QoS-based Service Selection and Ranking with Trust and Reputation Management[*]

Le-Hung Vu, Manfred Hauswirth and Karl Aberer

School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
{lehung.vu, manfred.hauswirth, karl.aberer}@epfl.ch

Abstract. QoS-based service selection mechanisms will play an essential role in service-oriented architectures, as e-Business applications want to use services that most accurately meet their requirements. Standard approaches in this field typically are based on the prediction of services' performance from the quality advertised by providers as well as from feedback of users on the actual levels of QoS delivered to them. The key issue in this setting is to detect and deal with false ratings by dishonest providers and users, which has only received limited attention so far. In this paper, we present a new QoS-based semantic web service selection and ranking solution with the application of a trust and reputation management method to address this problem. We will give a formal description of our approach and validate it with experiments which demonstrate that our solution yields high-quality results under various realistic cheating behaviors.

1 Introduction

One key issue in the Semantic Web Service area is to discover the most relevant services meeting the functional requirements of users. Equally important, e-Business applications also would like to discover services which best meet their requirements in terms of QoS, i.e., performance, throughput, reliability, availability, trust, etc. Thus QoS-based web service selection and ranking mechanisms will play an essential role in service-oriented architectures, especially when the semantic matchmaking process returns lots of services with comparable functionalities.

In this paper we present a QoS-based web service selection and ranking approach which uses trust and reputation evaluation techniques to predict the

[*] The work presented in this paper was (partly) carried out in the framework of the EPFL Center for Global Computing and was supported by the Swiss National Funding Agency OFES as part of the European project DIP (Data, Information, and Process Integration with Semantic Web Services) No 507483. Le-Hung Vu is supported by a scholarship of the Swiss federal government for foreign students. We also thank Zoran Despotovic, Amit Lakhani and the anonymous reviewers for their carefully reading and commenting this paper.

future quality of a service. Our work is based on requirements from a real-world case study on virtual Internet service providers (VISP) from an industrial partner in one of our projects [1]. In a nutshell, the idea behind the VISP business model is that Internet Service Providers (ISPs) describe their services as semantic web services, including QoS such as availability, acceptable response time, throughput, etc., and a company interested in providing Internet access, i.e., becoming a VISP, can look for its desired combination of services taking into account its QoS and budgeting requirements, and combine them into a new (virtual) product which can then be sold on the market. This business model already exists, but is supported completely manually. Since many ISPs can provide the basic services at different levels and with various pricing models, dishonest providers could claim arbitrary QoS properties to attract interested parties. The standard way to prevent this is to allow users to evaluate a service and provide feedbacks. However, the feedback mechanism has to ensure that false ratings, for example, badmouthing about a competitor's service or pushing one's own rating level by fake reports or collusion with other malicious parties, can be detected and dealt with. Consequently, a good service discovery engine would have to take into account not only the functional suitability of services but also their prospective quality offered to end-users by assessing the trustworthiness of both providers and consumer reports. According to several empirical studies [1, 2], the issue of evaluating the credibility of user reports is one of the essential problems to be solved in the e-Business application area.

We develop the QoS-based service selection algorithm under two basic assumptions which are very reasonable and realistic in e-Business settings: First, we assume *probabilistic behavior of services and users*. This implies that the differences between the real quality conformance which users obtained and the QoS values they report follow certain probability distributions. These differences vary depending on whether users are honest or cheating as well as on the level of changes in their behaviors. Secondly, we presume that *there exist a few trusted third parties*. These well-known trusted agents always produce credible QoS reports and are used as trustworthy information sources to evaluate the behaviors of the other users. In reality, companies managing the service searching engines can deploy special applications themselves to obtain their own experience on QoS of some specific web services. Alternatively, they can also hire third party companies to do these QoS monitoring tasks for them. In contrast to other models [3–7] we do not deploy these agents to collect performance data of all available services in the registry. Instead, we only use a small number of them to monitor QoS of some selected services because such special agents are usually costly to setup and maintain.

The QoS-based service selection and ranking algorithm we describe in this paper is a part of our overall distributed service discovery approach [8]. During the service discovery phase, after the functional matchmaking at a specific registry, we would obtain a list of web services with similar functionalities from the matchmaking of our framework, i.e., the services fulfilling all user's functional

[1] DIP Integrated Project, http://dip.semanticweb.org/

requirements. We need to select and rank these services based on their predicted QoS values, taking into consideration the explicit quality requirements of users in the queries. The output of the selection and ranking algorithm is the list of web services fulfilling all quality requirements of a user, ordered by their prospective levels of satisfaction of the given QoS criteria. So as to perform this selection and ranking accurately, we collect user reports on QoS of all services over time to predict their future quality. This prediction is also based on the quality promised by the service providers as well as takes into consideration trust and reputation issues.

The major contribution of our work is a new QoS-based web service selection and ranking approach which is expected to be accurate, efficient and reliable. First, we have taken into account the issue of trust and reputation management adequately when predicting service performance and ranking web services based on their past QoS data. Experimental results have shown that the newly proposed service selection and ranking algorithm yields very good results under various cheating behaviors of users, which is mainly due to the fact that *our use of trusted third parties observing a relatively small fraction of services can greatly improve the detection of dishonest behavior even in extremely hostile environments.* This is particularly important as without cheating detection, service providers will be likely to generate lots of false reports in order to obtain higher ranks in the searching results, thereby having higher probability to be selected by clients and gain more profits. Second, our algorithm is semantic-enabled by offering support for the semantic similarity among QoS concepts advertised by providers and the ones required by users. This allows the QoS-based service selection process to work more flexibly and produce more accurate results. Third, we adapt the idea of Srinivasan et al [9] to pre-compute all matching information between QoS capabilities of published services and possible QoS requirements of users to avoid time-consuming reasoning and to minimize the searching costs.

