Towards Probabilistic Estimating Quality of Online Services

Le-Hung Vu and Karl Aberer
Swiss Federal Institute of Technology Lausanne (EPFL),
School of Computer and Communication Sciences,
CH-1015 Lausanne, Switzerland
{lehung.vu|karl.aberer}@epfl.ch

Abstract

Accurate estimation of quality of online services is both an important and difficult problem, since a service has many interdependent quality attributes influenced by several contextual factors. It is even more challenging as quality ratings come from sources with unknown reliability, each source may rate a service on different quality aspects. Although several solutions have been proposed, there is little work addressing all these issues thoroughly. In this paper, we show that domain knowledge on service structure and related constraints, such as causal dependencies among quality attributes and contextual factors, while widely available, can be exploited to effectively address the above issues in a theoretically-sound framework. Theoretical analysis shows that computational cost of the approach is acceptable, and accurate evaluation of service quality requires a reasonable number of user feedback, provided services have a small number of quality attributes and contextual factors.

Keywords: service, quality, trust, probabilistic learning, probabilistic modeling, EM, MLE, Bayesian network

1 Introduction

While provisioning of services takes place electronically or traditionally depending on the domain, any services can be advertised, discovered and requested electronically via the Internet. Examples of traditional services published on advertisement sites such as Craigslist[1] are numerous, from car rental, travel planning to house relocation services.

Accurate estimation of service quality is of paramount importance in online environments. In business scenarios with competitive providers, quality rating is normally the decisive criteria influencing a user in selecting a service among several functionally equivalent ones. For instance, between two online travel planning services, a user would aim for the one with better route coverage, more user-friendly, and offer plans with lower price and transit time.

Ratings on quality of online services can be obtained in many ways. Automatic monitoring and measuring service quality is proposed in, for example, [1]. Such an approach is appropriate to measuring network-related quality attributes of a Web service, e.g., response-time or availability, but inapplicable to other domain-dependent quality properties of an *online service* in general, e.g., whether the service conforms to its advertised functions or whether it is easy to use the service. However, in reality users are even more interested in these domain-dependent quality features. O'Sullivan et al [2] give an extensive review of these quality attributes across different service domains.

Alternatively, feedback from previous users can be used as a relevant source to rate most quality attributes of a service. Feedback is available in various places, from extended registries [3] to professional forums and social networks. In this case, a fundamental issue is the *reliability* or *trustworthiness* of these feedback sources, as reports from unknown people can either be reliable or biased depending on the innate behaviors and motivation of the ones sharing the feedback. For example, competing service providers have strong incentives to advertise their services as having the highest possible quality and to badmouth quality of their competitors in order to increase their revenues [4, 5].

Accurate estimation of service quality ratings is even more challenging since these signals are *multi-dimensional*, *context-sensitive*, and may be *nondeterministic*. A service has several inter-dependent quality attributes whose values may be influenced by various contextual factors nondeterministically. Examples are dependencies between the membership type of a data hosting service, e.g., Amazon S3, and download/upload bandwidth available to a member given his network connection speed.

Several approaches propose using user feedback to evaluate quality ratings, e.g., [6, 7]. However, most solutions either estimate ratings on one service quality attribute, or ratings on a number of independent quality dimensions. There

[1] http://www.craigslist.org

are few work that address adequately all of the above issues: the *multi-dimensionality*, *context-sensitivity*, and *nondeterminism* of service quality signals, as well as the *reliability* of contributing feedback sources. Furthermore, very few work considers the case where a user may only give feedback on a few quality aspects of the service, resulting in a sparse set of quality ratings.

This paper presents an overall framework to use *service domain knowledge to model and estimate ratings on quality* of services *probabilistically*, taking into account all these above issues. The key idea is that knowledge on the structure of a service and related constraints, such as causal dependencies among quality attributes, contextual factors, are generally available in a service domain. This information is exploited for the QoS modeling and learning process in many aspects. First, we exploit the fact that for a service in a specific domain, their QoS parameters and dependencies to related contextual factors are well-known. With such an assumption, the modeling of QoS capabilities of a service results in a fixed-structure Bayesian network with small in-degree bound, thus subsequent probabilistic learning and inferences on it is scalable in terms of computational cost. Second, we use constraints among quality attributes and contextual factors, as well as the malicious behaviors of rating users and their rating values to define conditional probabilities of the model, and filter invalid ratings to improve performance of the quality rating estimation. Third, prior beliefs on rating sources' reliability and service provider's behavior are used to initialize the parameter learning of the Bayesian network-based model of the service to further improve the estimation accuracy. Our study shows that the proposed learning framework with appropriate use of commonly available knowledge in a service domain offers us many advantages:

- it gives theoretically-sound solutions to effectively address the issues of multi-dimensionality, context-sensitivity, and reliability of service quality ratings, while applicable to many service domains.
- it enables a user to subjectively model and evaluate various quality dimensions of interested services according to personalized preferences, prior beliefs and available information on trustworthiness of the feedback sources.
- the quality learning works well given a sparse rating set: we have obtained reasonably good results from previous experiments [8]: with a high fraction (more than 50%) of missing values from the reports of many biased sources, the estimation accuracy is still acceptable. This is mainly due to use of a QoS generative model: training the model using available quality reports helps to learn ratings on unobserved quality factors from observations on other quality attributes. Furthermore, correlation between opinions of honest and trusted users are seamlessly integrated in the learning step and help to identify reliable reports and isolate biased ones.

To our knowledge, this work is the first one that effectively exploits available knowledge in a service domain, model dependencies among service quality parameters, contextual factors, and reliability of raters for the benefits of the service quality estimation. Furthermore, while we focus on modeling and assessing service quality, the methodology is naturally applicable to other service's non-functional properties with probabilistic dependencies.

2 Framework overview

Fig. 1 gives an overview of our approach to probabilistic modeling and estimation quality of a service from ratings by many feedback sources. The modeling step builds a QoS generative model of a service to represent its quality capabilities. The learning (training) step trains the model with feedback on quality of the service from many sources to learn its unknown parameter. The third step estimates the service quality level under a certain context by probabilistic inference on the model learnt from two previous steps.

This framework can be integrated into a service reputation management framework [9] or a service search engine, namely Seekda[2] to help users selecting the best services. In this paper we will introduce candidate algorithms for the above steps to evaluate quality of an example service and analyze in details computational complexity of these algorithms to demonstrate the possibility of our approach. These algorithms, however, are subject to many optimizations and can be replaced by better ones, as discussed later on.

