When to Compete and When to Cooperate: Analysis on Strategies of Services within Communities

Babak Khosravifar[1], Ehsan Khosrowshahi Asl[2], Jamal Bentahar[2], Rabeb Mizouni[3], and Hadi Otrok[3]

[1]McGill University, Canada, [2] Concordia University, Canada, [3]Khalifa University, UAE
$babak.khosravifar@mcgill.ca$, $e\_khosr@encs.concordia.ca$,
$bentahar@ciise.concordia.ca$, $rabeb.mizouni@kustar.ac.ae$,$hadi.otrak@kustar.ac.ae$

Abstract. Recently, a number of frameworks have been proposed to aggregate web services within communities for the purpose of enhancing their capabilities with respect to providing the required services. Most of the proposed frameworks suggest that web services within these communities are competing but also exhibit cooperative behavior, so web services are said to be coopetitive. However, deciding which strategy to adopt, which means competing or cooperating is still an open question. The purpose of this paper is to answer this question by discussing a mechanism web services can use to effectively choose interacting strategies which bring maximum utility. In this direction, we investigate web services' characteristics and their expected utilities over different strategies. We enable web services that are hosted in communities with reasoning capabilities to enhance their quality of strategic interacting mechanisms as decision making procedures. The ultimate objective is to measure the threshold that web services can use in order to decide about different interacting strategies. Moreover, we develop a simulated environment where we analyze different scenarios and verify the obtained theoretical results using parameters from a real web services dataset.

Keywords: Web services; Reputation; Strategies.

1 Introduction

Web services are developed to continuously interact with others that could be different types of web services or service consumers. Abstracting and associating web services with knowledge-empowered agents without changing the web services implementation benefit them from interactions that those agents are able to manage [8, 12, 20]. That means web 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 a key factor [10, 13]. Those web services follow two different strategies of cooperating or competing towards serving service consumers and increasing their utilities in terms of payoffs. Analyzing web services acting strategies 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 and enable web services to cope with strategic decision making in a particular situation.

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

a task. In [9–11, 15, 16], the authors propose to rank web 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 using the reputation ranking system. There are other models that facilitate cooperation mechanisms using transaction-based web services [18] and communities that host and gather web services having similar or complementary functionalities but different QoS parameters [12]. However, deciding about which strategy to choose when web services are competing but still need to cooperate to accomplish complex tasks is still an open issue. We propose analyzing those different strategies to help web services in their decision making process when these web services function within communities. The objective is to enable web services to reasonably evaluate and decide over their coopetition strategies, which means when to compete and when to cooperate.

More precisely, we propose a mechanism within which web services in the community could choose either to compete for an announced task, or to cooperate with other competing web services in the same community to accomplish some subtasks of the announced task. We explore details behind the strategic decision making procedures and enable web services to apply different techniques to constrain high efficiency and obtain the maximum utility. We investigate web services' expected payoffs and the involved probabilities that are used to choose over the two interacting strategies. As intelligent entities, web services require a reasoning technique that enhances their abilities over best acting strategies and the attitude they could exhibit to yield maximum utility. In this paper, we obtain some theoretical results about the decision making algorithm that could be used by a typical web service. We also implement a simulation environment with a number of communities hosting a number of web services having parameters extracted from a real dataset [2][1]. This dataset represents 2507 real web 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 web services over a 6 day period. We equip some of those web services with our proposed strategic decision making procedure and compare the performance of the equipped services against other ordinary web services. We provide detailed discussions over the implemented environment and verify the effectiveness of the proposed mechanism.

The rest of the paper is organized as follows. Section 2 is dedicated to the presentation of some preliminaries about the system design that are needed to understand the proposed model. Section 3 is devoted to the theoretical presentation of our model along with web services' competitive and cooperative strategies. Section 4 outlines the experimental results and a detailed comparison of the equipped web services with strategic reasoning techniques with the ordinary services. Finally, related work and some conclusions are discussed in Section 5.

2 The Proposed Framework

In this Section, we first present the architecture of the proposed model. We explore the characteristics of intelligent web services and their network. We link this architec-

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

ture to the implemented system where we investigate the services' coopetitive attitudes. We compute the involved system parameters and explain the web services' interactive strategy profiles by highlighting their coopetitive choices.

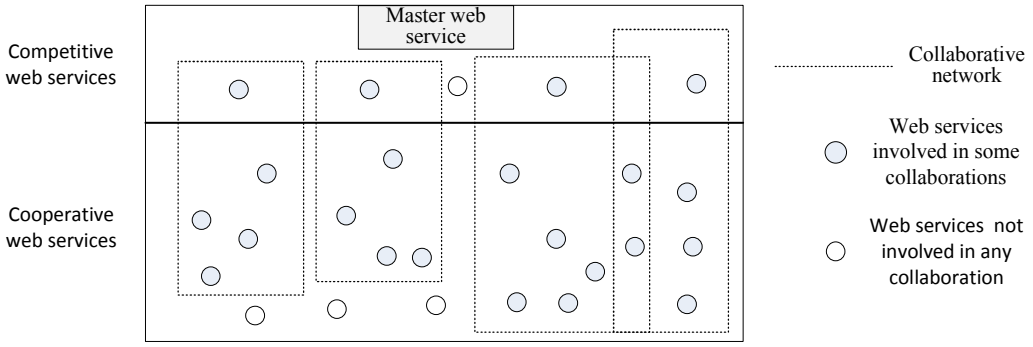

Fig. 1. Web services are partitioned into competitive and cooperative sets. Competitive web services may get tasks directly from the master web service and they can share it with other cooperative web services in their collaborative networks.