The rest of this paper is organized as follows. First, we briefly mention the related work in section 2. Section 3 presents our trust and reputation management model in a Web Service Discovery scenario. Our QoS-based service selection and ranking approach is described in detail in section 4. We discuss various experimental results in section 5 and conclude the paper in section 6.

2 Related Work

Although the traditional UDDI standard [2] does not refer to QoS for web services, many proposals have been devised to extend the original model and describe web services' quality capabilities, e.g., QML, WSLA and WSOL [10]. The issue of trust and reputation management in Internet-based applications has also been a well-studied problem [1, 2].

The UX architecture [11] suggests using dedicated servers to collect feedback of consumers and then predict the future performance of published services.

[2] Latest UDDI Version (3.0.2), http://uddi.org/pubs/uddi-v3.0.2-20041019.htm

[12] proposes an extended implementation of the UDDI standard to store QoS data submitted by either service providers or consumers and suggests a special query language (SWSQL) to manipulate, publish, rate and select these data from repository. Kalepu et al [13] evaluate the reputation of a service as a function of three factors: ratings made by users, service quality compliance, and its verity, i.e., the changes of service quality conformance over time. However, these solutions do not take into account the trustworthiness of QoS reports produced by users, which is important to assure the accuracy of the QoS-based web service selection and ranking results. [14] rates services computationally in terms of their quality performance from QoS information provided by monitoring services and users. The authors also employ a simple approach of reputation management by identifying every requester to avoid report flooding. In [15], services are allowed to vote for quality and trustworthiness of each other and the service discovery engine utilizes the concept of distinct sum count in sketch theory to compute the QoS reputation for every service. However, these reputation management techniques are still simple and not robust against various cheating behaviors, e.g., collusion among providers and reporters with varying actions over time. Consequently, the quality of the searching results of those discovery systems will not be assured if there are lots of colluding, dishonest users trying to boost the quality of their own services and badmouth about the other ones. [16] suggests augmenting service clients with QoS monitoring, analysis, and selection capabilities, which is a bit unrealistic as each service consumer would have to take the heavy processing role of a discovery and reputation system. Other solutions [3–7] use third-party service brokers or specialized monitoring agents to collect performance of all available services in registries, which would be expensive in reality. Though [7] also raises the issue of accountability of Web Service Agent Proxies in their system, the evaluation of trust and reputation for these agents is still an open problem.

Our QoS provisioning model is grounded on previous work of [3, 4, 11, 13, 14]. The trust and reputation management of our work is most similar to that of [17, 18] but we employ the idea of distrust propagation more accurately by observing that trust information from a user report can also be used to reveal dishonesty of other reporters and by allowing this distrust to be propagated to similar ones. Other ideas of the trust and reputation management method are based on [19–21].

3 A Trust and Reputation Management Model for QoS-based Service Discovery

The interaction among agents in our system is represented in Fig. 1 where S_0,..,S_n are web services, U_0,...,U_m are service users, RP_0,...,RP_k are service registries in a P2P-based repository network, based on our P-Grid structured overlay [22]. T_0,...,T_r are the trusted QoS monitoring agents from which we will collect trustworthy QoS data to use in our algorithm.

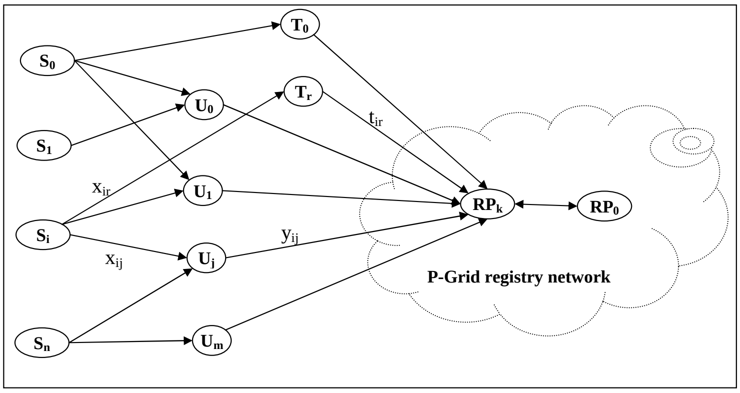

Fig. 1. Interaction among agents in a QoS-based service discovery scenario

After U_j's perception of a normalized QoS conformance value x_{ij} (a real number from Definition 1 in the following section) from a service S_i, it may report the value y_{ij} to the registry RP_k, which manages the description information of S_i. This registry peer will collect users' quality feedbacks on all of its managed web services to predict their future performance and support the QoS-enabled service discovery based on these data. Note that a user generally reports a vector of values representing its perception of various quality parameters from a service. Also, x_{ij}s and y_{ij}s are QoS conformance values, which already take into account the quality values advertised by providers (see Definition 1 in the next section). Since the prediction of QoS in our work mostly depends on reports of users to evaluate service performance, the cheating behaviors of providers are not explicitly mentioned henceforth. In this paper, we only consider the selection and ranking of services with reputation management in one registry peer. The study of interaction among different registries is subject to future work.

Given the above interaction model, we can make a number of observations. An honest user will report $y_{ij} = x_{ij}$ in most cases (in reality, they may not report if not willing to do so). On the other hand, a dishonest reporter who wants to boost the performance of S_i will submit a value $y_{ij} > x_{ij}$. Similarly, cheating reports generated by S_i's competitors will report $y_{ij} < x_{ij}$ to lower its QoS reputation. In addition, (colluding) liars may sometimes provide honest reports and behave dishonestly in other cases [2]. Accordingly, we presume that the differences between y_{ij}s and x_{ij}s follow certain distribution types which *expected values* depend on the behavior of the user, i.e., are equal to 0.0 for honest users and different from 0.0 in the case of liars. We use this assumption since in general dishonest users are likely to change their actions over time in order to hide their cheating behaviors with *occasionally* credible reports. Moreover, as the quality parameter values of a service really depend on many environmental factors, even trusted agents and honest users may obtain and report those values a little different to each other when talking about the same service. However, the expected value of trustworthy reports on QoS of S_i, e.g., the average of corresponding credible y_{ij} values, will reflect its real quality capability. In our opinion, the expected values of these distributions, e.g., means, normally reveal the behaviors of users, whereas other parameters, e.g., standard deviations, will represent the uncertainty in actions of users, either accidentally or by purpose. Our goal is not only to evaluate whether a user is honest but also to compute the expected conformance values from the reports of the most honest users, e.g., the average of values y_{ij}s by the most credible reporters, from which we will predict the future quality of S_is.