3 System model and notations

Denote $\mathcal{Q}$ the set of quality parameters of a service, where each $q \in \mathcal{Q}$ is assumed to have discrete values. While this assumption largely simplifies our problem and subsequent analysis, it is both realistic and advantageous. First, many quality attributes either have categorical values or are best represented with ranges of values due to their uncertain nature. Second, a rating on service quality is in fact the conformance between the quality values promised and delivered by the provider, as evaluated by the user. Such ratings are best modeled as discrete grading scales, as normally used in rating hotels or travel planning services. Similarly, let $\mathcal{E}$ be the set of contextual factors affecting values of quality parameters in $\mathcal{Q}$. In this paper we only consider the case where contextual factors have discrete/categorical values. The same methodology, however, applies in the case of continuous contextual factors and quality signals.

Motivating example: Consider a data hosting service similar to Amazon S3 or services by other data hosting providers. The following quality attributes are of our interests: its maximal number of concurrent downloads M, its download speed D and its upload speed U. Values of

[2] http://seekda.com/

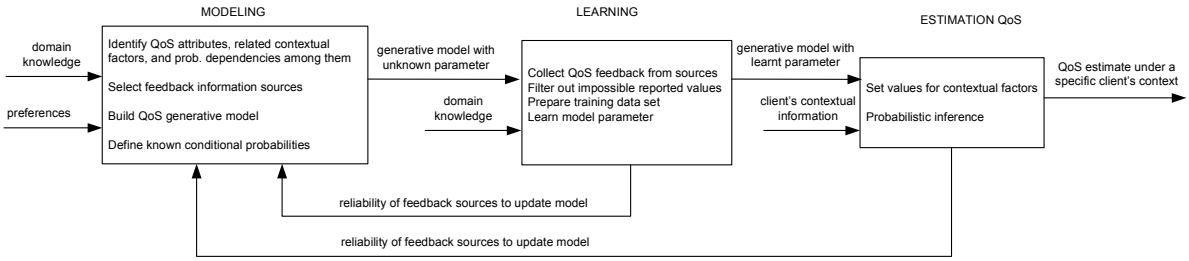

Figure 1: The framework overview.

these attributes are decided by the subscription price P and the Internet connection type I of the service consumer. In other words, P and I define the context, or the client-side setting in which the provider promises to offer its data hosting service with specific quality level of M, D, and U.

Fig. 2(a) shows a Bayesian network-based model of quality of the data hosting service with state spaces of all nodes and probabilistic dependencies among them. Such information is well-known in the data hosting domain and may even be declared by the provider in his service advertisement. Fig. 2(a), with conditional probabilities of each variable given state of parent nodes, is example of a *QoS generative model* of the data hosting service.

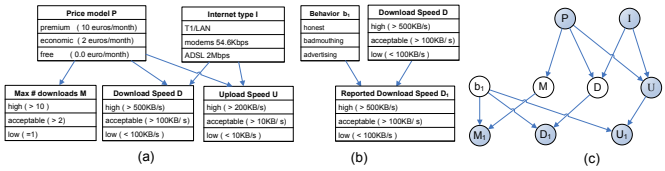

Figure 2: (a) Example quality attributes and related contextual factors of a data hosting service, state spaces of each node, and dependencies among them; (b) Dependencies among the trustworthiness (reporting behavior) of an untrusted feedback source and its corresponding reported values; (c) The QoS generative model of the data hosting service in simple form.

A user may obtain reports (ratings) on quality of a service from a list of sources $\mathcal{I}$, only some of which are reliable. Reliable (trusted) sources $\mathcal{T} \subseteq \mathcal{I}$ may include, for example, third-party/commercial QoS monitoring services. Untrusted sources $\mathcal{I} \backslash \mathcal{T}$ are usually previous service users. The trusted sources and those quality attributes to be monitored by them shall be private information of the learning user. Otherwise, an adversary may disguise as an honest reporter to manipulate the QoS prediction effectively. For example, the adversary can provide reliable information only on those quality attributes monitored by trusted sources, while reporting biased feedback on the others.

Let $\mathcal{U}$ be the set of variables in a QoS generative model of a service. A node $x \in \mathcal{U}$ represents either a quality attribute, a contextual factor, the trustworthiness of an information source, or the rating on a quality attribute. The domain value of x, or its state space, is denoted $\mathcal{D}_x$. A directed edge from one node to another denotes a probabilistic or causal dependency between two variables. For brevity, we denote π_x the set of parent nodes of x. If x represents a quality attribute in $\mathcal{Q}$, the set of contextual factors in $\mathcal{E}$ that x directly depends on is $\phi_x = \pi_x \cap \mathcal{E}$.

The realization of x to some value $\mathsf{v} \in \mathcal{D}_x$ is denoted as $x = \mathsf{v}$ (similarly we have $x \neq \mathsf{v}$). Whenever it is irrelevant to mention v, we use the brief notation x^*. Similarly, for any given set $\mathcal{S}$ of random variables, we denote $\mathcal{S}^*$ the joint event in which each variable $s \in \mathcal{S}$ is assigned certain value in $\mathcal{D}_s$. The conditional probability that x gets some value given states of its parents π_x, or a *conditional probability table (CPT)* entry of the node x, is simply $Pr(x^* \mid \pi_x^*)$.

We then define the following important concepts.

Definition 1. *The QoS generative model of a service* is a Bayesian network $\langle \mathcal{U}, \mathcal{D}, \theta \rangle$ with node set $\mathcal{U}$, edge set $\mathcal{D}$, and parameter θ. The tuple $\langle \mathcal{U}, \mathcal{D} \rangle$ is a directed acyclic graph where $\mathcal{D} = \{\langle p, x \rangle, x \in \mathcal{U}, p \in \pi_x\}$ defines probabilistic dependencies among nodes in $\mathcal{U}$. The parameter θ is the set of unknown *conditional probability table (CPT)* entries $Pr(x^* \mid \pi_x^*)$ of each node $x \in \mathcal{U}$.

Definition 2. *A QoS rating* by a source $j \in \mathcal{I}$ at time t on a service is defined as $\langle j, t, \mathcal{V}_j^* \rangle$, where $\mathcal{V}_j^*$ denotes a set of reported values of j on a subset $\mathcal{V}_j \subseteq \mathcal{Q} \cup \mathcal{E}$. Specifically, $\mathcal{V}_j^* = \{x = \mathsf{r}_{\mathsf{jt}}(x) \mid x \in \mathcal{V}_j\}$, where $\mathsf{r}_{\mathsf{jt}}(x) \in \mathcal{D}_x$ is the reported value of j on x at time t.

In general a source j may report only on a number of quality attributes or contextual factors $\mathcal{V}_j$. Thus values of several quality attributes and contextual factors may be missing in a QoS rating. To estimate the unknown parameter θ of a QoS generative model, related QoS ratings from many sources are combined to build a training set, which includes many *observations* (*samples*) on the model (Def. 3).

