l (\pi^l_{w,CO} + (-\epsilon)) 
\]

\[
\Rightarrow P_w^l (\pi^l_{w,CM} - COF - \epsilon) + (1 - P_w^l)(\pi^l_{w,CO}) 
\]

Which means:

\[
p_w^l (\pi^l_{w,CM} - COF - \epsilon) = p_w^l (\pi^l_{w,CO} - \epsilon) + (\epsilon)(p_w^l - p_w^l) 
\]

So, we obtain:

\[
\mu^l_{w,CM} - COF - \epsilon = \frac{p_w^l}{p_w^l} \frac{\mu^l_{w,CO}}{p_w^l^l} - \epsilon 
\]

Therefore:

\[
COF = \frac{\mu^l_{w,CM} - \mu^l_{w,CO}}{p_w^l - p_w^l} 
\]

From which, we derive:

\[
COF = \frac{\mu^l_{w,CM}}{p_w^l - p_w^l} 
\]

Replacing \( p_w^l \) and \( p_w^l \) using Eqs. (9) and (10), we derive the following:

\[
COF = \frac{\sin(Rep^l_w)}{\sin(Rep^l_w)} \frac{QoS}{\max(QoS - T_{QoS})} 
\]

By simplifying the \( \sin \) function from both the numerator and denominator sides and substituting the cooperation factor \( CL^l_w \) of service w we obtain Eq. (13):

\[
COF = \mu^l_{w,CM} - \frac{\mu^l_{w,CO} \max(QoS - T_{QoS})}{\sum_k \mu^l_{w,CO} \max(k) - T_{QoS}} 
\]

Eq. (13) computes the cooperation fee \( COF \) that is assigned by service w. This fee represents the amount that \( w \) spends to cooperate with other service \( s \) to accomplish the task. By so doing, we obtain the maximum amount of cooperation fee that service w can use to constrain the positive payoff out of competing. Otherwise, the service stays as cooperative entity.

**Proposition 9.** \( COF \) can be computed in time \( O(|t|, |C|) \), which means linear in both the size of the window \( t \) and the size of the community \( C \).

**Proof.** From Proposition 1, \( Rep^l_w \) can be computed in \( O(|t|) \). Consequently, the function \( \sum_k \mu^l_{w,CO} \) can be computed in \( O(|t|, |C|) \). Since \( QoS \) and \( T_{QoS} \) can be computed in time \( O(t) \) (from the second part of Eq. (4) and Proposition 6 respectively), we are done. □

**Lemma 1.** The competition payoff \( \pi^l_{w,CM} \) can be computed in time \( O(|t|, |C|) \).

**Proof.** The result follows directly from Propositions 3, 4, 7, and 9. □

**Lemma 2.** The cooperation payoff \( \pi^l_{w,CO} \) can be computed in time \( O(|t|, |C|) \).

**Proof.** The result follows directly from Propositions 4, 5, and 8. □

We use the maximum cooperation fee that a service agent considers to constrain the positive expected payoff when the competitive strategy is adopted to update the threshold for the consequent time window \( (t+1) \). We compare the maximum cooperation fee with the required fee \( Req^l_w \) that the service indicates to accomplish the task. The outcome of this comparison is a factor that uses the current threshold \( t^*_w \) to compute the consequent threshold \( t^*_{w+1} \). As in online learning, the idea is to compute...
iteratively the threshold until the fixed point is achieved, which indicates the threshold’s conversion, where the initial value is randomly chosen (in the simulation different initial values are used). Eq. (14) shows this computation. To investigate the effectiveness of this threshold on the outcomes of the services that follow this reasoning technique, in the next section, we compare the results of different agents with diverse strategic reasoning techniques.

\[
\tau_{w}^{t+1} = \begin{cases} 
\Theta & \text{if } 0 < \Theta \leq 1 \\
1 & \text{if } \Theta > 1; \\
0 & \text{if } \Theta < 0.
\end{cases}
\]

\[
\Theta = \frac{\text{COP}_w}{\text{ReqF}_w}
\]  

**Proposition 10.** The threshold \(\tau_{w}^{t}\) can be computed in time \(O(|t|,|C|)\), which means linear in both the size of the window \(t\) and the size of the community \(C\).

**Proof.** From Eq. (14), the computation of \(\tau_{w}^{t}\) is recursive on \(t\), and the algorithm works by keeping the last computed value in a variable, which saves the time of re-calculation. Thus, the complexity of calculating \(\tau_{w}^{t}\) is determined by the complexity of calculating \(\text{COP}_w\) since \(\text{ReqF}_w\) is constant during the period \(t\). Consequently, the result follows from Proposition 9. \(\square\)

**Theorem 3.** The time complexity of the proposed decision mechanism is \(O(|t|,|C|)\), which means linear in both the size of the window \(t\) and the size of the community \(C\).