4 QoS-based Service Selection and Ranking

Our QoS-enabled distributed service discovery framework is presented in detail in [8]. Quality properties of web services are described by concepts from a QoS ontology and then embedded into service description files using techniques suggested by WS-QoS [3] and Ran [4]. The value of a quality parameter of a web service is supposed to be *normalized* to a non-negative real-valued number regarding service-specific and call-specific context information where *higher normalized values represent higher levels of service performance*. We are aware that the issue of normalizing the values of various quality attributes is complicated, but this is out of the scope of our current study. However, with most frequently used QoS concepts, e.g., reliability, execution-time, response-time, availability, etc., the answer is well-defined and straight-forward. For instance, a web service with a normalized QoS parameter value for reliability of 0.90 will be considered as more reliable to another one with a normalized reliability value of 0.50. In this case the normalized reliability is measured as its degree of being capable of maintaining the service and service quality over a time period T. The ontology language we use to describe service semantics and to define the QoS ontology is WSMO [3], but other models, e.g., OWL-S [4], would also be applicable. For experimental evaluations, we have developed a simple QoS ontology for the VISP use case including the most relevant quality parameters for many applications, i.e., availability, reliability, execution time, etc. We currently assume that users and providers share a common ontology to describe various QoS concepts. However, this could be relaxed with the help of many existing ontology mapping frameworks.

4.1 Predicting Service Performance

In order to predict the quality of a web service S_i, we collect all QoS feedbacks on its performance over a time period W and use a *real-valued time series forecasting technique* to predict its future quality conformance from past data. To understand the concepts in our algorithms we start with two fundamental definitions.

[3] http://www.wmso.org/

[4] http://www.daml.org/services/owl-s/

Definition 1. The quality conformance value c_{ij}^k of a service S_i in providing a quality attribute q_{ij} at time k is defined as $c_{ij}^k = \frac{d_{ij}^k - p_{ij}^k}{p_{ij}^k}$ where d_{ij}^k is the normalized value of q_{ij} that S_i actually delivered to a specific user at time k and p_{ij}^k is the corresponding normalized QoS value promised by S_i's provider at that time.

Definition 2. A user QoS report R, either by a normal service consumer or by a trusted monitoring agent, is a vector $\{u, S_i, t, L\}$, where u is the identifier of the user that produced this report, S_i is the corresponding web service, t is the timestamp of the report and L is a quality conformance vector of $\{q_{ij}, c_{ij}^t\}$ pair values, with q_{ij} being a QoS attribute offered by S_i and c_{ij}^t being q_{ij}'s quality conformance that S_i provides to this user at time t.

In order to filter out as much dishonest reports as possible and to take only the most credible ones in the QoS predicting process, we apply our trust and reputation management techniques comprising of two steps: a report preprocessing and a report clustering phase. The first phase evaluates the credibility of collected user reports by applying a trust-and-distrust propagation approach, which relies on some initial trusted reports produced by special monitoring agents. We consider two QoS reports as *comparable* if they are related to the same service during a specific time interval δ_t and as *incomparable* otherwise. Generally, we can set this δ_t as big as the length of the period during which the corresponding service provider does not change the promised quality values of this service. Two *comparable* QoS reports are considered to be *similar* if the squared Euclidean distance between their conformance vectors is less than a specific threshold. On the contrary, they are regarded as *dissimilar* if this distance is greater than another threshold value.

The report preprocessing step is done according to Algorithm 1. n_{ch} and n_{h1}, n_{h2} $(n_{h1} < n_{h2})$ are threshold values to estimate a user as cheating or honest regarding to the similarity of its reports to other cheating/honest ones (line $9, 17$ and 18). N and T represent for how long and how frequent a user stay in and submit QoS reports to the system. The values of $n_{h1}, n_{h2}, n_{ch}, N$ and T are design parameters to be chosen depending on properties of the collected reports after running the algorithm several times. Generally, higher values of these parameters stand for higher level of caution when estimating behaviors of users regarding current evidences against them.

After finishing the preprocessing phase, we can identify a certain number of cheaters and honest users. However, this trust-distrust propagation phase may not be able to evaluate the credibility of all reports in case the user communities of certain services are isolated from other communities as well as if we set the values of n_{h1}, n_{h2} and n_{ch} too high in Algorithm 1.

Therefore, in the next step we have to estimate the trustworthiness of the remaining reports of which credibility has not been evaluated. To achieve this, we reason that colluding cheating users will cooperate with each other in order to influence the system with their dishonest feedbacks. As a result, users within each group will produce similar values and naturally form different clusters of

Algorithm 1 QosReportsPreprocessing()

```
all trusted agents are marked as honest users;
all reports of trusted agents are marked honest;
repeat
    all unmarked reports of each cheating user are marked cheating;
    for each unmarked report do
        if this report is dissimilar from an honest report then mark it cheating;
        if this report is similar with a cheating report then mark it cheating;
    end for
    users with at least $n_{ch}$ reports similar with cheating ones are marked cheating;
    users with at least $N$ reports in at least $T$ different time are marked stable;
until there is no new cheating user discovered;
repeat
    all unmarked reports of each honest user are marked as honest;
    for each unmarked report and each report marked cheating do
        if this report is similar with an honest report then mark it honest;
    end for
    unmarked users with at least $n_{h1}$ reports similar with honest ones are marked
      honest;
    users marked as cheating and having at least $n_{h2}$ reports similar with honest
      ones are re-marked honest;
until there is no new honest user discovered;
```

reports. Thus it is possible to apply data-mining techniques in this situation to discover various existing clusters of reports related to those user groups. In our work, we apply the *convex k-mean clustering algorithm* on each set of QoS reports related to a service during the time interval δ_t with the following metrics: The distance between two *comparable* reports is defined as the Euclidean squared distance between two corresponding quality conformance vectors and the distance between two *incomparable* ones is assigned a large enough value so that these reports will belong to different clusters.