Definition 3. *An observation* or *a sample on a QoS generative model* is defined as $\mathbf{v}_\mu = \{x = \mathsf{x}^\mu, x \in \mathcal{O}\}$, where $\mathsf{x}^\mu \in \mathcal{D}_x$ is a rating value on a node $x \in \mathcal{O}$, and the set $\mathcal{O} \subseteq \mathcal{Q} \cup \mathcal{E}$ is a subset of quality attributes or contextual factors whose values during a measurement epoch μ can be obtained by combining ratings from some sources.

4 Service quality modeling

Given own knowledge on the reliability of feedback sources, a user may set up a QoS generative model of the following types. *The basic QoS generative model* (Fig. 3a) is applicable when the user collects ratings from own experience and only trusted sources. *The extended QoS generative model* (Fig. 3b) is used when the user collects feedback from many sources $\mathcal{I}$, only a subset $\mathcal{T}$ of which is reliable. A QoS generative model is built with the following steps:

1. Identify relevant quality attributes $\mathcal{Q}$ of the service, related contextual factors $\mathcal{E}$ and their domain values. Discretize values of a continuous quality attribute into ranges if necessary. Identify the causal/probabilistic relationships among $\mathcal{Q}$ and $\mathcal{E}$. This results in a basic QoS generative model as shown in Fig. 3(a).
2. For each untrusted feedback source $j \in \mathcal{I} \backslash \mathcal{T}$, add a node b_j to represent the trustworthiness of j. For each $b_j, j \in \mathcal{I}$ and for each $q_i \in \mathcal{Q}$, add a node v_{ij} with the same domain as q_i. Add a directed link from each b_j and from each q_i to v_{ij}. Thus, v_{ij} represents the rating of a source $j \in \mathcal{I}$ on a quality attribute $q_i \in \mathcal{Q}$. The following nodes are marked *observable*: all nodes v_{ij}, $\mathcal{E}$, and each q_s rated by a trusted source $s \in \mathcal{T}$. Remaining nodes are *hidden* nodes. This step results in an extended QoS generative model (Fig. 3b) with an unknown parameter θ.
3. Well-known constraints among values of nodes are exploited to define certain CPT entries of the model. Particularly, one may set $Pr(v_{ij} = \mathsf{v} \mid b_i = \mathsf{reliable}, q_j = \mathsf{v}) = 1$, and $Pr(v_{ij} = \mathsf{v} \mid b_i \neq \mathsf{reliable}, q_j = \mathsf{v}) = 0$ for any $q_i \in \mathcal{Q}, \mathsf{v} \in \mathcal{D}_{q_i}, j \in \mathcal{I} \backslash \mathcal{T}$. Other value constraints of quality and contextual attributes can be exploited to set CPT entries of related nodes. Formally, a constraint may be of the form $\bigwedge_{x \in \mathcal{X}} x^* \rightarrow y^*$ or $\bigwedge_{x \in \mathcal{X}} x^* \rightarrow (y \neq \mathsf{v})$, for some $\mathcal{X} \subseteq \mathcal{U}$, some $y \in \mathcal{U}$, and some $\mathsf{v} \in \mathcal{D}_y$. Define $\mathcal{KB}$ be the domain knowledge built from those constraints. For each $x \in \mathcal{U}$, the following rules apply: if $\mathcal{KB} \models \{\pi_x^* \rightarrow x^*\}$, we define $Pr(x^* \mid \pi_x^*) = 1$[3]. If $\mathcal{KB} \models \{\pi_x^* \rightarrow (x \neq \mathsf{v})\}$, set $Pr(x = \mathsf{v} \mid \pi_x^*) = 0$. Prolog programs can help to analyze such complex constraints to set values of related CPT entries.

The above steps result in a (personalized) QoS generative model of the service, with some pre-defined CPT entries. Normally, a generative model is set up once for each service type in a domain, with possible help of domain experts. We emphasize that *it is possible to include dependencies among quality attributes q_i, or among contextual factors e_l* in Fig. 3(a,b). These dependencies are not shown for the presentation clarity. However, our proposed algorithms in coming sections are generic enough to handle these cases. Also, we assume that values of contextual factors are verifiable and thus are observable variables. This is not a strong assumption since there is no direct incentive for feedback sources to manipulate such values, but only values of concerned quality parameters. Nevertheless, whenever contextual values are subject to manipulation, it is trivial to extend the above models and use the same approach with contextual factors marked as hidden nodes. It is also noteworthy that available domain constraints are very useful as they help to define many conditional probabilities of the model (step 3.) and thus reduce the size of the parameter set θ to be estimated in later steps.

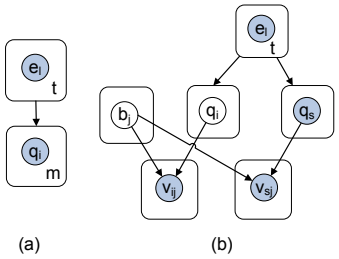

Figure 3: (a) The basic QoS generative model of a service; (b) The extended QoS generative model. A rounded square surrounding a node represents a group of similar variables, possibly with the number of variables in the group. Shaded nodes are *observable* (*visible*) variables whose values are known or obtained in some ratings, whereas blank nodes are *hidden* variables whose values are completely unknown.

Modeling example: Consider the basic QoS generative model for the previous data hosting service (Fig. 2a). Suppose that feedback on U is provided by a trusted source, and feedback on M, D, and U is obtained from an untrusted source with reporting behavior b_1. We use M_1, D_1, and U_1 to denote reported values by the untrusted source on M, D, and U, respectively. P, I, U, M_1, D_1, and U_1 are marked observable. Fig. 2(b) shows possible dependencies between the trustworthiness b_1 of a source and reported values U_1 on the download speed of the data hosting service. A feedback source observing a certain quality level may either report the same value, or deliberately give higher or lower rating value. E.g., a user may give a bad rating on service of its competitors irrespective to the quality it actually perceives. These reporting behaviors are denoted as *reliable*, *advertising*, and *badmouthing* respectively, i.e., the feedback source is reliable with some unknown probability. Fig. 2(c) shows the QoS generative model after the second modeling step.

