Journal of Universal Computer Science, vol. 16, no. 13 (2010), 1720-1744
submitted: 30/1/10, accepted: 6/5/10, appeared: 1/7/10 © *J.UCS*

Trust-Oriented Composite Service Selection with QoS Constraints

Lei Li, **Yan Wang**
(Department of Computing, Macquarie University, Sydney, Australia
{leili, yanwang}@science.mq.edu.au)

Ee-Peng Lim
(School of Information Systems, Singapore Management University, Singapore
eplim@smu.edu.sg)

Abstract: In Service-Oriented Computing (SOC) environments, service clients interact with service providers for consuming services. From the viewpoint of service clients, the trust level of a service or a service provider is a critical factor to consider in service selection, particularly when a client is looking for a service from a large set of services or service providers. However, a invoked service may be composed of other services. The complex invocations in composite services greatly increase the complexity of trust-oriented service selection. In this paper, we propose novel approaches for composite service representation, trust evaluation and trust-oriented composite service selection (with QoS constraints). Our experimental results illustrate that compared with the existing approaches our proposed trust-oriented (QoS constrained) composite service selection algorithms are realistic and enjoy better efficiency.

Key Words: composite service, composite service selection, composite service representation, trust evaluation, Monte Carlo method
Category: H.3.3, H.3.5, H.4.m

1 Introduction

In recent years, Service-Oriented Computing (SOC) has emerged as an increasingly important research area attracting much attention from both the research and industry communities. In SOC applications, a variety of services across domains are provided to clients in a loosely-coupled environment. Clients can look for preferred and qualified services via a discovery service of registries, invoke and receive services from the rich service environments [Papazoglou et al. 2008].

In SOC, a service can refer to a transaction, such as selling a product online (i.e. the traditional online service), or a functional component implemented by using Web service technologies [Papazoglou et al. 2008]. Quality of Service (QoS) is essential when a set of quality metrics have to be achieved during service provision. These metrics should be measurable and constitute a description of what a service can offer. The QoS of IT service is often expressed in terms of capacity, latency, bandwidth, number of service requests, number of incidents, etc. However, when a client looks for a service from a large set of services offered by different providers, in addition to functionality and QoS, the reputation-based trust is also a key factor for service selection. It is also a critical task for service registries to be responsible for maintaining the list of reputable and trustworthy services and service providers with their offered QoS values, and making these information available to clients [Vu et al. 2005].

Trust is the measure taken by one party on the willingness and ability of another party to act in the interest of the former party in a situation [Knight and Chervany 1996]. Trust is also the subjective probability by which, party A expects that another party B performs a given action if the trust value is in the range of [0,1] [Jøsang et al. 2007].

Different from P2P information-sharing networks or eBay reputation management system, where a binary rating system is used [Xiong and Liu 2004], in SOC environments [Vu et al. 2005, Wang and Lim 2008, Wang et al. 2009], a trust rating is usually a value in the range of [0,1] given by a service client, representing the subjective belief of the service client on the satisfaction of a service or a service provider. The trust value of a service or a service provider can be calculated by a trust management authority based on the collected ratings representing the reputation of the service or the service provider.

However, trust management is a very complex issue in SOC environments. To satisfy the specified functionality and QoS requirements, a service may invoke other services forming a composite service with complex invocations and trust dependencies among its component services [Menascé 2004]. Meanwhile, given a set of various services, different compositions may lead to different service structures. Although these certainly enrich the service provision, they greatly increase the computation complexity and thus make trustworthy service selection with QoS constraints a very challenging task.

In the literature, there are some existing studies for service composition and quality driven service selection [Adamopoulou et al. 2007, Haddad et al. 2008, Menascé 2004, Xiao and Boutaba 2005, Yu et al. 2007, Zeng et al. 2003]. However, for trust-oriented composite service selection, some research problems remain open.

1. The proper definition of the graph representation of composite services including both probabilistic and parallel invocations is still lacking. It is fundamental and important to define such representation to support the global trust evaluation of composite services.

2. From the definitions in [Jøsang et al. 2007, Knight and Chervany 1996], trust can be taken as the *subjective probability*, i.e. *the degree of belief an individual has in the truth of a proposition* [Hamada et al. 2008, Hines et al. 2003], rather than the *objective probability* or *classical probability*, which is *the occurrence frequency of an event* [Hines et al. 2003]. A subjective probability is derived from an individual's personal judgment about a specific outcome (e.g., the evaluation of teaching quality or service quality). It differs from person to person. Hence, the classical probability theory is not a good fit for trust evaluation. Instead, *subjective probability theory* [Hamada et al. 2008, Hines et al. 2003] should be adopted for trust evaluation.

3. Although there are a variety of trust evaluation methods in a number of different areas [Campo et al. 2006, Vu et al. 2005, Wang et al. 2009, Xiong and Liu 2004], no proper mechanism exists for evaluating the global trust of a composite service with a complex structure over service components with different trust values.

4. Taking trust evaluation and the complex structure of composite services into account, effective algorithms are needed for trust-oriented composite service selection (with QoS constraints), and are expected to be more efficient than the existing approaches [Menascé 2004, Yu et al. 2007].

In this paper, we first present the service invocation graph for composite service representation. In addition, we propose a trust evaluation method for composite services based on Bayesian inference, which is an important component in subjective probability theory. Furthermore, we propose composite service selection algorithms based on Monte Carlo method. Experiments have been conducted on composite services with various sizes to compare the proposed model with the existing exhaustive search method [Menascé 2004]. The results illustrate that our proposed algorithms are realistic and more efficient.

This paper is organized as follows. Section 2 reviews existing studies in service composition, service selection and trust management. Section 3 presents our proposed composite services oriented service invocation graph. Section 4 presents a novel trust evaluation method for composite services. In Section 5, Monte Carlo method based algorithms are proposed for trust-oriented composite service selection (with QoS constraints). Experiments are presented in Section 6 for further illustrating the properties of our models. Finally Section 7 concludes our work.