2.1 The Architecture

The proposed system consists of three types of autonomous entities with different goals.

1) *Web Services* that reside inside a community which aggregates a number of functionally similar or complementary web services as a group (more details about communities of web services are given in [14]). Within the same community, each web service might have a network consisting of some other web services that might get involved in a cooperative work (e.g. composition and substitution). As web 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 Web Service* is the manager and representative of the community of web services. Among other functionalities, the master web service is responsible for allocating the tasks to web services within the community. After the task being accomplished, the master rewards or penalizes the associated web service with respect to its reputation as a member of the community. 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 web services' interactive strategies.

Figure 1 illustrates the architecture of a typical community aggregating a number of web 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 competed web service as the task leader and do not

directly interact with the master (the master deals only with the web service that has bid for the task, which is responsible of choosing its collaborative network). In both sets, some web services 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 web services based on some parameters (such as reputation) that we explain in the rest of this section. If the web service is busy or unwilling to take the task, the master allocates the task to the second competing web service in the list. There is a chance that some tasks could not be assigned to any web service. These tasks are accumulated to the task pool to be allocated in the next task allocation round. Upon allocation of the task, the web service is responsible to offer the required QoS that is stated in the task being generated by a consumer. Afterwards, the master rewards or penalizes the competing web 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.

Task QoS (T^t_{QoS}) is the required QoS metric for a specific task at time t. Users define tasks with specific QoS requirements such as response time, availability, and successability (or accuracy) [4]. We aggregate and normalize these metrics to a value between 0 and 1.

Web service QoS (QoS^t_w) is the QoS provided by the web service w after performing a task at time t. 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^t_{QoS}. 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 web services' reputation.

Budget (B^t_w) is the amount of money the web service w has in its disposal at time t, which helps pay for the community membership fees (ϵ) and is one of the parameters that the web service considers when deciding about getting involved in a competition or not. This parameter has been used in other service computing settings such as [4].

Reputation is a factor in any online community where trust is important. Without any trust enabling mechanism, users cannot differentiate between services, specially the ones which offer the same type of service. Reputation mechanisms usually aggregate users' experiences and in our case it strongly depends on QoS that each web service provides. Users define tasks with specific quality T^t_{QoS}, therefore after performing a task with QoS^t_w, the reputation of w gets evaluated by the master web service. Rep^t_w refers to the reputation of w at time t.

In Equation 1, we compute the reward that the master computes considering the task QoS compared with the web service offered quality QoS^t_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 η which the master considers together with the proportional level of satisfaction as a weighted value (by υ). In this case, the higher the

offered quality is, the more weighted reward is. In case the offered quality did not meet the user expectations, the reward would be negative. In this case, we also have a default penalty value ρ (where $\rho > \eta$) together with the weighted proportional difference. The idea is to harshly penalize the web services rather than rewarding them. To this end, rational web services should carefully analyze their capabilities once the available tasks are announced. In our proposal, web services 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.

$$reward_w^t = \begin{cases} \eta + \frac{QoS_w^t}{T_{QoS}^t + QoS_w^t} * \upsilon & \text{if } T_{QoS}^t \leq QoS_w^t; \\ -(\rho + \frac{T_{QoS}^t}{T_{QoS}^t + QoS_w^t} * \upsilon) & \text{otherwise.} \end{cases} \tag{1}$$

The assigned reputation value is updated by the computed reward value. The computed reputation of web services is bounded by the minimum and maximum reputation values 0 and 1. Let $\Gamma = Rep_w^t + reward_w^t$. The updated reputation value is then computed as follows:

$$Rep_w^{t+1} = \begin{cases} \Gamma & \text{if } 0 \leq \Gamma \leq 1; \\ 0 & \text{if } \Gamma < 0; \\ 1 & \text{if } \Gamma > 1. \end{cases} \tag{2}$$