We then exploit the following constraints to define some CPT entries of the model. Given the above reporting behavior model, for any $\mathsf{u} \in \mathcal{D}_U$ we have $Pr(U_1 = \mathsf{u} \mid b_1 = \mathsf{reliable}, U = \mathsf{u}) = 1.0$, $Pr(U_1 = \mathsf{high} \mid b_1 = \mathsf{advertising}, U = \mathsf{u}) = 1.0$, and $Pr(U_1 = \mathsf{low} \mid b_1 = \mathsf{badmouthing}, U = \mathsf{u}) = 1.0$. Also, if the maximal number of downloads is deterministically defined by the price, we set $Pr(M = \mathsf{high} \mid P = \mathsf{premium}, I^*) = 1$,

[3]We use the notation $\models$ to represent the deduction (provability) of a fact from a knowledge-base in first-order logic

$Pr(M = \mathsf{acceptable} \mid P = \mathsf{economic}, I^*) = 1$, and $Pr(M = \mathsf{low} \mid P = \mathsf{free}, I^*) = 1$. Thus after the third modeling step we have an extended QoS generative model for the data hosting service with certain known CPT entries. The unknown parameter θ of the model consists of the remaining unknown CPT entries of b_1, M, and D.

Many existing works use certain instances of our above generative models for their QoS prediction, though the modeling is not stated explicitly. For example, those approaches considering a number of independent quality attributes [10] consider a generative model of independent nodes q_i. Those work assuming the total reliability of feedback sources [6, 11] simply use the basic QoS generative model and propose a training algorithm based on some heuristics to estimate the model parameter θ. As a result, this work provides a framework to apply various probabilistic machine learning techniques to evaluate service quality which generalizes existing approaches.

5 Training and estimating service quality

Before using the QoS generative model for the estimation of service quality, we need to train the model to learn its unknown parameter θ. Specifically, collected QoS ratings from many sources are used as training data to update and estimate remaining unknown CPT entries of the model.

Training data preparation: A training data set includes many samples on the QoS generative model, each sample is obtained by combining related QoS ratings from different sources during a measurement epoch (Def. 3).

As in the modeling, the knowledge base $\mathcal{KB}$ built from domain constraints can also be used to filter out impossible observations in the report data set. For example, in the data hosting service domain, the maximum concurrent number of downloads for a subscribed user must be greater than 1, meaning that $Pr(M = \mathsf{low} \mid P \neq \mathsf{free}) = 0$. Thus any report by a source j of the form $\{M = \mathsf{low}, P = \mathsf{economic}\}$ or $\{M = \mathsf{low}, P = \mathsf{premium}\}$ is impossible and should be filtered out.

Consider a QoS rating $\langle j, t, \{x = \mathsf{r}_{\mathsf{jt}}(x) \mid x \in \mathcal{V}_j\}\rangle$ submitted by a source j (see Def. 2). This report is removed if it violates certain domain constraints in $\mathcal{KB}$. Such a violation is formally equivalent to:

$$\exists x : \mathcal{KB}, \bigcap_{y \in V_j, y \neq x} \{y = \mathsf{r}_{\mathsf{jt}}(y)\} \models \{x \neq \mathsf{r}_{\mathsf{jt}}(x)\} \qquad (1)$$

Violations as (1) can be detected by Prolog programs to eliminate invalid ratings of a source j. As j is likely to be unreliable, its related nodes in Fig. 3b. ($b_j, v_{ij} : q_i \in \mathcal{Q}$) are also eliminated. This preprocessing reduces the learning cost in many ways: it reduces the size of the training data set, while preserving most relevant information for the learning phase. Also, many nodes related to unreliable sources are eliminated, thus simplifying the model and reducing the number of parameters to be estimated.

Ratings on quality of the target service are transformed into observations on the QoS generative model as in Algorithm 1. The function $selectRatingsInEpoch(\mu, \mathcal{R})$ returns those QoS ratings in the set $\mathcal{R}$ whose timestamps t are within a measurement epoch μ. The function $findReportedNode(j, x)$ returns the node v that represents the reported value by j on the node x.

Algorithm 1 $PrepareTrainingData(reportData\ \ \mathcal{R})$: $trainingData\ \mathfrak{T}$

```
for μ = 0 to NumMeasureEpochs do
    v_μ = ∅; R_μ = selectRatingsInEpoch(μ, R);
    for each report ⟨j, t, {x = r_jt(x) | x ∈ V_j}⟩ in R_μ do
        if j ∈ T { reports from a trusted source } then
            v_μ = v_μ ∪ {x = r_jt(x)};
        else
            for each x ∈ V_j do
                v = findReportedNode(j, x);
                v_μ = v_μ ∪ {v = r_jt(x)};
            end for
        end if
    end for
    Add a sample v_μ to 𝔗;
end for
```

Learning by Maximum Likelihood Estimation (MLE): The unknown parameter of a QoS generative model can be estimated with traditional MLE techniques. This applies when all nodes of the model are observable, e.g., the case of a basic QoS generative model, or if training samples do not contain missing values. Consider a simpler version of the example data hosting service with two QoS attributes D and U as shown in Fig. 4. Assume that feedback on D and U is collected from an untrusted source with behavior b_1, and feedback on U is also provided by a trusted source.

As explained before in section 4, for any value d of D, $Pr(D_1 = \mathsf{d} \mid b_1 = \mathsf{reliable}, D = \mathsf{d}) = 1, Pr(D_1 = \mathsf{high} \mid b_1 = \mathsf{advertising}, D = \mathsf{d}) = 1, Pr(D_1 = \mathsf{low} \mid b_1 = \mathsf{badmouthing}, D = \mathsf{d}) = 1$. Similarly, for any $\mathsf{u} \in \mathcal{D}_U$: $Pr(U_1 = \mathsf{u} \mid b_1 = \mathsf{reliable}, U = \mathsf{u}) = 1, Pr(U_1 = \mathsf{high} \mid b_1 = \mathsf{advertising}, U = \mathsf{u}) = 1, Pr(U_1 = \mathsf{low} \mid b_1 = \mathsf{badmouthing}, U = \mathsf{u}) = 1$. Unknown CPT entries of the model in Fig. 4 are: $Pr(b_1 = \mathsf{reliable}) = h, Pr(b_1 = \mathsf{advertising}) = a, Pr(D = \mathsf{high}) = x, Pr(D = \mathsf{low}) = m$, and $Pr(U = \mathsf{high}) = p, Pr(U = \mathsf{low}) = q$. The unknown parameter of the model is thus $\theta = \{h, a, x, m, p, q\}$.