2 Related Work

In SOC environments, the composition of services offered by different providers enriches service provision and offers flexibility to service applications. Medjahed et al. (2003) present some frameworks and algorithms for automatically generating composite services from specifications and rules.

In real applications, the criteria of searching services should take into account not only functionalities but also other properties, such as QoS and trust. In the literature, a number of QoS-aware Web service selection mechanisms have been developed, aiming at QoS improvement in composite services. In [Zeng et al. 2003], a general and extensible model is presented to evaluate the QoS of composite services. Based on their model, a service selection approach has been introduced using linear programming techniques to compute optimal execution plans for composite services. The work in [Haddad et al. 2008] addresses the selection and composition of Web services based on functional requirements, transactional properties and QoS characteristics. In this model, services are selected in a way that satisfies user preferences, expressed as weights over

QoS and transactional requirements. In [Xiao and Boutaba 2005], an autonomic service provision framework is presented for establishing QoS-assured end-to-end communication paths across domains. Their algorithms can provide QoS guarantees over domains. The above works have their merits in different aspects. However, none of them has taken parallel invocation into account, which is fundamental and one of the most common existing invocations in composite services [Menascé 2004, Yu et al. 2007].

With different kinds of invocations including parallel invocation, composite service selection with QoS constraints can be modeled as the Multi-Constrained Optimal Path (MCOP) problem, and several algorithms have been proposed to process the MCOP selection. In [Menascé 2004], an exhaustive search method is adopted to measure service execution time and cost involving probabilistic, parallel, sequential and fastest-predecessor-triggered invocations. However, the algorithm complexity is exponential. In [Korkmaz and Krunz 2001], the H_MCOP algorithm is proposed to select the multi-constrained optimal path with the utility function

$$g_\lambda(p) = \sum_{i=1}^{m} (\frac{q_i(p)}{Q_i})^\lambda, \tag{1}$$

where $\lambda \geq 1$; $q_i(p)$ is the aggregated value of the i^{th} QoS attribute of path p; Q_i is the i^{th} QoS constraint of path p. This algorithm adopts both backward and forward Dijkstra's algorithm [Dijkstra 1959] in optimal path selection. In [Yu et al. 2007], the MCSP_K algorithm is proposed to process the QoS-driven composite service selection. By taking the utility function

$$\xi(p) = \max\{(\frac{q_i(p)}{Q_i})\}, \tag{2}$$

this algorithm keeps the paths with up to K minimum ξ values at each intermediate service component, i.e. it keeps only K paths from the service invocation root to each intermediate service component. This K-path selection strategy aims to reduce the searching space and thus avoid excessive overhead in obtaining the near-optimal solution. Nevertheless, none of these works addresses any aspect of trust.

The trust issue has been widely studied in many applications. In e-commence environments, the trust management system can provide valuable information to buyers and prevent some typical attacks [Wang and Lim 2008, Zacharia and Maes 2000]. In Peer-to-Peer information-sharing networks, binary ratings work pretty well as a file is either the definitively correct version or not [Yu et al. 2004]. In SOC environments, an effective trust management system is critical to identify potential risks, provide objective trust results to clients and prevent malicious service providers from easily deceiving clients and leading to their huge monetary loss [Vu et al. 2005]. In social networks, many approaches are proposed to analyze social relationships to identify some attacks [Jung 2009, Liu et al. 2010a, Liu et al. 2010b].

In general, the trust from a service client on a service or a service provider can be taken as an extent with which the service client *believes* that the service provider

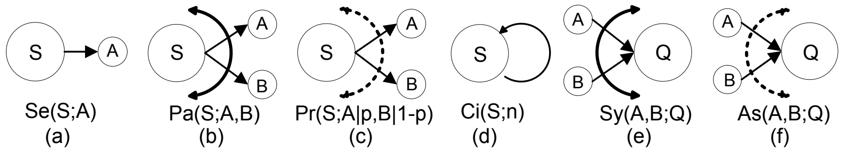

Figure 1: Atomic invocations

can satisfy the client's requirement with desirable performance and quality. Thus, as we have pointed out in Section 1, trust is a *subjective belief* and it is better to adopt *subjective probability theory* [Hines et al. 2003] to deal with trust.

There are some works to deal with subjective ratings. In [Jøsang 2002], a framework is described for combining and assessing subjective ratings from different sources based on Dempster-Shafer belief theory. In [Wang and Singh 2007], a bijection is set up from subjective ratings to trust values with a mathematical understanding of trust in a variety of multiagent systems. However, their models use either a binary rating (positive or negative) system or a triple rating (positive, negative or uncertain) systems that are more suitable for security-oriented or P2P file-sharing trust management systems.

As pointed in [Yu et al. 2004], in richer service environments such as SOC or e-commerce, a rating in $[0, 1]$ is more suitable. In [Xu et al. 2007], a reputation-enhanced QoS-based Web service discovery algorithm is proposed for service matching and selection based on existing Web service technologies. In [Malik and Bouguettaya 2009], a set of decentralized techniques are proposed aiming at evaluating reputation-based trust with the ratings from peers to facilitate trust-based selection and service composition. However, in these works, neither service invocation nor composite service structure are taken into account. Taking the complex structure of composite services into account, effective algorithms are needed for trust-oriented composite service selection.

3 Service Invocation Model

In this section, we present the definitions of our proposed service invocation graph for representing the complex structures of composite services. They are essential for our trust-oriented composite service selection algorithms to be introduced in Section 5.

3.1 Composite Services and Invocation Relation

A *composite service* is a conglomeration of services with invocation relations between them. Six atomic invocation relations [Li and Wang 2009, Li et al. 2009] are depicted as follows and in Fig. 1.

- *Sequential Invocation*: A service S invokes its unique succeeding service A. It is denoted as $\mathsf{Se}(S : A)$ (see Fig. 1(a)).