Growth Factor is a parameter which declares web services' performance based on their recent strategies and activities. Growth factor is relative to web services' reputation and QoS. This parameter is the main variable a typical web service uses to decide which strategy to adopt. The details about the decision making process is described in Section 3. We use Equation 3 to compute the growth factor G_w^t of the web service w at time t. The growth factor function should be monotonically increasing in QoS_w^t, Rep_w^t and B_w^t, which is satisfied by the equation and this could be easily proven by calculating the partial derivatives of this function in 1) QoS_w^t; 2) Rep_w^t; and then 3) B_w^t and show that they are all positive. The contribution of the budget B_w^t in the calculation of the growth factor should be proportional to the ideal budget $\beta_w \times t$ where the web service receives all the offered tasks during the periods t. The parameter β_w denotes the profit obtained considering the mean received service fee μ_w and the cost of community membership ϵ. The mean service fee depends on the strategy adopted by the web service because a competitive service receives higher fees $\mu_{w,CM}$ compared to a cooperative one $\mu_{w,CO}$ ($\mu_{w,CM} > \mu_{w,CO}$). The motivation behind this is that a competitive web service for a given task is the leader for that task while other cooperative web services are performing specific subtasks as asked by the leader.

$$G_w^t = \frac{Rep_w^t + QoS_w^t + \frac{B_w^t}{t \times \beta_w}}{3} \tag{3}$$

$$\beta_w = \mu_w - \epsilon, \qquad \mu_w \in \{\mu_{w,CM}, \mu_{w,CO}\}$$

The above explained parameters and other additional parametrs which we will use in the rest of this paper are listed in Table 1.

Table 1. List of proposed system parameters.

Notation	Definition	Notation	Definition
T^t_{QoS}	Required task QoS at time t	QoS^t_w	Web service w QoS at time t
B^t_w	Budget associated to web service w at time t	Rep^t_w	Reputation assigned for w at time t
$Reward^t_w$	The reward to update the reputation	G^t_w	The growth factor of w at time t
ϵ	The community membership fee	$\pi^t_{w,CM}$	Competition payoff of w
$\pi^t_{w,CO}$	Cooperation payoff of w	$p^t_{w,CM}$	Competition probability of w at time t
$p^t_{w,CO}$	Cooperation probability of w at time t	COF^t_w	Cooperation fee of w at time t
β_w	Profit of w	$\mu_{w,CM}$	Mean service fee for competing w
$\mu_{w,CO}$	Mean service fee for cooperating w	τ^t_w	Coopetitive threshold of w at time t
P^t_w	Probability of competing for w at time t	U^t_w	Utility of w at time t

2.3 Web Service Interactive Strategies

The main goal of each individual web service is to increase its income (payoff). This income can be earned from tasks (or requests) done by this web service. In our model, web services can decide to compete to get a task from the master web service or to cooperate with other web services in a given collaborative network (the way a collaborative network is set by a leader is based on the cooperative web services reputation and their QoS parameters that should coincide with the required QoS). Therefore we define two types of web service strategies; when a web 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. On the other hand, when it has a lower level of confidence that it does not feel it can compete with other web services to get a task, the web service waits for some other web services to cooperate with for completing a task and thus it adopts the cooperative strategy. Web 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 web services can switch from one strategy to the other and this dynamic attitude is referred to as coopetition. We discuss about this decision making process in more details in the next section.

3 Theoretical Results

3.1 Web Service Decision Making Procedure

In this section, we explore in details the interaction strategies and the outcome of each strategy in terms of web services' utilities. The main part of web services' decision making procedure falls into their growth factor analysis. In fact, the growth factor and its comparison to a particular threshold is the main reason that influences the web service's decision to follow either competitive or cooperative behavior. Web services 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 web services' strategic decision making procedures.

Figure 2 shows the decision making process that is followed by a typical web service. In case the web service 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 web service is willing to cooperate, it has to wait for a cooperation opportunity that could be triggered by another web service that competed and obtained the task, so both web services will be part of the same collaborative network. In the decision making process presented in Figure 2, we assume that the competing web service might get the task (denoted as $Bid/obtainedTask$) or not get the task in case of being rejected by the master web service, 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 web services' status (competitive or cooperative) over different decision making rounds which could be caused by internal factors (the web services) or by the external factor (the master web service).

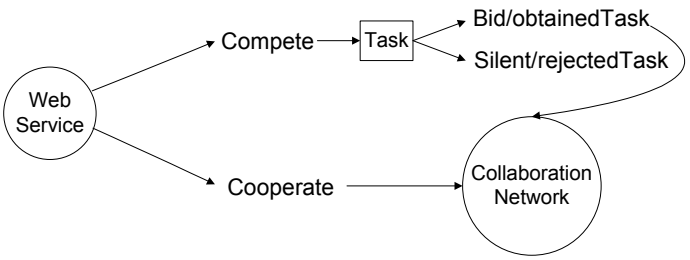

Fig. 2. Decision making process over competitive and cooperative strategies.

Consider a web service w that is willing to compete (that means the computed growth factor is more than the analyzed threshold). This web service can estimate the expected payoff associated to this decision, called competition payoff. Equation 4 computes this expected payoff for web service w ($\pi^t_{w,CM}$) considering the $Bid/obtainedTask$ probability of $p^t_{w,CM}$ and $Silent/rejectedTask$ probability of $1 - p^t_{w,CM}$.

$$\pi^t_{w,CM} = p^t_{w,CM}(\mu_{w,CM} - COF^t_w - \epsilon) + (1 - p^t_{w,CM})(-\epsilon) \tag{4}$$

In Equation 4, $\mu_{w,CM}$ is the mean service fee that is assigned by the master web service to w. This means that a competing web service w directly obtains this fee from the master web service. Moreover, the competing web service w expects a cooperation fee (COF^t_w) that it gives to its collaborators in case w needs to cooperate with other web services (cooperative web services in its collaboration network). In any case, the competing or cooperating web service pays a fixed amount of membership (ϵ) to the master web service as the coordinator of the community. This fee would be taken into account when a web 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.