Suppose that we have a training data set $\mathfrak{T} = \{\mathbf{v}_\mu, 1 \leqslant \mu \leqslant N\}$ from feedback on D, U built according to Algorithm 1. Each observation $\mathbf{v}_\mu$ is the combination of ratings from the trusted source (on U) and the untrusted source (on D and U), so $\mathbf{v}_\mu = \{U = \mathsf{u}^\mu, U_1 = \mathsf{u}_1^\mu, D_1 = \mathsf{d}_1^\mu\}$. The probability of getting an observation $\mathbf{v}_\mu$ is:

$$\begin{aligned} Pr(\mathbf{v}_\mu \mid \theta) &= Pr(u^\mu, u_1^\mu, d_1^\mu \mid \theta) \\ &= \sum_{b_1, D} Pr(b_1) Pr(D) Pr(u^\mu) Pr(d_1^\mu \mid b_1, D) Pr(u_1^\mu \mid b_1, U) \end{aligned}$$

We select θ maximizing the log likelihood $LL(\theta)$ of

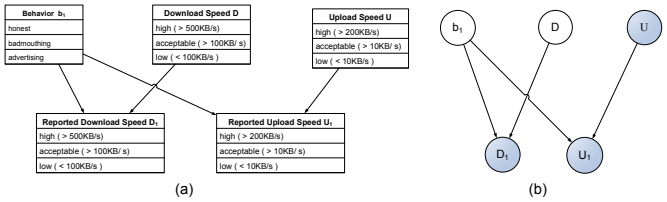

Figure 4: The generative QoS model of a data hosting service with two QoS parameters D and U, monitored by two sources, one of which is untrusted with behavior b_1.

obtaining the training set $\mathfrak{T}$, i.e., $\theta = \mathsf{argmax}_\theta LL(\theta) = \sum_\mu Pr(\mathbf{v}_\mu \mid \theta)$. Basic transformations give us[4]:

$$LL(\theta) = \sum_\mu \log Pr(u^\mu) + \sum_\mu \log\{1_{\{u_1=u_1^\mu\}} Pr(d_1^\mu)h + 1_{\{u_1^\mu=d_1^\mu=\mathsf{high}\}} a + 1_{\{u_1^\mu=d_1^\mu=\mathsf{low}\}}(1-a-h)\} \quad (2)$$

One may readily estimate θ from reported values $\mathsf{u}^\mu, \mathsf{u}_1^\mu, \mathsf{d}_1^\mu$ from (2). E.g., we have $p = \frac{\sum_\mu 1_{\{u^\mu=\mathsf{high}\}}}{\sum_\mu 1}$ and $q = \frac{\sum_\mu 1_{\{u^\mu=\mathsf{low}\}}}{\sum_\mu 1}$, similar to standard majority voting techniques for ratings on U. The reliability of the feedback source (h) can be estimated numerically via standard optimization techniques. This results in an estimate of h based on similarity between ratings u^μ of the feedback source and ratings of the trusted one u_1^μ, similar to intuition.

Learning by EM algorithm: For real services with many quality attributes and contextual variables, the resulting QoS generative models are more complex. Additionally, the training data set is likely to contain many missing values since each source may report on only a number of quality attributes. Direction maximization of the log likelihood function $LL(\theta)$ as above becomes non-trivial and does not give a closed-form solution of the estimated parameter θ.

Many approaches for estimating the parameter θ under such situations are applicable [12]. In this paper, we propose the Expectation-Maximization (EM) algorithm [13] to estimate θ for its many advantages. First, it works well on any QoS generative model and is especially useful if the log likelihood function is too complex to be directly optimized. Furthermore, it deals with incomplete data and converges rapidly. Its main disadvantages are the possibility to reach to a sub-optimal estimate (a local maximum of the log likelihood) with a bad initialization. However, we believe that it is possible to have an acceptably good initialization of θ to compensate these disadvantages, given personalized beliefs of the learning user on the behaviors of service providers and raters. The EM algorithm is given in Algorithm 2.

Line 1 of Algorithm 2 initializes unknown CPT entries of the generative model. Each conditional probability $Pr(x^* \mid \pi_x^*)$ can be initialized in many ways depending on available information: from prior beliefs of the user, as in the service advertisement by the provider, or as completely random. For the data hosting service, the learning user may assign $Pr(b_j)$ with his or her subjective belief on the trustworthiness of the source j. CPT entries of M, D, U in the data hosting service may be initialized as in the service advertisement. Lines $3-6$ implement the Expectation step of the EM algorithm, where the expected counts E_{xpx} and E_{px} of two joint events (x^*, π_x^*) and π_x^* are computed, given each observation $\mathbf{v}_\mu$ in collected reports and current parameter θ. Any exact or approximate probabilistic inference algorithm can be used to compute the posterior probabilities $Pr(\pi_x^* \mid \mathbf{v}_\mu, \theta)$ and $Pr(x^*, \pi_x^* \mid \mathbf{v}_\mu, \theta)$. The Junction Tree Algorithm (JTA) [14], is a good candidate as it produces exact results, works for all QoS generative models, and is still computationally scalable as shown in our later analysis.

The Maximization step of the EM algorithm is implemented in lines $7-9$ of Algorithm 2, therein we update the model parameters $Pr(x^* \mid \pi_x^*)$ such that they maximize the training data's likelihood, assuming that the expected counts computed in lines $3-6$ are correct. The two Expectation and Maximization steps are iterated till the convergence of the log likelihood $LL(\theta)$ of the training data set $\mathfrak{T}$, which gives us an approximation of unknown CPT entries $Pr(x^* \mid \pi_x^*)$ of the QoS generative model.

The set of visible variables in the QoS generative model may be changed after a user runs Algorithm 2, uses the service, and updates statistics of some quality attributes q_i with his own experience. Thus the learning of the model parameters is a reinforcement process with increasing accuracy over time, given availability of more training data to estimate a fewer number of parameters.

We do not consider possible optimization techniques, e.g., adjust the weight of each sample $\mathbf{v}_\mu$ according its recency, to accelerate the learning convergence. The use of other parameter learning algorithms, e.g., a Bayesian learning method, is also possible. These issues are subject to future work and thus beyond the scope of this paper.

Service quality estimation: Our goal is to compute the joint probability $Pr(\mathcal{D}^* \mid \Phi^*)$ that the service offers a set $\mathcal{D} \subseteq \mathcal{Q}$ of quality attributes at a desired level $\mathcal{D}^* = \{q^*, q \in$

Algorithm 2 *LearnParameter*(*model* $\mathcal{M}$, *trainingData* $\mathfrak{T} = \{\mathbf{v}_\mu, 1 \leqslant \mu \leqslant N\}$)

1: Init unknown $Pr(x^* \mid \pi_x^*)$ for each node x of $\mathcal{M}$; /*Initialize θ*/
2: **repeat**
3: **for** each observation $\mathbf{v}_\mu$ in $\mathfrak{T}$ **do**
4: Compute $Pr(x^*, \pi_x^* \mid \mathbf{v}_\mu, \theta)$, $Pr(\pi_x^* \mid \mathbf{v}_\mu, \theta)$ for each node x;
5: **end for**
6: $E_{xpx} = \sum_\mu Pr(x^*, \pi_x^* \mid \mathbf{v}_\mu, \theta)$; $E_{px} = \sum_\mu Pr(\pi_x^* \mid \mathbf{v}_\mu, \theta)$;
7: **for** each node x **do**
8: $Pr(x^* \mid \pi_x^*) = E_{xpx}/E_{px}$; /* Update θ */
9: **end for**
10: Recompute $LL(\theta) = \sum_\mu logPr(\mathbf{v}_\mu \mid \theta)$ with new θ;
11: **until** convergence in $LL(\theta)$ or after a maximal number of iterations;

[4]The function 1_A evaluates to 1 if A is true and 0 otherwise

$\mathcal{D}\}$ and under a client context $\Phi^* = \{e^* \mid e \in \Phi \subseteq \mathcal{E}\}$. This probability implies whether a service performs better than another in terms of the quality features $\mathcal{D}^*$ under environmental setting Φ^*. Thus, the result can be used for ranking and selection of appropriate services among functionally equivalent ones given their expected quality levels. The trustworthiness of an untrusted feedback source j, i.e., $Pr(b_j)$, is also of our interests, as this helps to select more reliable sources for future estimation, especially it is costly to obtain information from such feedback sources.