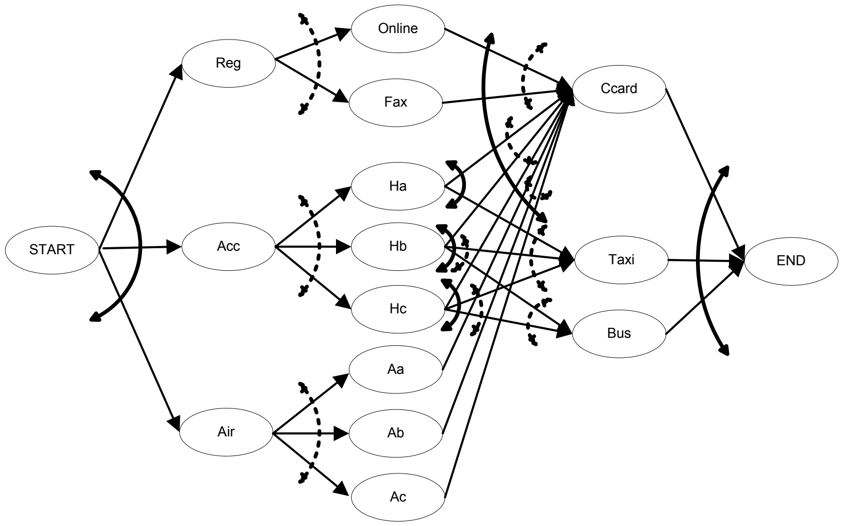

Figure 2: The *SIG* for the travel plan of Smith

- *Parallel Invocation*: A service S invokes its succeeding services in parallel. E.g., if S has successors A and B, it is denoted as $\mathsf{Pa}(S : A, B)$ (see Fig. 1(b)).
- *Probabilistic Invocation*: A service S invokes its succeeding services each with a probability. E.g., if S invokes successors A with the probability p and B with the probability $1 - p$, it is denoted as $\mathsf{Pr}(S : A|p, B|1 - p)$ (see Fig. 1(c)).
- *Circular Invocation*: A service S invokes itself for n times. It is denoted as $\mathsf{Ci}(S|n)$ (see Fig. 1(d)). A circular invocation can be unfolded by cloning itself n times [Yu et al. 2007]. Hence, it can be replaced by Se in advance.
- *Synchronous Activation*: A service Q is activated only when all its preceding services have been completed. E.g., if Q has synchronous predecessors A and B, it is denoted as $\mathsf{Sy}(A, B : Q)$ (see Fig. 1(e)).
- *Asynchronous Activation*: A service Q is activated as the result of the completion of one of its preceding services. E.g., if Q has asynchronous predecessors A and B, it is denoted as $\mathsf{As}(A, B : Q)$ (see Fig. 1(f)).

3.2 An Example: Travel Plan

Here we introduce an example of composite services.

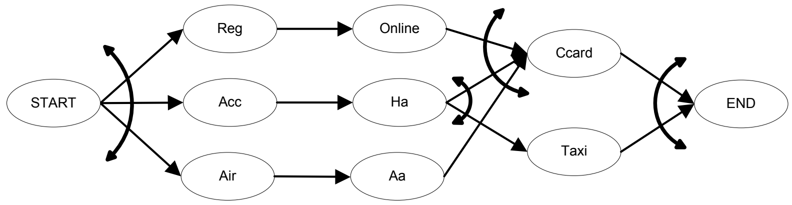

Figure 3: A service execution flow (*SEF*)

Smith in Sydney, Australia is making a travel plan to attend an international conference in Stockholm, Sweden. His plan includes conference registration, airline from Sydney to Stockholm, accommodation and local transportation.

Regarding conference registration *Reg*, Smith could pay *Online* or by *Fax* with a credit card *Ccard*. Regarding accommodation reservation *Acc*, Smith could make a reservation at Hotel *Ha*, *Hb* or *Hc* with credit card *Ccard*. According to the hotel choice, Smith could arrange the local transportation, e.g., take a *Taxi* to *Ha*, take a *Taxi* or a *Bus* to either *Hb* or *Hc*. Regarding airplane booking *Air*, Smith could choose from Airlines *Aa*, *Ab* and *Ac* with the credit card *Ccard* for the payment. Smith chooses the services according to their trust values. He will have a higher probability to choose the service with a better trust value.

In this example, with a starting service *START* and an ending service *END*, the composite services consisting of all possibilities of the travel plan can be depicted by a service invocation graph (*SIG*) (Fig. 2). One of all feasible travel plans is a service execution flow as depicted in Fig. 3.

3.3 Service Invocation Graph

The structure of a composite service can be represented by a service invocation graph (*SIG*), with the initial definition as follows.

Definition 1. The *service invocation graph* (*SIG*) is a directed graph $G = (V, E, R)$, where V is a finite set of vertices, E is a finite set of directed edges and R is the set of atomic invocations Se, Pa, Pr, Ci, Sy and As. In G, each vertex $v \in V$ represents a service. $\forall e = (v_1, v_2) \in E$ $(v_1, v_2 \in V)$ is a directed edge, where v_1 is the *invoking vertex* and v_2 is the *invoked vertex*. Here v_1 is the *direct predecessor* of v_2 and v_2 is the *direct successor* of v_1. It is denoted as $v_1 \succeq v_2$.

Definition 2. Given a service invocation graph $G = (V, E, R)$, vertex $v_2 \in V$ is *invocational* from vertex $v_1 \in V$ if $(v_1, v_2) \in E$ or there is a directed path P in G where v_1 is the staring vertex and v_2 is the ending vertex. If v_2 is invocational from v_1, it is denoted as $v_1 \succ v_2$.

In addition, if $v_1 \succ v_2$, v_1 is the predecessor of v_2 and v_2 is the successor of v_1. Obviously, the *invocational* relation is transitive, i.e. if $v_1 \succ v_2$, $v_2 \succ v_3$, then $v_1 \succ v_3$.