Similar to the competitive web service case, if a web service w declares cooperative status, its expected cooperation payoff ($\pi^t_{w,CO}$) is computed in Equation 5. In this equation, $p^t_{w,CO}$ is the probability of getting involved in a cooperative task with other web services and $1 - p^t_{w,CO}$ is the probability of failure to find such a cooperation opportunity. These two probabilities are set when w decides to compete. We recall that $\mu_{w,CO}$ denotes the mean cooperation fee that is directly obtained from the leader (i.e., the competitive) web service of the underlying collaborative network. Compared to $\mu_{w,CM}$, $\mu_{w,CO}$ is relatively smaller since the competitive web service generally dedicates a portion of its obtained income to pay other cooperative web services.

$$\pi^t_{w,CO} = p^t_{w,CO}(\mu_{w,CO} - \epsilon) + (1 - p^t_{w,CO})(-\epsilon) \tag{5}$$

To analyze the expected payoffs obtained from different strategies, web services need to compute the estimated probabilities that distinguish subcases in each behavioral status ($p^t_{w,CM}$ for competitive and $p^t_{w,CO}$ for cooperative). To estimate these probabilities, we should notice that they are functions of web services' reputation values (Rep^t_w). Furthermore, $p^t_{w,CM}$ is also function of the difference between the offered QoS (QoS^t_w) and the requested one (T^t_{QoS}); and $p^t_{w,CO}$ is function of the reputation of other web services in the community because the leader is supposed to be selective when it comes time 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. The desired properties of $p^t_{w,CM}$ are as follows:

Property 1. $p^t_{w,CM}$ is continuous with regard to Rep^t_w, QoS^t_w, and T^t_{QoS}.

Property 2. $p^t_{w,CM}$ is monotonically increasing/decreasing in Rep^t_w and $QoS^t_w - T^t_{QoS}$ while $QoS^t_w - T^t_{QoS}$ is positive.

Property 3. $p^t_{w,CM}$ is null if $QoS^t_w - T^t_{QoS}$ is negative.

Property 4. The increase slope of $p^t_{w,CM}$ is higher when the reputation Rep^t_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 $p^t_{w,CM}$ can be always estimated as far as Rep^t_w, QoS^t_w, and T^t_{QoS} are available. Property 2 says that the reputation and QoS are two key factors that influence the value of $p^t_{w,CM}$ in the sense of positive correlation. Property 3 indicates that the probability $p^t_{w,CM}$ 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 $p^t_{w,CO}$ are as follows:

Property 5. $p^t_{w,CO}$ is continuous with regard to Rep^t_w and the reputation of other web services in the community.

Property 6. $p^t_{w,CO}$ is monotonically increasing in Rep^t_w and monotonically decreasing in the community average reputation.

Property 7. The increase slope of $p^t_{w,CO}$ is higher when the reputation Rep^t_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 web services have higher reputation. Property 7 is similar to Property 4.

Equations 6 and 7 respectively compute the estimated success probability in cases where the web service w is competing and cooperating. These values are computed considering web service's reputation value (Rep^t_w computed by the master), web service's offered QoS (QoS_w), the task required QoS (T^t_{QoS}) which is the mean required QoS computed from previous tasks, the maximum offered QoS (QoS^t_k, which is provided by another competitive web service k), and the cooperative factor CL^t_w of the web service w at time t, which is computed as the portion of web service's current reputation on the average reputation of the community.

$$p^t_{w,CM} = \begin{cases} \sin(Rep^t_w \frac{\pi}{2}) \frac{QoS^t_w - T^t_{QoS}}{Max_k(Qos^t_k - T^T_{QoS})} & \text{if } QoS^t_w \geq T^t_{QoS}; \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

$$p^t_{w,CO} = sin(Rep^t_w \frac{\pi}{2}) CL^t_w \tag{7}$$

$$CL^t_w = \frac{Rep^t_w}{\sum_{k \in Community} Rep^t_k / |Community|}$$

It is easy to prove that Equation 6 satisfies Property 1. The partial derivative $\frac{\partial p^t_{w,CM}}{\partial Rep^t_w}$ is positive as the function $\cos$ (the derivative of $\sin$) is positive on $[0, \frac{\pi}{2}]$ and $Rep^t_w \in [0, 1]$. The partial derivative $\partial p^t_{w,CM}$ with regard to $QoS^t_w - T^t_{QoS}$ is also positive, which proves the satisfaction of Property 2. Property 3 is straightforward. Finally, the increase slope of the function $\sin$ on $[0, \frac{\pi}{2}]$ proves Property 4.