Given a generative QoS model with a known parameter θ, the computation of the above probabilities using probabilistic inference algorithms is straightforward [14] as follows. Values of contextual variables in Φ are defined according to setting of the user, then a probabilistic inference algorithm, such as JTA [14] is run on the model to compute the probability $Pr(\mathcal{D}^* \mid \Phi^*)$. Regarding the trustworthiness of feedback sources, all probabilities $Pr(b_j)$ are already computed during the parameter learning step.

For example, suppose that we want to estimate the probability that the service with the QoS generative model in Fig. 2(c) provides data hosting service with download speed level $D = \mathsf{high}$. Also assume that the user wants a free service. The probability $Pr(D = \mathsf{high} \mid P = \mathsf{free})$ can be done automatically with available probabilistic inference algorithm, given a QoS generative model with known CPT entries. Due to space limitation, we refer readers to [14] for a comprehensive tutorial on possible inference algorithms. The most important aspect is if we know conditional probability entries of a QoS generative model, the whole above inference procedure can be done fully automatically with acceptable computational cost (Section 6).

6 Analysis of the approach

Detailed experiments to evaluate accuracy of the estimation of quality rating on a service are much dependent on the services. Empirical experimental analysis of our approach with different services are subject to future work. However, experimental results on the example data hosting services are available [8]. Even with a high fraction (more than a half) of missing values from the reports of biased sources, the accuracy of our QoS estimates is still acceptable.

At the moment, we have obtained some theoretical results showing possibilities of our approach, whose applications are widely general to any service domain. These results show that domain knowledge can be exploited effectively to help the probabilistic modeling and estimation of service ratings with reasonable cost and acceptable accuracy. However, we emphasize that the algorithms presented in this paper are merely suggestive, despite their general suitability to many scenarios. Therefore, further optimizations and better algorithms can be used to increase the efficiency and accuracy of the QoS estimations.

We will estimate the most significant computational cost in our framework, which is the parameter learning and probabilistic inferences on the QoS generative model. Consider a worst case scenario where each of m quality attributes has t contextual factors as parent nodes. There are n feedback sources being used, none of which are trusted. Suppose that every node has a k-ary state space. In a specific domain and for a certain service, t, m, and k are fixed values, known to the service user, and typically much smaller than n. The Bayesian network of such a generative model has a total of $t + m + n + mn = O(n)$ nodes. The number of unknown CPT entries $Pr(x^* \mid \pi_x^*)$ in the parameter θ is at most $n_\theta = (k-1)t+(k-1)k^t m+(k-1)n+(k-1)k^2 mn = (k-1)(k^2 mn + n + k^t m + t) = O(n)$. The term $k-1$ is due to the normalization constraints as each variable has a k-ary state space.

Assume that a source j sends N reports on some quality attributes. In fact, N approximates the number of measurement epochs, or the number of samples obtained from Algorithm 1. The functions $selectRatingsInEpoch$ and $findReportedNode$ can be implemented with computational cost $O(1)$, e.g., with hash-based storage techniques. Since the cost of three loops in lines $1, 3, 7$ of Algorithm 1 are respectively $O(N), O(n)$, and $O(t+m)$, the computational cost of Algorithm 1 is $O(Nn(t+m)) = O(Nn)$. Even better, this step is usually done off-line. The computation that needs to be done on-demand is the estimate of $Pr(\mathcal{D}^* \mid \Phi^*)$, which has a cost of $O(n)$ (Proposition 1).

Proposition 1. *(see Appendix for a proof) The computational cost of one probabilistic inference on the worst-case QoS generative model using the Junction Tree Algorithm [14] is $O(n)$.*

From Proposition 2, the cost of one EM iteration is $O(Nn^2)$. Given the fact that in practice the EM algorithm converges fast, and we consider a limited number of contributing sources, Algorithm 2 is scalable in terms of computational cost with respects to the training data size N.

Proposition 2. *(see Appendix for a proof) The computational cost of one EM iteration of Algorithm 2 is $O(Nn^2)$.*

Learning errors and sample complexity: the sample complexity to learn the parameter of a fixed-structure Bayesian network, or the required number of samples N to learn θ with optimal error, is well-known in the computational learning literature [12, 15, 16]. In fact, the required number of samples in our learning framework is exponential in terms of the in-degree bound of nodes and grows less than linearly with the network size [15, 16]. For the QoS generative model in our worst-case scenario with a bound of node's in-degree of $O(t+m)$ and network size of $O(n)$, the sample complexity is $N = O(n2^{t+m})$. This complexity implies that our approach is feasible for estimating QoS of

services with a moderate number $t + m$ of quality attributes and contextual factors.

7 Related work

Many work propose to measure service quality using dedicated monitoring services or QoS brokers, namely [1, 17–19]. Quality ratings from these sources are reliable yet expensive to obtain and maybe inscalable in terms of costs. Some work [1] is only applicable to measuring network-related performance metrics of Web services, e.g., response-time, availability, but not for online services in general.

There are several approaches to using feedback from users to estimate quality and reputation of a service, namely [7, 10, 20]. [10] only suggests to identify every requester to avoid report flooding. [7] proposes another approach for evaluating and selection of Web services based on the difference among their QoS capability vectors. More recently, [6] proposes a collaborative filtering to estimate the quality of a new service using only reports of users whose experience is similar to him. [20] proposes the selection of services that are popularly used by many service users through the analysis of a network of services. These approaches usually have one or some of the following limitations: they either ignore the estimation of reliability of ratings, consider only a single quality attribute, or a number of independent quality properties of a service. Second, they are mostly based on ad hoc heuristics that are not theoretically sound. The notions of context are usually not included. Our proposed framework takes into account all these important issues and generalizes many above approaches in a theoretically sound way.