Definition 3. In a service invocation graph, the *service invocation root* is the entry vertex without any predecessors, and the *service invocation terminal* is the exit vertex without any successors.

Based on the above definitions, *SIG* is well-defined as follows.

Definition 4. A composite service can be represented by a *service invocation graph*

$$SIG = (V, I_p, R_p, I_s, R_s), \tag{3}$$

where

- In an *SIG*, there are only one service invocation root *START* and only one service invocation terminal *END*;
- $V = \{v_i | v_i$ is a vertex, $v_i =$*START* or *START* $\succ v_i\}$;
- $I_p = \{I_{p_i} | v_i \in V\}$ and I_{p_i} is a set of direct predecessors invoking v_i, i.e. $I_{p_i} = \{p_{ij} | p_{ij} \succeq v_i\}$;
- R_p represents a set of activation relations between I_p and V, which includes atomic activations Sy and As;
- $I_s = \{I_{s_i} | v_i \in V\}$ and I_{s_i} is a set of direct successors invoked by v_i, i.e. $I_{s_i} = \{s_{ij} | v_i \succeq s_{ij}\}$;
- R_s represents a set of invocation relations between V and I_s, which includes atomic invocations Se, Pa, Pr and Ci.

Let $\emptyset$ denote the empty invocation relation set. In an *SIG*, if $I_{p_i} = \emptyset$, then $v_i =$ *START*. Similarly, if $I_{s_i} = \emptyset$, then $v_i =$ *END*.

Definition 5. A *service execution flow* (*SEF*) of an *SIG* $G = (V, I_p, R_p, I_s, R_s)$ is a graph $G' = (V', E', R')$, where R' contains Se, Pa, Sy and Ci, $V' \subseteq V$ and $E' \subseteq E$. In addition, $\forall v' \in V'$, v' is invocational from service invocation root *START* of G, and service invocation terminal *END* of G is invocational from v'.

4 Trust Evaluation in Composite Services

In this section, we introduce our trust evaluation models for composite services. In Section 4.1, a trust estimation model is proposed to estimate the trust value of each service component from a series of ratings according to Bayesian inference[Hamada et al. 2008, Hines et al. 2003], which is an important component in subjective probability theory. These ratings are provided by service clients and stored by a trust management authority. In Section 4.2, a global trust computation model is proposed to compute the global trust value of a composite service based on the trust values of all service components.

4.1 Trust Estimation Model

Since subjective probability is a person's degree of belief concerning a certain event [Hamada et al. 2008, Hines et al. 2003], the trust rating in $[0, 1]$ of a service given by a service client can be taken as the *subjective possibility* with which the service provider can perform the service satisfactorily. Hence, *subjective probability theory* is the right tool for dealing with trust ratings. In this paper, we adopt *Bayesian inference*, which is an important component in *subjective probability theory*, to estimate the trust value of a provided service from a set of ratings. Each rating is a value in $[0, 1]$ evaluated from the subjective judgements of a service client.

The goal of adopting *Bayesian inference* [Hamada et al. 2008, Hines et al. 2003] is to summarize the available information that defines the distribution of trust ratings through the specification of probability density functions, such as: prior distribution and posterior distribution. The *prior distribution* summarizes the subjective information about the trust prior to obtaining the ratings sample $x_1, x_2, \ldots, x_n$. Once the sample is obtained, the prior distribution can be updated. The updated probability distribution on trust ratings is called the *posterior distribution*, because it reflects probability beliefs posterior to analyzing ratings.

According to [Hu et al. 2006], if all service clients give ratings for the same service, the provided ratings conform to normal distribution. The complete set of ratings can be collected based on honest-feedback-incentive mechanisms [Jurca and Faltings 2006, Jurca and Faltings 2007]. Let μ and σ denote the mean and the variance of ratings respectively in the normal distribution. Thus, a sample of ratings $x_1, x_2, \ldots, x_n$ ($x_i \in [0, 1]$) has the normal density with mean μ and variance σ. In statistics, when a ratings sample with size n is drawn from a normal distribution with mean μ and variance σ, the mean of the ratings sample also conforms to a normal distribution which has mean μ and variance $\sigma/\sqrt{n}$ [Hamada et al. 2008]. Let $\delta \in [0, 1]$ denote the prior subjective belief about the trust of a service that a client is requesting for. We can assume that the prior normal distribution of μ has mean δ and variance $\sigma/\sqrt{n}$, i.e.

$$f(\mu) = \begin{cases} \frac{\sqrt{n}}{\sigma\sqrt{2\pi}} e^{\frac{n(\mu-\delta)^2}{-2\sigma^2}}, & 0 < \mu < 1; \\ 0, & \text{otherwise}. \end{cases} \tag{4}$$

The posterior density for μ can be estimated [Li et al. 2009]

$$f(\mu|x_1, x_2, \ldots, x_n) = \frac{f(x_1, x_2, \ldots, x_n; \mu)}{f(x_1, x_2, \ldots, x_n)} \tag{5}$$

$$= \frac{\frac{\sqrt{n}}{\sigma^{n+1}(2\pi)^{\frac{n+1}{2}}} e^{\frac{\Sigma x_i^2 - 2\mu n\bar{x} + n\mu^2 + n(\mu-\delta)^2}{-2\sigma^2}}}{\frac{1}{\sqrt{2}\sigma^n(2\pi)^{\frac{n}{2}}} e^{\frac{\Sigma x_i^2 + n\delta^2 - \frac{n(\bar{x}+\delta)^2}{2}}{-2\sigma^2}}} = \frac{\sqrt{n}}{\sigma\sqrt{\pi}} e^{\frac{n(\mu - \frac{\bar{x}+\delta}{2})^2}{-\sigma^2}}. \tag{6}$$

Therefore, the posterior distribution of μ is normal with mean $\frac{\bar{x}+\delta}{2}$ and variance $\sigma/\sqrt{2n}$. If the loss function is squared error [Hamada et al. 2008, Hines et al. 2003], the mean of the posterior normal distribution can be used as the estimation of trust value from ratings. Hence,

Theorem 6. The Bayesian estimation of the trust value of a service with n ratings $x_1, x_2, \ldots, x_n$ ($x_i \in [0,1]$) is

$$T(x_1, x_2, \ldots, x_n, \delta) = \frac{\bar{x} + \delta}{2} = \frac{\Sigma_{i=1}^{n} x_i + n\delta}{2n}, \tag{7}$$

where $\delta \in [0,1]$ denotes the requesting client's prior subjective belief about the trust.