After the trust and reputation evaluation in the two above phases, for each service S_i, we will have a list G_i of groups of reports on QoS of S_i over time. Generally, G_i includes the groups containing those reports that were previously marked as honest/cheating during the trust-distrust propagation phase, as well as other clusters of reports obtained after the clustering step. We will assign the credibility w_i^g of a report group $g_i \in G_i$ as follows. Firstly, we filter out all dishonest ratings by assign $w_i^g = 0.0$ for all groups of reports marked as cheating during the trust-distrust propagation. If there exists the group g_i^0 of reports previously marked honest during that step, we assign $w_i^{g_0} = 1.0$ whereas letting $w_i^g = 0.0$ for the remaining groups. Otherwise, we try to compute w_i^g so that this value would be proportional to the probability that the group g_i is trustworthy among all of the others. Our heuristic is to assign higher weight to clusters which are populated more densely, having bigger size and with more stable users. This assumption is reasonable, as the reports of independent cheaters are likely to be scattered, and in the case liars cooperate with each other to cheat the system, the size of their corresponding clusters will not exceed those consisting only of honest reports as it would be too costly to dominate the system with numerous and clustered dishonest ratings. Even if dishonest providers try to produce lots of more condense reports so that they could get high influences to the final ranking results at any cost, these values will be separated from honestly reported values and therefore are likely to be discovered during the distrust-propagation phase (line 3 to line 11 in Algorithm 1), provided we have enough trustworthy reports to use. Specifically, w_i^g could be estimated based on the following information: the number of users in the cluster g_i ($size_i^g$), the number of all users producing reports in all clusters of G_i ($allusers_i$), the number of stable users in this cluster ($stable_i^g$), the total number of stable users in all clusters of G_i ($allstable_i$), as well as the average distance d_i^g from the member reports of cluster g_i to its centroid values. Based on our model in section 3, the credibility of one report would depend on the distance between its conformance vector and that of an honest report. Therefore, the similarity among credibility of different reports in one cluster g_i would be inversely proportional to its d_i^g value. Furthermore, a report in g_i would be honest in two cases: (1) it is submitted by a *stable and honest* user; (2) it is produced by an *unstable and honest* user. Let P_{stbl} and P_{unstbl} be the probability that this report is of a stable user and of an unstable user, respectively, and let P_{stblcr} and $P_{unstblcr}$ be the probability that stable and unstable users report credibly, then we have $w_i^g = \frac{C}{d_i^g}.(P_{stbl}.P_{stblcr} + P_{unstbl}.P_{unstblcr})$, where $P_{stbl} = \frac{stable_i^g}{size_i^g}$ and $P_{unstbl} = 1 - P_{stbl}$. P_{stblcr} and $P_{unstblcr}$ could be estimated by comparing reports of trusted agents with those of sample stable/unstable users to derive an appropriate value at the whole network level. The value of C represents our belief in the relation between the density of a report cluster and the credibility of its members, which is considered as parameters of the reputation system and to be set by experience.

The future quality conformance $\hat{C}_{ij}$ of a service S_i in providing a QoS attribute q_{ij} is predicted using a linear regression method, thus we have: $\hat{C}_{ij} = LinearRegression(\overline{C}_{ij}^t)$, $t \in \{0, \delta_t, \ldots, (W-1).\delta_t\}$, where $\overline{C}_{ij}^t$ is the evaluated QoS conformance value of the quality parameter q_{ij} at time t. We compute $\overline{C}_{ij}^t$ as the average of conformance values reported by various user groups in the system at that specific time point, using the evaluated credibility w_i^gs as weights in the computation. In other word, $\overline{C}_{ij}^t = \frac{\sum_{g_i \in G_i} w_i^g C_{ij}^t}{\sum_{g_i \in G_i} w_i^g}$ where C_{ij}^t is the mean of conformances of a report group g_i on S_i's quality attribute q_{ij} at time t, i.e., a centroid value of a cluster/group of reports produced after the trust and reputation evaluation phase. Regarding the probabilistic behavior of users and services as in section 3, we consider $\hat{C}_{ij}$ as an approximate estimate of the expected value of S_i's QoS conformance in providing quality attribute q_{ij} to users.

4.2 QoS-based Service Selection and Ranking

We define QoS requirements in a user query as a vector Q of triples $\{q_j, n_j, v_j\}$ where q_j represents for the required QoS attribute, n_j is the level of importance of this quality attribute to the user and v_j is the minimal delivered QoS value that this user requires. To rank services according to its prospective level of satisfying user's QoS requirements, we utilize the Simple Additive Weighting method, which produces ranking results very close to those of more sophisticated Decision Making techniques [5]. Thus, the QoS rank of a service S_i in fulfilling all quality criteria depends on the weighted sum $T_i = \frac{\sum_{q_j \in Q} n_j.P_{ij}}{\sum_{q_j \in Q} n_j}$ in which $P_{ij} = w_{ij}.\widehat{nd}_{ij}$ represents the capability of S_i in providing the QoS concept q_{ij} for users at the query time. The value $\widehat{nd}_{ij} = \frac{\hat{d}_{ij} - v_j}{v_j}$ evaluates the difference between the QoS value $\hat{d}_{ij}$ of the quality attribute q_{ij} that S_i is be able to offer to its users according to the prediction and the corresponding value v_j of q_j required by the service query. In combination with Definition 1, we can compute $\widehat{nd}_{ij} = \frac{(1+p_{ij}).\hat{C}_{ij} - v_j}{v_j}$, where $\hat{C}_{ij}$ is the predicted QoS conformance value of quality attribute q_{ij} and p_{ij} is the corresponding QoS value promised by provider of S_i at current time. w_{ij} is a weight proportional to the semantic similarity m_{ij} between q_{ij} and the QoS ontology concept q_j required by the user, i.e., the degree of match as in [23]. In other word, we will give higher ranks for services which offer the most accurate QoS concepts at the higher levels compared to the ones required by users. In our program we simple use the following definition to evaluate w_{ij}:

$$w_{ij} = \begin{cases} 1.0 \text{ if } m_{ij}{=}\mathit{exact}; \text{(i.e., } q_{ij} \text{ is equivalent to } q_j) \\ 0.5 \text{ if } m_{ij}{=}\mathit{pluggin}; \text{(i.e., } q_{ij} \text{ is more general than } q_j) \\ 0.0 \text{ if } m_{ij} \in \{\mathit{subsume}, \mathit{failed}\}; \text{(i.e., otherwise)} \end{cases} \tag{1}$$

In order to accelerate the selection of only services fulfilling all required QoS parameters, we use the idea of Srinivasan et al [9] to avoid the time-consuming semantic reasoning step. Specifically, we use a *QoS matching table* to store the matching information for all frequently accessed QoS attributes. With each QoS attribute q_j in this table, we have a list $Lqos_j$ of records $\{S_{ij}, w_{ij}, \hat{d}_{ij}\}$ where w_{ij}, $\hat{d}_{ij}$ are computed as above and S_{ij} identifies a service having certain support for q_j. Given the list L of services with similar functionalities, the discovery engine performs the QoS-based service selection and ranking process as in Algorithm 2.

5 Experimental Results and Discussions

To evaluate the selection and ranking algorithm, we have implemented it as a QoS support module in a registry peer of our distributed discovery framework [8] and studied its effectiveness under various settings. The service selection and ranking is performed on three representative quality parameters, namely *availability*, *reliability* and *execution-time* taken from the VISP case study.

We observed the dependency between the quality of selection and ranking results and other factors, such as the percentage of trusted users and reports, the rate of cheating users in the user society and the various behaviors of users.

Algorithm 2 QosSelectionRanking(ServiceList L, ServiceQuery Q)

```
Derive the list of QoS requirements in Q: $L_q = \{[q_1, n_1, v_1], ..., [q_s, n_s, v_s]\}$
Initialize $Score[S_{ij}] = 0.0$ for all services $S_{ij} \in L$;
for each quality concept $q_j \in L_q$ do
  for each service $S_{ij} \in L$ do
    Search the list $L_{qos}$ of $q_j$ for $S_{ij}$;
    if $S_{ij}$ is found then
      $Score[S_{ij}] = Score[S_{ij}] + \frac{n_j.w_{ij}}{\sum n_j}(\frac{\hat{d}_{ij} - v_j}{v_j})$;
    else
      Remove $S_{ij}$ from $L$;
    end if
  end for
end for
Return the list $L$ sorted in descending order by $Score[S_{ij}]$ s;
```

Specifically, we evaluated the effectiveness of the service discovery by studying the differences in the quality of results when running evaluations in four different settings: In the *ideal* case the discovery engine has complete knowledge, such that it knows all correct QoS conformance values of all published services in the system over a time window W and performs the selection and ranking of services from this ideal data set. In the *realistic* case, we ran our algorithm with the application of trust and management techniques to filter out incredible reports and to evaluate the credibility for the others. The *optimistic* case corresponds to the QoS-based discovery of services without regarding to trust and reputation issues, i.e., the system simply uses the average of all reported conformance values to predict services' performance and to perform the QoS ranking. The *naive* case corresponds to the selection of services based only on their QoS values promised by the providers, i.e., the system trusts all providers completely. Our goal was to show that the obtained results of the QoS-based service discovery process are more accurate and reliable in the realistic case with various cheating behaviors of users, they would be much worse in the optimistic case, and the worst with the naive method thus clearly showing the contribution of our approach.

The quality of selection results produced by our algorithm can be measured by four parameters, namely *recall*, *precision*, *R-precision* and *Top-K precision*, of which the *R-precision* is the most important quality parameter as recognized by the Information Retrieval community. In our settings, these parameters generally represent the fraction of services that are most relevant to a user among all returned services in terms of their *real* QoS capabilities. Apparently, the results would be the best in the ideal case, i.e., its recall, precision, R-precision and Top-K precision parameters are all equal to 1.0. Therefore, we use the results of the ideal case as a reference to compare with the quality parameters in the other three situations. Due to space limitations we only show the *R-precision* values as the most representative experimental results in this section. We also measured the other parameters as well as computed the absolute QoS ranks of returned services using weighting Spearman's footnote method and had similar results.