Approaches most related to our work are in trust management literature, which we do not survey due to space limitation and refer the readers to [4, 5] for an comprehensive review. The work in this paper is based on our previous work on Bayesian modeling and learning of peer's quality [8]. In [21] the authors propose modeling e-market services as a Bayesian network and use a Bayesian learning approach to estimate distribution of the model parameters. [22] uses a simple Bayesian network to learn the trustworthiness of a party by the EM algorithm. These approaches do not exploit the dependencies among quality attributes and contextual factors, as well as the presence of trusted parties to reduce the cost of probabilistic learning and inference on the models. Also, the problem of learning on a QoS generative model given a set of observations with missing data values have not been studied. As a side-effect, our framework, though independently developed, appears to subsume these specific approaches.

8 DISCUSSION & FUTURE TRENDS

The problem of estimation service quality in SOA environments has many challenging requirements, of which the most important are the multi-dimensionality and context-sensitivity of the quality signal of a service. Fortunately, domain knowledge on the structure of a service and related characteristics, such as causal dependencies among quality attributes, contextual factors, are generally available. This information should be exploited effectively for the benefits of the QoS modeling and learning process.

The framework presented in this paper is developed on the light of the above. Our goal is to provide a starting point to the application of several probabilistic machine learning techniques in estimating the quality of Web services. The proposed framework is independent of the service domain and thus is generally applicable to a variety of application scenarios. We have also obtained reasonably good results from previous experiments [8]: even with a high fraction of missing values from the reports of biased sources, the accuracy of our QoS estimates is still acceptable. However, it is noteworthy that the algorithms presented in this paper are merely suggestive, despite their general suitability to many scenarios. Therefore, further optimizations and better algorithms can be used to increase the efficiency and accuracy of the QoS estimations. We provide hereafter a list of possible improvements.

In reasonably stable environment with mostly reliable reporting sources, using even simpler QoS generative models are sufficient. For example, instead of using a separate variables b_j for each feedback source j, one can use a common variable b to model overall behavior of these sources. The number of unknown parameters in such simple generative models is much smaller, thus computational cost of probabilistic learning and inferences can be reduced significantly. The training data set becomes less sparse and as a result, the convergence speed and the accuracy of the parameter learning can be improved.

Several other parameter learning methods, e.g., a Bayesian learning approach, can be used in our framework in place of the EM algorithm. Many domain-dependent optimizations, e.g., adding weights for each sample based on its recency, are also possible. The readers are referred to [12] for a comprehensive review on existing techniques and their comparative advantages. Similarly, instead of using the standard JTA algorithms [14], approximate yet more efficient probabilistic inference procedures may be used, especially if we know the shape of the QoS generative model. Incremental versions of the training data preprocessing, the data preparation, and the EM algorithms are also available.

In practice, our approach can be implemented with little effort, given the availability of several Bayesian network tools and libraries. Interested readers are referred to [23] for a comparative review. Based on our own experience, for research experiments and analysis purposes, the BNT MATLAB toolbox [24] is best suited. The Netica library [25] is an easy to use and a very powerful tool for other implemen-

tations that need to integrate with existing working systems.

We do not address in this paper the mechanism to motivate (rational) users to provide truthful feedback on their consumed services. This is an important problem since any QoS estimation method relies on sufficient samples for a reasonable accurate learning. Side-payment mechanisms via scoring rules [26] or based on trusted reference reports [27] are promising approaches to this problem. Extension of these mechanisms to the case of services with multi-dimensional quality signals and with bounded-rational users are both challenging and interesting research questions.

9 CONCLUSION

This paper presents an overall framework for the probabilistic modeling and estimation of quality ratings of online services. We present methods to exploit available knowledge in a service domain to build a probabilistic graphical model that generates ratings on quality of a service. We have proposed and analyzed some candidate algorithms for the model parameter learning and service quality estimations. Our QoS modeling and learning steps take into consideration feedback information provided by both trusted and untrusted reporting users on different QoS attributes of the evaluated service. As a result, the learning framework enables the effective elimination of possibly biased information in favor of or against certain service providers and thus gives an accurate picture of service quality to support quality-based selection and ranking of services.

10 APPENDIX

Proof of Proposition 1. The worst case scenario of the QoS generative model is shown in Fig. 5(a).

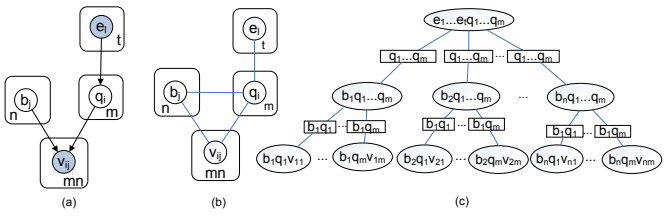

Figure 5: (a) The extended QoS generative model in worst-case scenario; (b) The extended QoS generative model after the moralization and triangulation step; (c) The junction tree of the extended QoS generative model.

The result of the moralization and triangulation steps of the Bayesian network in Fig. 5(a) is shown in Fig. 5(b). The corresponding junction tree of this Bayesian network is given in Figure 5(c). This junction tree has $nm+n$ edges and the maximal clique size with $t+m$ nodes. Thus the computational complexity of each inference using the JTA algorithm is $O(2(nm+n)2^{t+m}) = O(nm2^{t+m})$, where the factor $2(nm+n)$ corresponds to the number of messages passed during the inferences, and the complexity 2^{t+m} is the computation cost for marginalization of variables in the maximal clique $e_1 \dots e_t q_1 \dots q_m$ of the constructed junction tree. Therefore, one probability inference of the JTA algorithm cost $O(nm2^{t+m}) = O(n)$ for fixed t, m.

Proof of Proposition 2. From Proposition 1, the cost of computing a probability $Pr(x^* \mid C^*)$, where C^* denotes the setting of some variables in the generative model to certain states, is $O(n)$. This cost is bigger than the cost to compute either $Pr(x^*, \pi_x^* \mid \mathbf{v}_\mu, \theta)$ or $Pr(\pi_x^* \mid \mathbf{v}_\mu, \theta)$ in line 4 of Algorithm 2, as the former requires an additional sum over all unwanted variables.

Using the above result, one can verify that the computation cost for lines $3-6$ of Algorithm 2 is $O(Nn^2k) = O(Nn^2)$, with N is the number of observation cases. The computation cost for the loop in lines $7-9$ is $O(n)$, since there are $n_\theta = O(n)$ CPT entries $Pr(x^* \mid \pi_x^*)$ in the model.