If the requesting client has no prior subjective information about the trust of the requested service, by default, let $\delta = \frac{1}{2}$ since $\frac{1}{2}$ is the middle point of $[0,1]$ representing the neutral belief between distrust and trust. After the Bayesian inference, the Bayesian estimation of the trust can be taken as the requesting client's prior subjective belief about the trust for the Bayesian inference next time.

Now we can estimate the trust of a requested service by combining the requesting client's prior subjective belief about the trust and ratings. Since trust is subjective, it is more reasonable to include the requesting client's prior subjective belief about the trust.

4.2 Global Trust Computation in Composite Services

Our goal is to select the optimal one from multiple *SEF*s (service execution flows) in an *SIG* aiming at maximizing the global trust value of *SEF*, which is determined by the trust values of vertices and invocation relations between vertices in the *SEF*.

According to Definition 5, in *SEF* we only need consider Se (Fig. 1 (a)), Pa (Fig. 1 (b)) and Sy (Fig. 1 (e)). From Se and Pa, Sy in *SEF* can be determined. Due to space constraints, the details are omitted. Hence, there are two kinds of atomic structures to determine the trust value of an *SEF*: Se and Pa. Se in the *SEF* can be selected from the service invocation relation Se (Fig. 1(a)) or Pr (Fig. 1(c)) in the *SIG*. Pa in the *SEF* can be selected from the service invocation relation Pa (Fig. 1 (b)) in the *SIG*.

Definition 7. The global trust value T_g of an Se structure where service S uniquely invokes service A (see Fig. 1 (a)) can be computed by

$$T_g = T_S \cdot T_A, \tag{8}$$

where T_S and T_A are the trust values of S and A respectively, which are evaluated from Theorem 6. Since S and A are independent, the probability that S and A both occur is equal to the product of the probability that S occurs and the probability that A occurs.

Definition 8. The global trust value T_g of a Pa structure where service S invokes services A and B in parallel (see Fig. 1 (b)) can be computed from T_S and the combined trust value T_{AB} by Definition 7, and

$$T_{AB} = \frac{\omega_1}{\omega_1 + \omega_2} \cdot T_A + \frac{\omega_2}{\omega_1 + \omega_2} \cdot T_B, \tag{9}$$

where T_S, T_A and T_B are the trust values of S, A and B respectively, which are evaluated from Theorem 6; ω_1 and ω_2 are weights for A and B respectively which are specified in a requesting client's preference or specified as the default value by the service trust management authority.

According to Definitions 7 & 8, each atomic structure Se or Pa can be converted to a single vertex. Hence, in the process of trust computation, an *SEF* consisting of Se and Pa structures can be incrementally converted to a single vertex with its trust value computed as the global trust. Due to space constraints, we briefly introduce the following global trust computation algorithm. For details, please refer to [Li and Wang 2009].

Global Trust Computation Algorithm. In order to obtain the global trust value of an *SEF*, firstly the trust value of each atomic Se structure in the *SEF* should be computed by Definition 7. Each computed atomic Se structure is then taken as a vertex in the *SEF*. After that, the trust value of each atomic Pa structure is computed by Definition 8. Similarly, each computed atomic Pa structure is then taken as a vertex in the *SEF*. Thus, the computation can repeat until the final *SEF* is simplified as a vertex, and the global trust value is obtained.

5 Trust-Oriented Composite Service Selection

Here we assume that a service trust management authority stores a large volume of services with their ratings. In response to a client's request, the service trust management authority first generates an *SIG* containing all relevant services and invocation relations. Then, the trust-oriented (QoS constrained) service selection algorithm is applied to find the most trustworthy *SEF* (satisfying QoS constraints).

5.1 Monte Carlo Method Based Algorithm (MCBA) in Trust-Oriented Composite Service Selection without QoS Constraints

If there are only Pr (probabilistic invocation) structures in an *SIG* (i.e. there are only Se (sequential invocation) structures in the *SEF*), the *SEF* is a path in the *SIG*. By extending Dijkstra's shortest path algorithm [Dijkstra 1959], the optimal *SEF* can be determined as an execution flow (path) from *START* to *END* so that the multiplication of trust values of all vertices in the path is the maximal according to Definition 7.

If there are only Pa structures in an *SIG*, the unique *SEF* is the same as the *SIG*.

If an *SIG* consists of both Prs and Pas, since there is no existing method to consider such kind of structure as we have analyzed in Section 2, we propose a *Monte Carlo method based algorithm* (*MCBA*) to find the optimal *SEF*.

Monte Carlo method [Gentle et al. 2004] is a computational algorithm which relies on repeated random sampling to compute results. It tends to be adopted when it is infeasible to compute an exact result with a deterministic algorithm. Monte Carlo method is useful for modeling phenomena with significant uncertainty in inputs, such as the calculation of risk in business [Gentle et al. 2004]. The specific areas of application of the Monte Carlo method include computational physics, physical chemistry, global illumination computations, finance and business, and computational mathematics (e.g., numerical integration and numerical optimization) [Gentle et al. 2004, Morton and Popova 2009]. It is also one of the techniques for solving NP-complete problems [Gentle et al. 2004, Morton and Popova 2009].