We prepared our experiments with the probabilistic assumption on the behavior of service providers and consumers. In this paper we only present the experiments with Gaussian (normal) distributions. The extension to other probabilistic distributions is subject to future work. The society of service consumers was modeled as a collection of different types of users. As mentioned in section 3, honest users and trusted agents would report values with the difference $D_h \sim Normal(0.0, {\sigma_h}^2)$ to the real QoS conformance capabilities of services. On the contrary, cheaters would report values with the difference $D_c \sim Normal(M_c, {\sigma_c}^2)$ to the real quality conformances that they had obtained. The values of the mean M_c varied according to the purpose of the cheaters, i.e., advertise or badmouth a service. The values of σ_c represented the variation in reported values of users among different quality attributes of different services. Users with higher values of σ_c had higher levels of inconsistency in their behaviors and therefore were harder to be detected. We further divided these liars into three sub-types: *badmouthing users* who mostly reported badly about services of their competitors, *advertising users* who usually exaggerated performance of their own services and *uncertain users* with indeterministic actions and who might act as advertising, badmouthing or even as honest users. We set the values of M_c for each type of cheaters in the most pessimistic situation, making our testing environment be very hostile. Specifically, cheaters had their corresponding M_cs set to high/low enough values such that badmouthing users would succeed in pushing services of their competitors out of the selection results and advertising users would be able to raise the QoS ranks of their own services, provided that their reports had been taken into the predicting process of the QoS-based service discovery engine. This setting is realistic because in business, companies generally have knowledge of the base requirements of their users as wells as owning certain statistics of their competitors' capabilities. More complicatedly, uncertain users had their M_cs values belonging to N_c specific values each of which was a randomized real value, with N_c was the number of groups of cheaters with varied behaviors. These types of liars would be harder to be detected since their M_c values were uniformly distributed around 0.0. Though they did not contribute directly to the boosting and lowering the reputation of certain services, their reports were not so dissimilar from honest reports in most cases and therefore they would act as good recommenders for other badmouthing/advertising guys. To assure the scalability of the approach, we also tested it with an increasing number of services, users and QoS reports while keeping the percentage of user types and other parameters the same. The remaining experiments were run in a society of 1000 users which produced a total of 50000 QoS reports on 200 services during a time window of length $W = 5$ and $\delta_t = 1$. The results of each experiment were averaged over 10 runs.

As a first question, we wanted to study the effects of the trusted reports on the quality of results in the realistic case. Specifically, we wanted to observe the effects of the percentage of the services monitored by trusted agents $F_{special}$ to the results of our QoS-based service selection and ranking algorithm expressed by *R-Precision* values. We increased $F_{special}$ from 1.0% to 10.0%. The percentage of trusted users/reports was also increased from 0.1% to 1.0% with the increment of 0.1% each step as well. The results of this experiment are shown in Fig. 2.

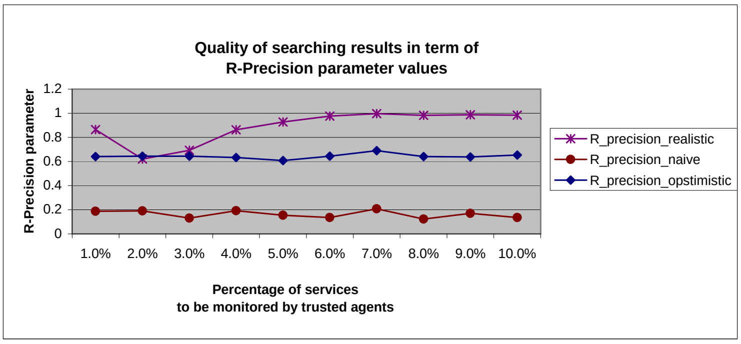

Fig. 2. $F_{special}$ vs. *R-Precision*

Correspondingly, Fig. 3 shows the percentage of cheating and honest reports correctly identified during the report preprocessing phase with our trust-distrust propagation method.

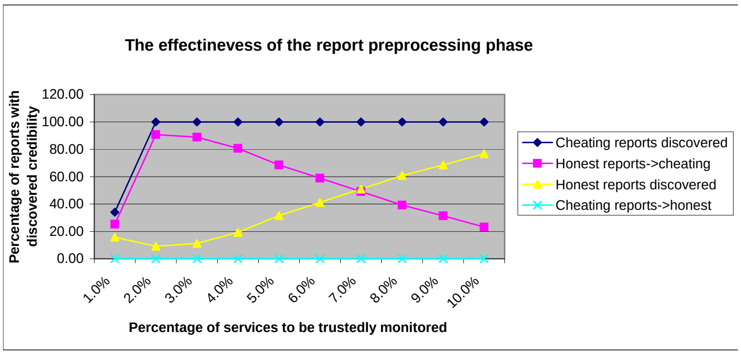

Fig. 3. $F_{special}$ vs. *Correctness of the report preprocessing phase.*

In this experiment, we assumed a very high number of cheaters (74.0% of the total users) consisting of badmouthing users, advertisers and five different groups of uncertain users. We also did various experiments to study the effects of σ_c's and σ_h's values and found that our algorithm performed very well with different settings of σ_c provided the quality of the honest user population was good, i.e., σ_h was low. Given the fact that the QoS conformance values in our simulation are in the interval $[-1.0, 1.0]$, we kept the standard deviations of cheating reports very high ($\sigma_c = 0.5$) and those of trusted and honest users at an acceptably low level ($\sigma_h = 0.01$). With the increase of $F_{special}$, we could correctly discover almost all cheating reports and an increasing percentage of honest reports. Accordingly, the quality of the results was significantly increased as well. The clustering phase was actually not performed with $F_{special} > 1.0\%$ because above this threshold, using only the trust-distrust propagation was enough to evaluate the credibility of all reports. Although a lot of honest reports were wrongly identified as cheating, which was due to our cautious approach in estimating report credibility, the quality of the results was always very good if $F_{special}$ was kept high enough (about 5.0%). The results were the worst with $F_{special}$ around 2.0% and 3.0%. In these cases, as the trust-propagation was not effective and did not identify enough honest reports, we had to (partly) use the quality value advertised by providers instead. When $F_{special}$ was very small (1.0%), the result was still acceptable (*R-Precision* = 0.8), since in this situation we could actually combine the trust-propagation and the report clustering phase together. i.e., there were enough reports with unknown credibility after the preprocessing of reports such that the clustering step had enough input data to process. As a whole, the result of our algorithm with the trust and reputation management scheme in place is much better than that of the optimistic and the naive cases, as we expected.

Next, we studied the effects of the fraction of cheaters to the quality of results. We increased the total percentage of all types of cheating users ($F_{cheating}$), which consists of badmouthing, advertising and uncertain users, from 4.0% to 94.0% in increment of 10% each step. More specifically, we raised the percentage of badmouthing and advertising users/reports, from 3.33% to 78.33% while generating five different groups of uncertain users with corresponding percentage of dishonest reports increased from 0.67% to 15.67%. This setting represents the realistic situation when there are various types of dishonest providers colluding with the generated cheating users to boost the reputation of certain services and badmouth other ones, which could be considered as the most important case where there are various types of users with changing behaviors. The results for this experiment are shown in Fig. 4. We kept the percentage of trusted reports at 0.5% and let $F_{special} = 5.0\%$, as an acceptable fraction collected from observations in the first experiment. The standard deviations of of cheating and honest reports were kept at $\sigma_c = 0.5$ and $\sigma_h = 0.01$ respectively.