Similarly, we can easily prove that Equation 7 satisfies Property 5. Property 6 can be shown by calculating the partial derivative $\partial p^t_{w,CO}$ first with regard to Rep^t_w and second with regard to the community average reputation $\sum_{k \in Community} Rep^t_k / |Community|$. The first partial derivative is positive and the second is negative, which proves the satisfaction of Property 6. The proof of satisfaction of Property 7 is similar to the one of Property 4.

3.2 Coopetition Threshold

In this part, we compute the coopetition threshold that a typical web service 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 web service w compare its growth factor (G^t_w) with the coopetition threshold it

holds at current time t (τ_w^t) and choose to compete with probability P_w^t that we compute in Equation 8. Based on this probability, we calculate the total utility U_w^t in Equation 9.

$$P_w^t = \begin{cases} \frac{G_w^t}{\tau_w^t} & \text{if } G_w^t \leq \tau_w^t; \\ 1 & \text{otherwise.} \end{cases} \tag{8}$$

$$U_w^t = P_w^t(\pi_{w,CM}^t) + (1 - P_w^t)(\pi_{w,CO}^t) \tag{9}$$

The key factor in the computation of the probability P_w^t and the associated utility is the threshold value. To compute the threshold, we use the game theoretic best response technique. A typical web service 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 the web service w could react with best response to further decision making procedures. We use the obtained conditions to compute the threshold τ_w^t at time t. By equalizing $\pi_{w,CM}^t$ and $\pi_{w,CO}^t$, we obtain:

$$\pi_{w,CM}^t = \pi_{w,CO}^t \rightarrow$$

$$p_{w,CM}^t(\mu_{w,CM} - COF_w^t - \epsilon) + (1 - p_{w,CM}^t)(-\epsilon) = p_{w,CO}^t(\mu_{w,CO} - \epsilon) + (1 - p_{w,CO}^t)(-\epsilon)$$

The fixed membership fee ϵ could be taken out and canceled from both sides of the equation so we obtain the following:

$$p_{w,CM}^t(\mu_{w,CM} - COF_w^t) = p_{w,CO}^t(\mu_{w,CO}) \rightarrow$$

$$sin(Rep_w^t \frac{\pi}{2}) \frac{QoS_w^t - T_{QoS}^t}{Max_k(Qos_k - T_{QoS}^t)}(\mu_{w,CM} - COF_w^t) = sin(Rep_w^t \frac{\pi}{2}) CL_w^t(\mu_{w,CO})$$

By simplifying the $sinus$ variable from both sides and substituting the cooperation factor CL_w^t of the web service w we obtain the following:

$$\frac{QoS_w^t - T_{QoS}^t}{Max_k(Qos_k^t - T_{QoS}^t)}(\mu_{w,CM} - COF_w^t) = \frac{Rep_w^t}{\sum_{k \in Community} Rep_k^t}(\mu_{w,CO})$$

We use the obtained equation to obtain the cooperation fee COF_w^t that is assigned by the web service w. This fee represents the amount that w spends to cooperate with other web service(s) to accomplish the task. By so doing, we obtain the maximum amount of cooperation fee that the web service w can use to constrain the positive payoff out of competing. Otherwise, the web service stays as cooperative entity. The cooperation fee is computed in Equation 10.

$$COF_w^t = \mu_{w,CM} - \frac{Rep_w^t}{\sum_{k \in Community} Rep_k^t} \frac{\mu_{w,CO} Max_k(Qos_k^t - T_{QoS}^t)}{QoS_w^t - T_{QoS}^t} \tag{10}$$

We use the maximum cooperation fee that a web service considers to constrain the positive expected payoff when the competitive strategy is adopted to update the threshold for the consequent time interval $(t + 1)$. We compare the maximum cooperation fee

with the required fee ($ReqF_w^t$) that the web service indicate to accomplish the task. The outcome of this comparison is a factor that uses the current threshold τ_w^t to compute the consequent threshold τ_w^{t+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). Equation 11 shows this computation. To investigate the effectiveness of this threshold on the outcomes of the web services that follow this reasoning technique, in the next section, we compare the results of different agents with diverse strategic reasoning techniques.

$$\tau_w^{t+1} = \begin{cases} \Theta & \text{if } 0 \leq \Theta \leq 1 \\ 1 & \text{if } \Theta > 1; \\ 0 & \text{if } \Theta < 0. \end{cases} \tag{11}$$

$$\Theta = \tau_w^t \times \frac{COF_w^t}{ReqF_w^t}$$

4 Experimental Results

In this section, we provide an empirical analysis over the observed results regarding the characteristics of intelligent web services hosted in different communities of web services. In the implemented system, we simulate the behaviors of service consumers as request generators, web services as service providers, and master web services 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 web services' overall budget and also the average quality and quantity of tasks performed by community web services which directly affects user satisfaction. To verify these objectives, we study the overall performance of the community hosting the reasoning-empowered web services compared to the ones hosting stochastic and purely competitive services. The simulation application is written in *C#* using *Visual Studio*. Developed web services are initialized with values taken from a real dataset that includes 2507 real web services functioning on the web. The dataset records the QoS values of 9 parameters including *availability*, *throughput*, and *reliability* [2].