The main strategy in *MCBA* is as follows. In an *SIG*, the direct successors of a service need to be selected according to their trust values. Usually, the direct successor with a larger trust value is preferred, which indicates a higher probability to be invoked, and vice versa. Then, according to this, a uniform distributed random number is generated to decide which succeeding service is selected.

When determining the optimal *SEF* from an *SIG*, we only need *MCBA* for Pr structures. Let's take Pr in Fig. 1(c) as an example to explain the details of our *MCBA*. If successor A has a trust value T_A computed following Theorem 6 and successor B has a trust value T_B computed following Theorem 6, the probability for vertex S to select successor A is

$$P_A = \frac{T_A}{T_A + T_B}. \tag{10}$$

Similarly, the probability to select successor B is

$$P_B = \frac{T_B}{T_A + T_B}. \tag{11}$$

Obviously, $0 < P_A, P_B < 1$. Then a uniform distributed random number r_0 in $(0, 1)$ is generated to decide which successor is selected. In detail, if $r_0 < P_A$, successor A is selected; If $P_A < r_0 < P_A + P_B = 1$, successor B is selected.

Therefore, given an *SIG*, an *SEF* could be obtained by repeating *MCBA* from the service invocation root *START* until the service invocation terminal *END* is reached. Once an *SEF* is generated, its global trust value can be calculated by global trust computation algorithm in Section 4.2. By repeating this process for l simulation times, a set of *SEF*s can be generated, from which the locally optimal *SEF* with the maximal global trust value can be obtained. A high value of l is necessary to obtain the optimal solution. *MCBA* for trust-oriented composite service selection is illustrated in Algorithm 1.

In Theorem 6, the trust estimation algorithm has a complexity of $O(n)$ with n ratings. Hence, in global trust computation algorithm in Section 4.2, the complexity of trust evaluation for a composite service with N services is $O(nN)$. Therefore, *MCBA* with l simulations incurs a complexity of $O(nlN)$.

Algorithm 1 *MCBA* for Trust-Oriented Composite Service Selection

Input: Simulation times l; *SIM*, and service ratings *Reputation*.
Output: The optimal *SEF* with maximum global trust value $Trust_{global}$.

Let *Trust* be the trust value for each service evaluated from *Reputation* by Theorem 6;
for all i such that $1 \le i \le l$ **do**
  Initialize $active = [root]$, *SEF*$= [root]$;
  while $active \neq \emptyset$ **do**
    Select a vertex $vertex$ from $active$, and remove $vertex$ from $active$;
    Let vectors Pr and Pa be the Pr and Pa structures from $vertex$;
    if vector $Pa \neq \emptyset$ **then**
      if $vertex$ is in *SEF* **then**
        for all $Pa(j)$ in Pa **do**
          if $Pa(j)$ is not in *SEF* **then**
            Add $Pa(j)$ into *SEF*
          end if
        end for
      end if
      for all $Pa(j')$ in $Pa(j)$ **do**
        if $Pa(j')$ is not $terminal$ and $Pa(j')$ is not in $active$ **then**
          Add $Pa(j')$ into $active$
        end if
      end for
    end if
    if vector $Pr \neq \emptyset$ **then**
      if $vertex$ is in *SEF* **then**
        if none of Pr is in *SEF* **then**
          for all $Pr(k)$ in Pr **do**
            Generate a uniform distributed random number $rand$ in $[0, 1]$;
            Select the smallest k' such that $rand < Trust(k')/sum(Trust(k))$
          end for
          Add $Pr(k')$ in *SEF*
        end if
      end if
      if $Pr(k')$ is not $terminal$ and $Pr(k')$ is not in $active$ **then**
        Add $Pr(k')$ into $active$
      end if
    end if
  end while
  Let $Trust_{SEF}$ be the trust value of *SEF* according to Global Trust Computation Algorithm
  $Trust_{global} = \max Trust_{SEF}$;
end for
return Optimal *SEF* and $Trust_{global}$.

5.2 QoS Constrained Monte Carlo Method Based Algorithm (QC_MCBA) in Trust-Oriented Composite Service Selection with QoS Constraints

The trust-oriented composite service selection with QoS constraints can be modeled as the Multi-Constrained Optimal Path (MCOP) problem, which is an NP-complete problem [Korkmaz and Krunz 2001, Yu et al. 2007]. In composite services, each service component can be associated with multiple QoS attributes, which can be roughly classified as additive or non-additive [Korkmaz and Krunz 2001].

- The aggregated value of an *SEF* with respect to an additive QoS attribute, such as delay, cost, execution time, etc, is given by the sum of QoS values of service components along that *SEF* [Menascé 2004]. In addition, multiplicative constraints, e.g., reliability, can be transformed into additive constraints [Korkmaz and Krunz 2001].
- In contrast, for non-additive QoS attributes (e.g., bandwidth), the aggregated value of an *SEF* is determined by the value of that QoS attribute at the bottleneck.

  It is known that constraints associated with non-additive QoS attributes can be easily dealt with a preprocessing step by pruning all service components that do not satisfy these constraints to simplify the structure of composite services.

  Therefore, in this paper, we will mainly focus on additive QoS attributes and assume that composite service selection with QoS constraints is only based on additive QoS attributes.

Selecting the optimal *SEF* with QoS constraints is an NP-complete problem. For this problem, we propose a ***QoS constrained Monte Carlo method based algorithm*** (*QC_MCBA*) to find the most trustworthy *SEF* satisfying QoS constraints.