With the reduction of honest users and the corresponding increase of $F_{cheating}$, the values of the *R-Precision* parameter were also reduced. However, the quality of the results in the realistic case was always much better than that of the optimistic case and the naive case. Even when the fraction of cheaters $F_{cheating}$ was very high (0.84), the *R-Precision* parameter value in the realistic case was still acceptable (higher than 0.8). On the other hand, the quality of results without taking into account trust and reputation management issues, i.e., the optimistic and the naive case, dropped dramatically in hostile settings. This phenomenon

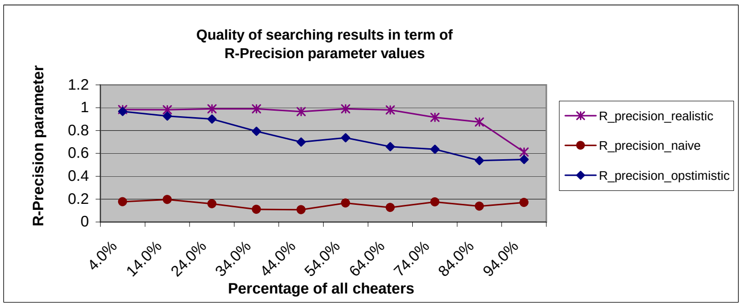

Fig. 4. $F_{cheating}$ vs. *R-Precision*

was due to the fact that in a society with a very high cheating rate, our trust and reputation evaluation mechanism could discover and filter out almost all incredible reports, as shown in Fig. 5.

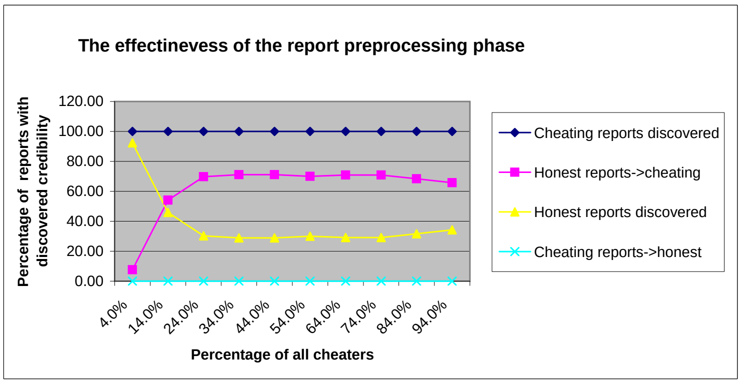

Fig. 5. $F_{cheating}$ vs. *Correctness of the report preprocessing phase.*

From these experiments we can draw a number of conclusions. Regarding efficiency and effectiveness, our selection and ranking approach exhibits the following properties: As the trust and reputation evaluation phase uses social network-based analysis (trust-distrust propagation) and data-mining (report clustering) methods, it requires high-computational cost. Fortunately, in our case, the computation involves mainly local processing in one registry and thus does not require much communication overheads. Additionally, it could be done *off-line* on a periodical basis and therefore will not affect much to the system performance. Another important observation is that almost no cheating report was wrongly evaluated as honest even in very hostile settings due to our cautious reputation evaluation mechanism. Therefore, in reality one can observe the results of the trust-distrust propagation step and incrementally adjust the parameters of the system, e.g., increase $F_{special}$, in order to collect enough honest reports for the performance prediction phase. The service selection and ranking can be performed fast and efficiently thanks to the pre-computation of the matching information between QoS of existing services with possible quality requirements of users. Similar to [17], we conclude that *the use of trusted third parties monitoring a relatively small fraction of services can greatly improve the detection of dishonest behavior even in extremely hostile environments*. However, this effectiveness mainly depends on the following properties of the set of service users and their corresponding reports:

1. *The overlaps among the set of users of each service*, i.e., whether the services monitored by trusted agents have many users who are also consumers of other services.
2. *The inconsistency in the behavior of users*, i.e., whether a user is honest while reporting on a service but behaves dishonestly on other cases.

These factors suggest that *we should deploy trusted agents to monitor the QoS of the most important and most widely-used services* in order to get a *"high impact"* effect when estimating behaviors of users. Currently as the user reports are distributed uniformly for services in our experiments, we have not yet taken into account this factor. However, we have employed another technique to fight back the inconsistency behaviors of users by *detecting cheaters first* and *evaluating honest users later* in the preprocessing phase. This helps us to collect all possible evidences against cheaters by making use of the distrust propagation among reports at the first place. The philosophy behind is that it would be better to filter out a certain number of honest users rather than accept some dishonest reports. However, lots of credible users will be accidentally detected as cheating because of the similarity between their reports with other ones produced by well-disguised liars in the system. Thus, in the honest detecting (trust-propagation) phase, we also give these users a second chance to prove their honesty provided they have produced lots of credible reports which could be certified by trusted reports and other honest users. Additionally, other techniques can also be utilized to make this method more robust. For example, we can *pre-select the important services to monitor* and *increase the number of them* as well as *keep the identities of those specially chosen services secret* and *change them periodically*. Thus, cheaters will not know on which services they should report honestly in order to become high-reputation recommenders and have to pay a very high cost to have great influences in the system. In reality, this also help us to reduce the cost of setting-up and maintaining trusted agents as we only need to deploy them to monitor changing sets of services at certain time periods. Moreover, we can extend our algorithm so that *neighboring registries are allowed to exchange with each other the evaluated credibility of users* to further find out other possibly well-disguised cheaters who frequently change their behaviors. Note that the adjacent registry peers in our system are assigned to manage services with similar functional characteristics and therefore they are likely to attract comparable sets of users in the system [8].