We start our discussions with cumulative budget comparison regarding different communities within which services follow different reasoning techniques. Figure 3 part (a) illustrates three graphs for three different communities. Each community hosts web 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 random reasoning technique so decisions about selecting competitive or cooperative strategies are totally random (referred to as random coopetitive); 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 web 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 web services to follow cooperative strategies that overall maintain an optimal community budget. In the same Figure, 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 analyses 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 the last 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.

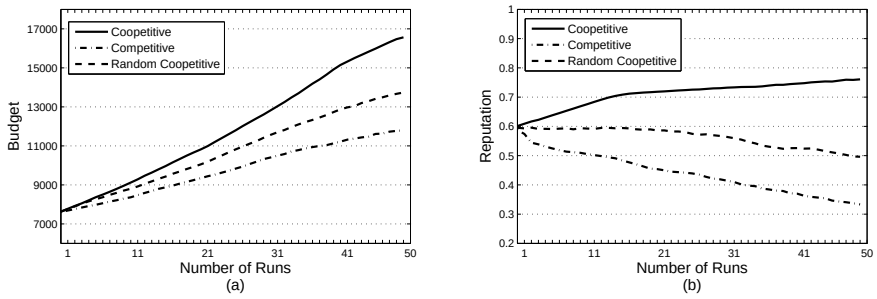

Fig. 3. Part (a): Cumulative community budget comparison. Part (b): Average community reputation comparison over different strategic decisions.

The results illustrated in Figure 3 part (a) verify the importance of the strategic decision making procedure to logically decide over the possible competitive and cooperative choices. Figure 3 part (b) illustrates communities average reputation of involved web services. The graphs represent the influence of the rewards that the master web service imposes to encourage highly capable web 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 web 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 for the coopetitive community.

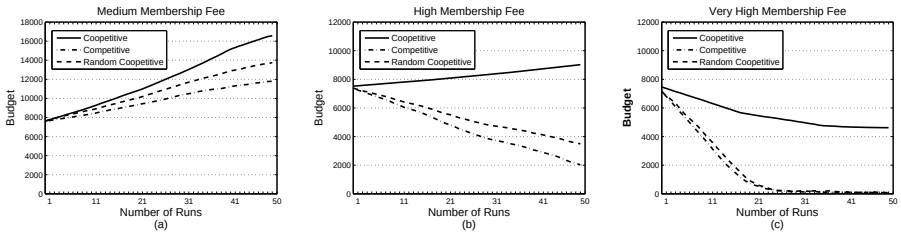

Fig. 4. High membership fees or less task income impacts on different selection strategies, (a) medium membership fee, (b) high membership fee, (c) very high membership fee

In Figure 4 we analyze the scenario where membership fee is high which is analogous to situations where task income is low. For any reason our web services may need to stay in a particular community with economic downtimes. We need verify how well the web services can perform in these situations. 4 part (a) depicts the results in normal scenario. In part (b) we have increased the membership fee to the high amount of 5 units per run. Web services will need to pay 5 units on each iteration of tasks being offered. As results the strategic coopetitive decision making process helps web services to still have benefit and random or all compete methods will lose a lot in these scenarios since making mistakes and trying to compete for jobs they cannot afford while having to pay higher membership fees is making them lose budget in a faster pace. In part 4(c) the membership is higher than average of possible income of what tasks offer. It might not be rational for selfish self-interest web services to stay in these communities from utility point of view but as mentioned there might be other political reasons they might need to stay in these communities. As results show the random and all compete strategies make web services bank deplete in no time. However our strategic coopetitive decision making process loses budget in a slower pace.

In Figure 5 parts (a) and (b), we observe the competitive and cooperative probability of four different web services where two of them (w_1 and w_3) are following optimal strategies (competitive for w_1 and cooperative for w_3) and the two others (w_2 and w_4) are following non-optimal strategies. Over elapsing runs, web services that follow optimal strategies bring best budget. In fact, the master web service rewards the high quality web service that chooses the competitive strategy, cooperates with other web services and successfully accomplishes the task. In this system, the reputation regarding such a web service is increasing over time and the possibility of allocating further tasks is increasing as well. By increasing the growth factor, such a web service (here shown as w_1) increases the probability of selecting the competitive strategy. On the other hand, the other web service (here shown as w_2) that is incapable of competing is penalized by the master web service 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 web 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 Figure 5 part (b) regarding web services w_3 and w_4 where unlike w_4, w_3 is strategically following the cooperative strategy. Therefore, w_4 is more seeking the competitive strategy where it can increase its growth factor.