The main strategy in *QC_MCBA* is as follows. In an *SIG*, the direct successor of a service needs to be selected according to the values of the utility function defined by

$$U_{\omega_3\omega_4}(X) = \begin{cases} \omega_3 \cdot T(X) + \sum_{i=1}^{m} (\frac{Q_i - q_i(X)}{Q_i})^{\omega_4}, & \forall Q_i \geq q_i(X) \\ 0, & \exists Q_i < q_i(X) \end{cases}, \quad (12)$$

where ω_3 and ω_4 ($\omega_4 \geq 1$) are the weights for trust and all QoS attributes respectively specified in a requesting client's preference or specified as default values by the service trust management authority; $T(X)$ is the trust value of direct successor X computed following Theorem 6; $q_i(X)$ is the aggregated value of the i^{th} QoS attribute about *SEF'*, which is part of the *SEF* from the service invocation root to service component X; Q_i is the i^{th} QoS constraint and m is the total number of QoS constraints.

In *QC_MCBA*, the direct successor with a larger utility value is preferred, which indicates a higher probability to be invoked. Then, according to this, a uniform distributed random number is generated to decide which succeeding service is selected. When determining the optimal *SEF* with QoS constraints from an *SIG*, we only need *QC_MCBA* for Pr structures. Let's take the Pr structure in Fig. 1(c) as an example to explain the details of *QC_MCBA*. If successor A has the utility value $U_{\omega_3\omega_4}(A)$ and successor B has the utility value $U_{\omega_3\omega_4}(B)$, the probability for vertex S to select successor A is

$$P_A = \frac{U_{\omega_3\omega_4}(A)}{U_{\omega_3\omega_4}(A) + U_{\omega_3\omega_4}(B)}. \quad (13)$$

Similarly, the probability to select successor B is

$$P_B = \frac{U_{\omega_3\omega_4}(B)}{U_{\omega_3\omega_4}(A) + U_{\omega_3\omega_4}(B)}. \quad (14)$$

Table 1: Ratings & subjective belief of each service component in the travel plan example

	Reg	*Acc*	*Air*	*Online*	*Fax*	*Ha*	*Hb*	*Hc*	*Aa*	*Ab*	*Ac*	*Ccard*	*Taxi*	*Bus*
x_1	0.88	0.83	0.78	0.92	0.51	0.17	0.35	0.89	0.30	0.95	0.25	0.95	0.94	0.32
x_2	0.84	0.82	0.87	0.92	0.38	0.18	0.32	0.86	0.36	0.98	0.30	0.95	0.86	0.37
x_3	0.97	0.85	0.77	0.94	0.25	0.22	0.46	0.82	0.34	0.91	0.24	0.96	0.86	0.34
x_4	0.87	0.82	0.83	0.96	0.40	0.12	0.34	0.87	0.29	0.91	0.31	0.96	0.89	0.18
x_5	0.91	0.74	0.79	0.95	0.41	0.16	0.28	0.88	0.41	0.97	0.29	0.96	0.90	0.35
δ	0.92	0.85	0.91	0.95	0.32	0.20	0.50	0.91	0.32	0.92	0.51	0.98	0.89	0.33

Table 2: Weights of service components in Pa

Reg	*Acc*	*Air*	*Ccard*	*Taxi*	*Ccard*	*Bus*
0.1	0.3	0.6	0.6	0.4	0.6	0.4

Obviously, $0 < P_A, P_B < 1$. Then a uniform distributed random number r_0 in $(0, 1)$ is generated to decide which successor is selected. In detail, if $r_0 < P_A$, successor A is selected; If $P_A < r_0 < P_A + P_B = 1$, successor B is selected.

Therefore, given an *SIG*, a feasible *SEF* satisfying QoS constraints could be obtained by repeating *QC_MCBA* from *START* until *END* is reached. Once a feasible *SEF* is generated, its global trust value can be calculated by the global trust computation algorithm in Section 4.2. By repeating this process for l simulation times, a set of feasible *SEF*s can be generated, from which the locally optimal QoS constrained *SEF* with the maximal global trust value can be obtained. The value of l determines the performance and overhead of *QC_MCBA*. If l is large enough, this algorithm can obtain the optimal solution but its computational cost will be very high.

Our proposed *MCBA* & *QC_MCBA* are not designed to consider all *SEF*s in composite services. If we know the information of service components (such as: trust values and QoS values), after l simulation times, a set of feasible *SEF*s with better trust values are generated, from which the locally optimal *SEF* can be obtained. Therefore, the selection process in *MCBA* & *QC_MCBA* is performed at run time, rather than design time, making our proposed method practical in applications.

6 Experiments

In this section, we will illustrate the results of our experiments to evaluate the trust-oriented composite service selection strategy in *MCBA* and *QC_MCBA*.

6.1 Experiment on Trust-Oriented Composite Service Selection

6.1.1 Comparison Using Travel Plan Composite Services

In this experiment, we compare our proposed *MCBA* with the exhaustive search method by applying it to the travel plan composite services (with 16 vertices and 30 *SEF*s). The

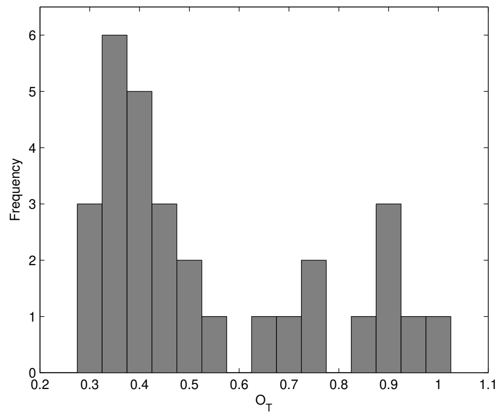

Figure 4: Histograph of O_T for each *SEF*

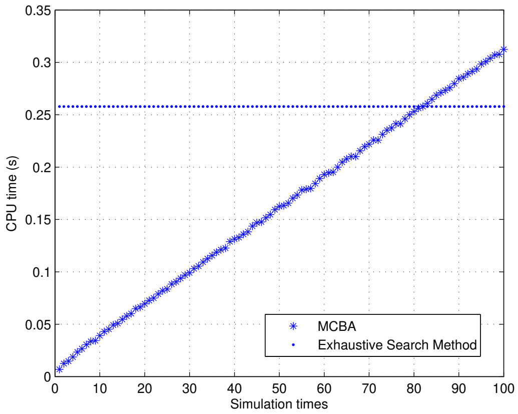

Figure 5: CPU time with different simulation times

corresponding ratings and Smith's prior subjective belief of each service component are listed in Table 1. The weights of service components in all Pa structures of the composite services are listed in Table 2.