**Proof.** The procedure mechanism is based on comparing the growth factor \(G_w\), with the coopetition threshold \(\tau_{w}^{t}\) as shown in Eqs. (11) and (12). Thus, the result follows from Propositions 2 and 10. \(\square\)

### 4. Experimental results

In this section, we provide an empirical analysis over the theoretical results regarding the characteristics of intelligent service agents hosted in different communities of services. In the implemented system, we simulate the behaviors of service consumers as request generators, service agents as service providers, and master agents as community representatives. These entities are developed with respect to what is explained in Section 1. In this section, the objective is to investigate the effectiveness of the proposed strategic system on intelligent services’ overall budget and also the average quality and quantity of tasks performed by the community of services, which directly affects user satisfaction. To verify these objectives, we study the overall performance of the community hosting the reasoning-empowered services compared to the ones hosting stochastic and purely competitive services. By stochastic services, we mean services that adopt at each moment competitive or cooperative strategies equally, but in a random way. By equally, we mean the choices are fairly divided between the two strategies.

Our simulation application is written in C# using Visual Studio. The implementation is performed on a single Intel Xeon X3450 machine with 6GBs of memory. We modeled web services as classes, and by using Await and Async models, we initiated different web services, each running as a thread. We implemented the SOAP standard as XML based messaging system with request parameters and a list of XML based responses. We used the domain of flight booking to implement our services. Each user’s request contains the flight dates, the origin and destination, type of tickets (one way or return), and number of guests. The response contains different flights with different companies, prices, timing, etc. We have gathered around 200,000 flights and stored them in our MongoDB database. A pool of services are initialized with values taken from a real dataset that includes 2507 real services functioning on the web. The dataset records the QoS values of 9 parameters including availability, throughput, and reliability (Al-Masri & Mahmoud, 2007).

We start our discussions with cumulative budget comparison regarding different communities within which services follow different reasoning techniques. Fig. 3 Part (a) illustrates three graphs for three different communities. Each community hosts services that follow different reasoning techniques: (1) a community that follows the interactive reasoning techniques presented in this paper (referred to as coopetitive); (2) a community that follows a stochastic reasoning technique so decisions about selecting competitive or cooperative strategies are totally random (referred to as random competitive); and (3) a competitive community where all services follow the competitive strategy (referred to as competitive).

The proposed model’s reasoning mechanism enables services to effectively select their interacting strategies and the obtained budget represents the best outcome over the strategic decision making procedure they run all the time. This procedure avoids cases where a service selects the competitive strategy but gets refused to obtain a task from the master. The procedure allows services to make decisions that maximize their utilities, so that if the service cannot compete, the procedure would suggest to collaborate, which is better than competing and failing to obtain the task. In this case (i.e., competing and not getting the task), the service stays idle but still pays the community membership fee, which means losing utility. The developed strategic decision making mechanism leads some service agents to follow cooperative strategies that overall maintain an optimal community budget.

In the same figure (Fig. 3), we observe the cumulative budget of a community where services follow random interacting strategies. The outcome is clearly lower because services at each run randomly decide over their acting strategies. This potentially influences the community budget because a low quality service if randomly selects to follow the competitive strategy, it will fail to perform a task with high QoS requirements. This kind of strategy selection is totally stochastic while the task allocation algorithm follows a logical path. The ideal system is the one that analyzes the optimal strategic path and consistently follows strategies that bring maximum outcome. The result regarding the community that follows the random strategy shows how stochastic decision making degrades the community budget, but still this result outperforms the one of the purely competitive community. In this purely competitive case, all the services follow the competitive strategy and the frequency of getting rejected from a task is relatively high. In a community with limited number of tasks, the competitive strategy for all services highly influences the community budget because a potential group of services are losing at every run.

The results illustrated in Fig. 3 Part (a) verify the importance of the strategic decision making procedure to logically decide over the possible competitive and cooperative choices. Fig. 3 Part (b) illustrates communities average reputation of involved services. The graphs represent the influence of the rewards that the master agent uses to encourage highly capable services to compete for a task. As for the cumulative budget, we observe that the coopetitive community outperforms the random coopetitive and competitive communities in terms of average reputation. The proposed model’s average reputation increases because services follow optimal strategies where they can perform better so obtain higher rewards. For the same reasons as for the cumulative budget, the average
reputation of the random coopetitive community outperforms the one of the competitive community.