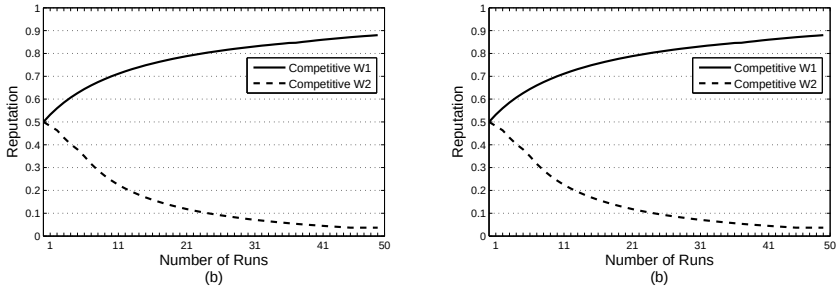

Fig. 5. Competitive and cooperative probabilities regarding four different web services.

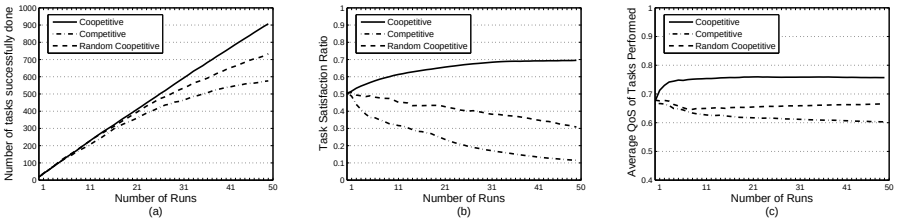

Fig. 6. Overall performance from comunity point of view, (a) Total Number of tasks succesfuly done, (b) Ratio of tasks satisfied with required QoS, (c) Average QoS of performed tasks

The web services are selfish, self interested agents. They try to maximise their own utiliy. Now we analyse how their strategy effects the social welfare or from user and community point of view how good the tasks are being performed. This directly affects user satisfation and community reputation in general. The Higher quality and quantity of tasks perfromed leads to higher user satisfaction for the community which results in better reputaiton for the community. The results in Figure 6 shows the quality of quantity of tasks being done successfuly in three communities adopting the three different mentioned strategy decision algorithms. Figure 6 Part (a) and (b) depicts the number tasks succesfuly done. When web 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 how ever if they opt for cooperating although they have less income, they

have higher chance of performing tasks collabratively with other web serives. Figure 6Part (b) shows the probablilty of successfully performin tasks. As results suggest web services using our coopetitive strategy when deciding whether to compete or cooperate have higher chance of tasks to be done than random or all compete strategies. The quality of tasks performed are depicted in Figure 6 Part (c). It shows the average QoS of tasks perfromed which are finished and done sucessfuly. Colabrating increases the performed QoS of tasks, therefore forming collabrative groups improves user satisfaction. As results suggest, the coopetetive community outperforms the random and all compete communities in quality of tasks performed.

We conclude out analysis by discussing how effective our cooperative decision making process is by comparing the final utility of our agents here the end budget our web services have while deviating from our strategy adopted by an agent which can be either compete or cooperate to the other one. In 7 part (a) we randomly made the web services deviate from the calculated strategy. As the results depict, the more web services deviate from cooperative strategy the more they less budget they gain. Figure 7 part (b) we pick one random web service and once simulate the scenario with its default cooperative strategy and then we redo the simulation with exact same enviroment parameters while adapting the other strategy which is to compete if our cooperative strategy suggested to cooperate or vice versa. We run this scenario 50 times and at the end we check their budget and see if they gain more by deviating or not. We did this for one single deviation to ten different deviations in lifetime of a web service in community. As results indicate with high probability our cooperative strategy yields higher budgets at the end for web services.

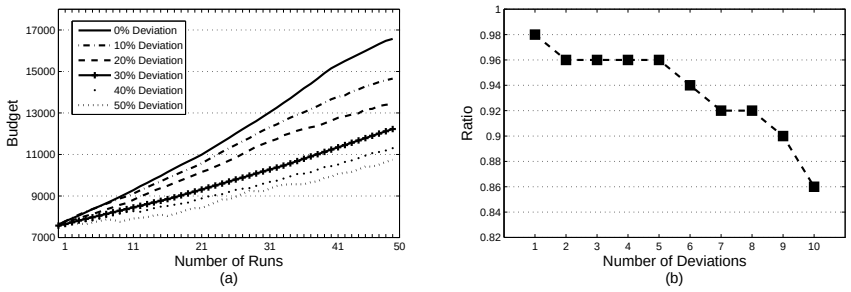

Fig. 7. Utiliy loss while deviating from cooperating decision process, (a) Overall budget when deviating in 0, 10, 20, 30, 40, 50 percent of decision, (b) Ratio of getting earning utility (budget) when deviating from cooperative strategy in 1 to 10 decisions

5 Related Work and Conclusion

Relevant proposals to the model presented in this paper are the ones that address service selection and task allocation mechanisms. In many frameworks proposed in the litera-

ture, service selection and task allocation are regulated based on the reputation parameter [3, 11, 18, 19]. In [7], the authors propose a framework to match potential benefits of web services while cooperating with one another. The interesting idea is to consider the benefits under four categories: innovation and learning, internal business process, customer, and financial benefits. In [1], the authors present a dependable framework for cooperative web services 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 web services. However, the opposite strategy of competing is not analyzed where web services might be more successful when competing within a same group. In fact, web 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, web services 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.