The computation of the log likelihood $LL(\theta) = \sum_\mu logPr(\mathbf{v}_\mu \mid \theta)$ (line 10 of Algorithm 2) involves the summation over all hidden variables $\mathbf{h}_\mu$ (variables whose values not reported) for each observation $\mathbf{v}_\mu$. Since the nodes $b_j, 1 \leqslant j \leqslant n$ are independent, each sum $Pr(\mathbf{v}_\mu \mid \theta)$ can be implemented with the cost $O(k^{t+m+1}n) = O(n)$ as follows:

$$
\begin{aligned}
Pr(\mathbf{v}_\mu \mid \theta) &= \sum_{\mathbf{h}_\mu} Pr(\mathbf{h}_\mu, \mathbf{v}_\mu \mid \theta) \\
&= \sum_{\mathbf{h}_\mu \setminus \{b_1,\dots,b_n\}} Pr(q_i^\mu \mid \pi_{q_i^\mu}, \theta) Pr(e_l^\mu \mid \pi_{e_l^\mu}, \theta) \\
&\times \prod_{j=1}^{n} \sum_{b_j} Pr(b_j \mid \theta) Pr(\mathbf{v}_{ij}^\mu \mid b_j, q_i^\mu, \theta)
\end{aligned}
$$

The terms $Pr(q_i^\mu \mid \pi_{q_i^\mu}, \theta)$, $Pr(e_l^\mu \mid \pi_{e_l^\mu}$, $Pr(b_j \mid \theta)$, and $Pr(\mathbf{v}_{ij}^\mu \mid b_j, q_i^\mu, \theta)$ are known CPT entries of the current parameters θ already computed after the Maximization step (line $7-9$ of Algorithm 2).

Consequently, the total computation cost of each EM iteration in Algorithm 2 is $O(Nn^2) + O(n) + O(n) = O(Nn^2)$ for n contributing sources and N training samples.

References

[1] L. Fei, Y. Fangchun, S. Kai, and S. Sen, "A policy-driven distributed framework for monitoring quality of web services," in *ICWS*, pp. 708–715, 2008.

[2] J. O'Sullivan, D. Edmond, and A. H. M. ter Hofstede, "Formal description of non-functional service properties," tech. rep., Business Process Management Group, Centre for Information Technology Innovation, Queensland University of Technology, Australia, February 2005.

[3] S. C. Wong, V. Tan, W. Fang, S. Miles, and L. Moreau, "Grimoires: Grid registry with metadata oriented in-

terface: Robustness, efficiency, security," *IEEE Distributed Systems Online*, vol. 6, p. 2005, 1981.

[4] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decis. Support Syst.*, vol. 43, no. 2, pp. 618–644, 2007.

[5] Y. Wang and K.-J. Lin, "Reputation-oriented trustworthy computing in e-commerce environments," *IEEE Internet Computing*, vol. 12, no. 4, pp. 55–59, 2008.

[6] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized qos prediction forweb services via collaborative filtering," *icws*, vol. 0, pp. 439–446, 2007.

[7] Z. Luo, K. Qian, D. Cai, and J. S. Li, "QoS driven web services assessment and selection," *Int. J. Services Operations and Informatics*, vol. 1, no. 1–2, 2006.

[8] L.-H. Vu and K. Aberer, "A probabilistic framework for decentralized management of trust and quality," in *CIA*, pp. 328–342, 2007.

[9] D. Bianculli, W. Binder, L. Drago, and C. Ghezzi, "Transparent reputation management for composite web services," in *ICWS'08*, pp. 621–628, IEEE Computer Society Press, September 2008.

[10] Y. Liu, A. Ngu, and L. Zheng, "QoS computation and policing in dynamic web service selection," in *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, (New York, NY, USA), pp. 66–73, ACM Press, 2004.

[11] E. M. Maximilien and M. P. Singh, "Agent-based trust model involving multiple qualities," in *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, (New York, NY, USA), pp. 519–526, ACM Press, 2005.

[12] W. Buntine, "A guide to the literature on learning probabilistic networks from data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 2, pp. 195–210, 1996.

[13] R. Neal and G. Hinton, "A view of the EM algorithm that justifies incremental, sparse, and other variants," in *Learning in Graphical Models* (M. I. Jordan, ed.), Kluwer, 1998.

[14] C. Huang and A. Darwiche, "Inference in belief networks: A procedural guide," *International Journal of Approximate Reasoning*, vol. 15, no. 3, pp. 225–263, 1996.

[15] S. Dasgupta, "The sample complexity of learning fixed-structure bayesian networks," *Mach. Learn.*, vol. 29, no. 2-3, pp. 165–180, 1997.

[16] P. Wocjan, D. Janzing, and T. Beth, "Required sample size for learning sparse Bayesian networks with many variables," *Arxiv preprint cs.LG/0204052*, 2002.

[17] S. Ran, "A model for Web services discovery with QoS," *SIGecom Exch.*, vol. 4, no. 1, pp. 1–10, 2003.

[18] M. Ouzzani and A. Bouguettaya, "Efficient access to Web services," *IEEE Internet Computing*, pp. 34–44, March/April 2004.

[19] C. Patel, K. Supekar, and Y. Lee, "A QoS oriented framework for adaptive management of web service based workflows.," in *Proceeding of Database and Expert Systems 2003 Conference*, pp. 826–835, 2003.

[20] L. Mei, W. Chan, and T. Tse, "An adaptive service selection approach to service composition," in *Proceedings of the IEEE International Conference on Web Services (ICWS'08)*, IEEE Computer Society, 2008.

[21] K. Regan, P. Poupart, and R. Cohen, "Bayesian reputation modeling in e-marketplaces sensitive to subjectivity, deception and change.," in *Proceedings of AAAI'06*, 2006.

[22] J. Y.-J. Hsu, K.-J. Lin, T.-H. Chang, C.-J. Ho, H.-S. Huang, and W.-R. Jih, "Parameter learning of personalized trust models in broker-based distributed trust management," *Information Systems Frontiers*, vol. 8, no. 4, pp. 321–333, 2006.

[23] K. Murphy, *Software Packages for Graphical Models / Bayesian Networks*. http://www.cs.ubc.ca/~murphyk/Software/bnsoft.html, last accessed October 2008, 2008.

[24] K. Murphy, *Bayes Net Toolbox for Matlab*. http://bnt.sourceforge.net/, last access October 2008, 2007.

[25] *Netica API*. http://www.norsys.com/netica.html, last accessed October 2008, 2008.

[26] N. Miller, P. Resnick, and R. Zeckhauser, "Eliciting informative feedback: The peer-prediction method," *Management Science*, vol. 51, no. 9, pp. 1359–1373, 2005.

[27] R. Jurca and B. Faltings, "Minimum payments that reward honest reputation feedback," in *Proceedings of the ACM Conference on Electronic Commerce*, (Ann Arbor, Michigan, USA), pp. 190–199, June 11-15 2006.