In Fig. 4, we analyze the scenario where the membership fee is high, which is analogous to situations where the task income is low. We need to verify how well services can perform in these situations. Fig. 4 Part (a) depicts the results in a normal scenario. In part (b), we have increased the membership fee to the high amount of 5 units per run. Services will need to pay 5 units on each iteration of tasks being offered. As results, the strategic coopetitive decision making process helps services make benefits, but stochastic and compete methods lead to lose income since making mistakes and trying to compete for jobs the services cannot afford while having to pay higher membership fees is making them lose budget in a faster pace. In part (c), the membership fee is higher than an average income that a task can generate. The simulation shows that the stochastic and compete strategies make services budget decreases drastically. However, our strategic coopetitive decision making process loses budget in a slower pace.

In Fig. 5 parts (a) and (b), we observe the competitive and coopetitive probability of four different services where two of them ($w_1$ and $w_3$) are following optimal strategies as suggested by our model (competitive for $w_1$ and cooperative for $w_3$) and the two others ($w_2$ and $w_4$) are following non-optimal strategies, which are the opposite of what our model suggests. Over elapsing runs, services that follow optimal strategies bring best reputation. In fact, the master agent rewards the high quality service that chooses the competitive strategy, cooperates with other peers and successfully accomplishes the task. In this system, the reputation regarding such a service is increasing over time and the possibility of allocating further tasks is increasing as well. By increasing the growth factor, such a service (here shown as $w_1$) increases the probability of selecting the competitive strategy.

On the other hand, the other service (here shown as $w_2$) that is incapable of competing is penalized by the master agent because the offered quality might not meet the required task quality. Thus, $w_2$ degrades its growth factor by following the competitive strategy. As intelligent entity, this service is encouraged to change its strategy to the cooperative one and thus, its probability of selecting the competitive strategy is decreasing over time. We have similar results in Fig. 5 Part (b) regarding services $w_3$ and $w_4$ where unlike $w_4$, $w_3$ is strategically following the cooperative strategy allowing for an increase in the overall reputation.

In our model, services are managed by selfish agents in the sense they try to maximize their own utilities. We analyze how their strategies affect the social welfare, and from user’s and community’s point of view how good the tasks are being performed. This directly impacts user’s satisfaction and community’s reputation in general. The Higher quality and quantity of tasks performed leads to higher user’s satisfaction for the community which results in better reputation for the community.

The results in Fig. 6 show the quality and quantity of tasks being done successfully in three communities adopting the three

Fig. 4. High membership fees or less task income impacts on different selection strategies. Part (a) Medium membership fee. Part (b) High membership fee. Part (c) Very high membership fee.
different aforementioned strategy decision algorithms. Part (a) depicts the number of tasks successfully done. When services are not capable of performing tasks alone and decide to compete for tasks with high QoS requirements, they usually fail to perform the task. However, if they opt for cooperating although they have less income, they have higher chance of performing tasks collaboratively with other services.

Fig. 6, Part (b) shows the probability of successfully performing tasks, which influences users’ satisfaction. The results show that services using our coopetitive strategy when deciding whether to compete or cooperate have higher chance to get high satisfaction ratio compared to stochastic and competitive strategies.

The quality of tasks performed are depicted in Fig. 6 Part (c). The figure shows the average QoS of tasks successfully performed. As clearly confirmed by the simulation, the coopetitive community outperforms the stochastic and competitive communities.

We conclude our analysis by discussing how effective our coopetitive decision making model is by comparing the final utility (in terms of income) of services following our model with other services deviating from that coopetitive model. In Fig. 7 Part (a), we made the services deviate from the suggested strategy. As the simulation shows, the more services deviate from our coopetitive strategy the more they make less benefits.

In Huang and Hu (2004), the authors propose a framework to match potential benefits of services while cooperating with one another. The interesting idea is the consideration of the benefits under four categories: innovation and learning, internal business process, customer, and financial benefits. The innovation and learning perspective focuses on the knowledge, skills, and systems needed by the services to improve the business continually. Necessary factors to build strategic capabilities and efficiency are addressed in the internal business process. Values and criteria that customers seek when requesting services are considered in the customer perspective. Finally, the final performance to maximize the shareholder value is analyzed in the financial perspective. The authors goal is to design the framework for cooperating web services, inline with business strategy of firms in IT industry. However, competition among those services has not been analyzed.