6 Conclusion

In this paper, we have introduced a new QoS-based web service selection and ranking algorithm with trust and reputation management support. We have shown that our selection and ranking solution yields very good results in most cases. As the proposed reputation management mechanism is robust against various cheating behaviors, the results are generally of good quality even in hostile situations in which many different types of cheaters make up a high percentage of the overall users and report values with remarkable variances. By combining a trust-distrust propagation approach with a data-mining method, we could filter out almost all cheaters and find out honest reports to be used in the quality prediction process with very high probability. However, there are a lots of open issues and improvements we can apply to the current model. First of all, the selection of services to be monitored by trusted agents is an interesting point not yet to be mentioned. Another open problem is how to accurately predict the performance of newly published services with only few QoS reports and how to motivate users to evaluate services' performance and submit their feedback to the service search engine. The selection of an optimal configuration for many design parameters of our proposed solutions is also an important question to be studied in further. Additionally, in each iteration of the trust-distrust propagation, the comparison should be done between an unmarked report with the average of current honest/cheating ones to make the credibility evaluation more accurate since the reports are generally different from each other. Also, after this propagation step, there maybe a number of users whose real behaviors are not revealed due to insufficient evidences while their marked reports include both cheating and honest ones. It is possible to take this factor into account while clustering and evaluating the credibility of various report groups. As part of future work, we will also use QoS properties as ranking criteria for service queries without explicit QoS requirements. Another plan is to develop a so-called meta-behavior model of users, which is more general to describe user behaviors with various possible probabilistic responses and to obtain analytical results of the proposed solution. Last but not least, we are going to deploy our algorithm in a decentralized setting to observe the effectiveness of our trust and reputation techniques where there are many registry peers exchanging among each other the information of users and services' quality data.

References

1. A. Jøsang, R. Ismail and C. Boyd: A Survey of Trust and Reputation Systems for Online Service Provision, *Decision Support Systems*, 2005 (to appear).
2. Z. Despotovic and K. Aberer: Possibilities for Managing Trust in P2P Networks, *EPFL Technical Report No. IC200484*, Switzerland, November, 2004.

3. M. Tian, A. Gramm, T. Naumowicz, H. Ritter, J. Schi: A Concept for QoS Integration in Web Services, *Proceeding of WISEW'03.*
4. S. Ran: A Model for Web Services Discovery with QoS, *ACM SIGecom Exchanges*, Vol. 4, Issue 1 Spring, pp. 1-10, 2003.
5. M. Ouzzani, A. Bouguettaya: Efficient Access to Web Services, *IEEE Internet Computing*, Mar./Apr., pp. 34-44, 2004.
6. C. Patel, K. Supekar, Y. Lee: A QoS Oriented Framework for Adaptive Management of Web Service-based Workflows, *Database and Expert Systems 2003 Conf.*
7. E. M. Maximilien and M. P. Singh: Reputation and Endorsement for Web Services, *SIGEcom Exch., 3(1):24-31, ACM Special Interest Group on e-Commerce*, 2002.
8. L.- H. Vu, M. Hauswirth and K. Aberer: Towards P2P-based Semantic Web Service Discovery with QoS Support, *Proceeding of Workshop on Business Processes and Services (BPS)*, Nancy, France, 2005 (to appear).
9. N. Srinivasan, M. Paolucci, K. Sycara: Adding OWL-S to UDDI, Implementation and Throughput, *Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition*, USA, 2004.
10. G. Dobson: Quality of Service in Service-Oriented Architectures, 2004, http://digs.sourceforge.net/papers/qos.html.
11. Z. Chen, C. Liang-Tien, B. Silverajan, L. Bu-Sung: UX - An Architecture Providing QoS-Aware and Federated Support for UDDI, *Proceedings of ICWS'03.*
12. A. S. Bilgin and M. P. Singh: A DAML-Based Repository for QoS-Aware Semantic Web Service Selection, *Proceedings of ICWS'04.*
13. S. Kalepu, S. Krishnaswamy and S. W. Loke: Reputation = f(User Ranking, Compliance, Verity), *Proceedings of ICWS'04.*
14. Y. Liu, A. Ngu, and L. Zheng: QoS Computation and Policing in Dynamic Web Service Selection, *Proceedings of WWW 2004 Conf.*
15. F. Emekci, O. D. Sahin, D. Agrawal, A. E. Abbadi: A Peer-to-Peer Framework for Web Service Discovery with Ranking, *Proceedings of ICWS'04.*
16. J. Day and R. Deters: Selecting the Best Web Service, *the 14th Annual IBM Centers for Advanced Studies Conf.*, 2004.
17. R. Guha and R. Kumar: Propagation of Trust and Distrust, *Proceedings of WWW 2004 Conf.*
18. M. Richardson, R. Agrawal, P. Domingos, Trust Management for the Semantic Web, *Proceedings of ISWC'03*, LNCS 2870, p.p. 351-368, 2003.
19. K. Aberer and Z. Despotovic: Managing Trust in a Peer-2-Peer Information System, *Proceedings of ACM CIKM'01.*
20. A. Whitby, A. Jøsang and J. Indulska: Filtering Out Unfair Ratings in Bayesian Reputation Systems, *Icfain Journal of Management Research*, Vol. IV, No. 2, p.p. 48-64, Feb. 2005.
21. F. Cornelli, E. Damiani, S. C. Vimercati, S. Paraboschi and P. Samarati: Choosing Reputable Servents in a P2P Network, *Proceeding of WWW 2002 Conf.*, USA.
22. K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt: P-Grid: a Self-Organizing Structured P2P System. *ACM SIGMOD Record*, 32(3), 2003.
23. M. Paolucci, T. Kawamura, T. Payne, and K. Sycara, Semantic Matching of Web Services Capabilities, *Proceedings of ISWC'02.*