The contribution of this paper is the proposition of a coopetitive strategic model to analyze the interacting behavior of intelligent web services that are active within communities. We considered two acting strategies where web services expect different sort of payoffs: (1) competitive strategy where the web 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 web service does not take the responsibility to accomplish the task and only cooperates with other competitive web services. Our proposed model advances the state-of-the-art in cooperative systems by enabling intelligent web services to effectively choose their interacting strategies that lead to optimal outcomes. The proposed framework provides a reasoning technique that web services can use to increase their overall obtained utilities. The theoretical results presented in this paper are also backed by simulation results using a real web services dataset. 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. 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

1. E.A.P. Alchieri, A.N. Bessani, and J. S. Fraga. A dependable infrastructure for cooperative web services coordination. Proc. of the Int. Conf. on Web Services (ICWS), pp. 21-28, 2008.
2. E. Al-Masri and Q.H. Mahmoud. Discovering the best web service. Proc. of the 16th int. conf. on World Wide Web (WWW), pp. 1257-1258, 2007.
3. E.M. Maximilien. Multiagent system for dynamic web services selection. Proc. of the 1st Workshop on Service-Oriented Computing and Agent-based Eng., pp. 25-29, 2005.
4. E. Lim, P. Thiran, Z. Maamar, and J. Bentahar. On the analysis of satisfaction for web services selection. Proc. of the 9th Int. Conf. on Services Computing (SCC), 2012.
5. C. Ferguson and C. Gawargy. U(0,1) Two-person poker models. Game Theory and Applications, 12:17-37, 2007.
6. T. Ferguson, L. Shapley and R. WeberNotes. Notes on a stochastic game with information structure. Journal of Game Theory, 31: 223-228, 2003.

7. C.D. Huang, Q. Hu. Integrating web services with competitive strategies:The balanced scored approach. Communications of the Association for Information Systems, 13(1)57-80, 2004.
8. M. Jacyno, S. Bullock, M. Luck, and T.R. Payne. Emergent service provisioning and demand estimation through self-organizing agent communities. Proc. of the 8'th Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS), pp. 481-488, 2009.
9. R. Jurca and B. Faltings. Obtaining reliable feedback for sanctioning reputation mechanisms. Journal of Artificial Intelligence Research, 29:391-419, 2007.
10. R. Jurca and B. Faltings. Reputation-based service level agreements for Web services. Proc. of the Int. Conf. on Service Oriented Computing (ICSOC), Lecture Notes in CS, Volume 3826, pp. 396-409, 2005.
11. S. Kalepu, S. Krishnaswamy, S. W. Loke. A QoS metric for selecting Web services and providers. Proc. of the 4'th Int. Conference on Web Information Systems Engineering Workshops, pp. 131-139, 2003.
12. B. Khosravifar, J. Bentahar, A. Moazin, and P. Thiran. On the reputation of agent-based web services. Proc. of the 24'th Int. Conf. on Artificial Intelligence (AAAI), pp. 1352-1357, 2010.
13. B. Khosravifar, J. Bentahar, K. Clacens, C. Goffart, and P. Thiran. Game-theoretic analysis of a web services Collaborative mechanism. Proc. of the Int. Conf. on Service Oriented Computing (ICSOC), pp. 549-556, 2011.
14. B. Khosravifar, J. Bentahar, A. Moazin, and P. Thiran. Analyzing communities of web services using incentives. Journal of Web Services Research, 7(3):30-51, 2010.
15. Z. Malik and A. Bouguettaya. Evaluating rater credibility for reputation assessment of web services. Proc. of the 8'th Int. Conf. on Web Information Systems Engineering (WISE), pp. 38-49, 2007.
16. E.M. Maximilien, and M.P. Singh. Reputation and endorsement for web services, ACM SIGEcom Exchanges, 3(1):24-31, 2002.
17. T. Rahwan, T. Michalak, E. Elkind, P. Faliszewski, J. Sroka, M. Wooldridge, and N. Jennings. Constrained coalition formation. Proc. of The 25'th Int. Conf. on Artificial Intelligence (AAAI), pp. 719-725, 2011.
18. S. Rosario, A. Benveniste, S. Haar, and C. Jard. Probabilistic QoS and soft contracts for transaction based Web services. Proc. of the Int. Conf. on Web Services (ICWS), pp. 126-133, 2007.
19. M. Ruth and T. Shengru. Concurrency issues in automating RTS for web services. Proc. of the Int. Conf. on Web Services (ICWS), pp. 1142-1143, 2007.
20. A. Yassine, A.A. Shirehjini, S. Shirmohammadi, and T. Tran. Knowledge-empowered agent information system for privecy payoff in ecommerce. Knowledge and Information Systems, DOI: 10.1007/s10115-011-0415-3, 2011.