In Alchierri, Bessani, and Fraga (2008), the authors present a dependable framework for cooperative service agents that is based on the tuple space coordination model. The intrusion-tolerant perspective is emphasized in this paper where several security mechanisms are developed to enable a reliable coordination system. The proposed frameworks mostly aim to facilitate the coordination mechanism between services. However, the opposite strategy of cooperating is not analyzed where services might be more successful when competing within the same group. In fact, services are not always willing to cooperate even if they have some common goals, particularly when they operate within groups such as communities. In such a context, service agents can follow different interacting strategies and have to decide when to compete and when to cooperate so that their ultimate goal, maximizing their incomes, can be better achieved. In our framework, we analyze those different strategies to help services in their decision making process when these agents function within communities. We enable service agents to reasonably evaluate and decide over their cooperation strategies.

Furthermore, there are a number of related proposals that take into account the correlation between (web) services and the ways these services coordinate their actions to accomplish the required tasks. In Jurca and Faltings (2007, 2005), Malik and Bouguettaya (2007), Maximilien and Singh (2002) and Yahyaoui (2012), the authors propose to rank services based on their reputation in the system and to use this ranking as a means to facilitate cooperation of services. In those models, services rely on one another on the basis of the reputation ranking system, using, among other parameters, the QoS (Tao, Chang, Gu, & Yi, 2012).

There are other models that facilitate cooperation mechanisms among services using various techniques. Examples of those techniques include (1) coordination between two types of behaviors.
associated with component services: operational and control behaviors (Yahyaoui, Maamar, & Boukadi, 2010); (2) Services-based workflows (Wu & Lin, 2011); (3) transaction-based approaches (Gao, Urban, & Ramachandran, 2011; Rosario et al., 2007); (4) agent coordination mechanisms (Charif & Sabouret, 2013; Gutierrez-García & Sim, 2013); (5) logical techniques (Okutan & Cicikli, 2010; Tang, Jiang, & Zhou, 2011); and (6) community models, which are virtual structures that aim at increasing the visibility of services and facilitating their discovery and composition by hosting and gathering services having similar or complementary functionalities but different QoS parameters (Khosravifar et al., 2010a). However, deciding about which strategy to choose when services are competing but still need to cooperate to accomplish complex tasks has not been addressed and kept as open issue in all these proposals as

![Fig. 6. Overall performance from community's point of view. Part (a) Total number of tasks successfully done. Part (b) Ratio of tasks satisfied with required QoS. Part (c) Average QoS of performed tasks.](image)

![Fig. 7. Utility loss while deviating from our coopetitive decision process. Part (a) Overall budget when deviating from our model in 0, 10, 20, 30, 40, and 50 percent of decisions. Part (b) Ratio of getting earning utility (budget) when deviating from our coopetitive strategy in 1 to 10 decisions.](image)
The contribution of this paper is the proposition of a coopetitive strategic model to analyze the interacting behavior of intelligent services that are active within communities. We considered two acting strategies where service agents expect different sort of pay-offs: (1) competitive strategy where the service claims that it can accomplish a task, and therefore can take the responsibility over the service consumer satisfaction; and (2) cooperative strategy where the service does not take the responsibility to accomplish the task and only cooperates with competitive peers. Our proposed model advances the state-of-the-art in cooperative systems by enabling intelligent agent-based services to effectively choose their interacting strategies that lead to optimal outcomes. The proposed framework provides a reasoning technique that service agents can use to increase their overall obtained utilities. The theoretical results presented in this paper are also backed by simulation results using a real services dataset. Those results showed that our model outperforms existing competitive and random cooperative strategies and the more services deviate from the coopetitive strategy suggested by our decision-making mechanism the more they make less benefits. Thus, deviating from our coopetitive strategy yields less income for the service.

As future work, we plan to emphasize the service consumer role in the proposed model to obtain more accurate results when consumers post their service satisfaction feedback. Adding users feedback as a system parameter in our algorithm can improve the decision making process, by considering, not only the system and services perspective, but also the users perspective. This will potentially lead to more efficiency, payoff and income for web services. Moreover, we would like to enhance the reasoning technique’s features to cope with different unexpected scenarios. We also need to expand the work to enable services to choose their collaboration networks.

6. Conclusion

The theoretical and simulation results presented in this paper are also backed by simulation results using a real services dataset. Those results showed that our model outperforms existing competitive and random cooperative strategies and the more services deviate from the coopetitive strategy suggested by our decision-making mechanism the more they make less benefits. Thus, deviating from our coopetitive strategy yields less income for the service.

As future work, we plan to emphasize the service consumer role in the proposed model to obtain more accurate results when consumers post their service satisfaction feedback. Adding users feedback as a system parameter in our algorithm can improve the decision making process, by considering, not only the system and services perspective, but also the users perspective. This will potentially lead to more efficiency, payoff and income for web services. Moreover, we would like to enhance the reasoning technique’s features to cope with different unexpected scenarios. We also need to expand the work to enable services to choose their collaboration networks.

References