The exhaustive search method is inefficient as it aims to enumerate all solutions. In the work [Menascé 2004], the exhaustive search method is adopted to calculate execution time and cost of all *SEF*s in a composite service.

According to global trust computation algorithm in Section 4.2, the global trust

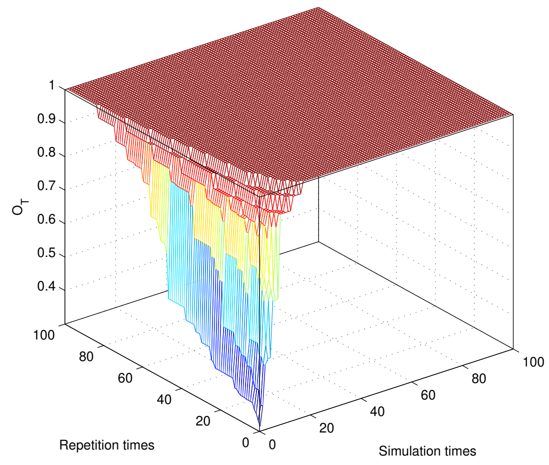

Figure 6: O_T in the travel plan example

value T_i of *SEF* i $(i = 1, 2, \ldots, 30)$ can be calculated. Let ***trust-based SEF optimality*** be

$$O_T(T_i) = \frac{T_i}{\max(T_i)}. \tag{15}$$

The corresponding histograph of $O_T(T_i)$ values of 30 *SEF*s is plotted in Fig. 4. From it, we can observe that 80% of $O_T(T_i)$ values are less than 0.8, implying that if we select an *SEF* randomly, it is very likely to obtain an *SEF* with a low trust value.

In *MCBA*, there are multiple simulations, in each of which an *SEF* is generated and its global trust value is calculated. After l simulations, a locally optimal *SEF* can be obtained from l generated *SEF*s. In order to study the distribution of global trust of locally optimal *SEF*s, we take l simulations as a repetition and repeat for m times.

Our experiments use Matlab 7.6.0.324 (R2008a) running on a Dell Vostro V1310 laptop with an Intel Core 2 Duo T5870 2.00GHz CPU and a 3GB RAM. l, the number of simulation times, is set from 1 to 100. m, the number of repetition times, is set from 1 to 100. The experimental results are plotted in Fig. 6. We could observe that with a fixed number of repetitions, the more simulations, the closer to 1 O_T becomes. Namely more simulations lead to a higher probability to obtain the optimal *SEF*.

Furthermore, we compare the execution time of *MCBA* with that of the exhaustive search method. Each CPU time in this paper is the average of ten independent executions. In Fig. 5, we can observe that when the number of simulation times $l \leq 82$, our *MCBA* is faster than the exhaustive search method. From Figs 5 and 6, we can see that the probability to obtain the optimal *SEF* is 97% when there are 20 simulations.

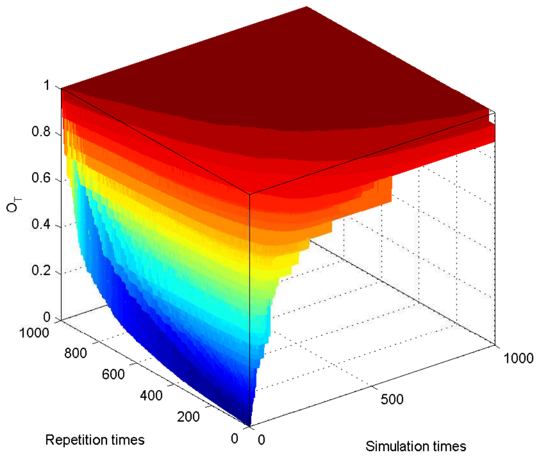

Figure 7: O_T in the composite service of 100 vertices

Meanwhile, the execution time of our *MCBA* is 27% of the one of the exhaustive search method. According to Table 1, theoretically the probability to obtain the optimal *SEF* for each simulation in *MCBA* is 17.8%, due to *SIG* and the strategy in *MCBA* in Section 5.1. Hence after 20 simulations theoretically *MCBA* has the probability of 98.04% to obtain the optimal *SEF*. Hence the experimental result about the probability to obtain the optimal *SEF* confirms to the theoretical conclusion.

With this simple travel plan example, *MCBA* outperforms the exhaustive search method. More significant performance differences can be observed with some complex composite services to be introduced in the next section.

6.1.2 Comparison Using Complex Composite Services

In this experiment, we further compare our proposed *MCBA* and the exhaustive search method on three more complex composite services. The numbers of vertices of these composite services are 35, 52 and 100 respectively. The numbers of Ses, Pas, Prs, Sys, Ass and *SEF*s in corresponding composite services are listed in Table 3.

In this experiment, we use the same platform as the experiment in Section 6.1.1. In the case of composite service with 35 vertices, the *MCBA* takes 0.3219 second to finish 20 simulations with the probability of 95.45% to obtain the optimal *SEF*, while the exhaustive search method uses 17.09 seconds. When the number of vertices becomes 52, our *MCBA* takes 0.8625 second to finish 52 simulations, with which the probability to obtain the optimal *SEF* is 95.29%. However, when taking the same time, the exhaustive

Table 3: Structures of complex composite services

Number of vertices	Ses	Pas	Prs	Sys	Ass	*SEF*s
35	17	8	11	4	11	1.8×10^3
52	24	13	16	7	16	5.4×10^4
100	51	24	32	12	32	2.92×10^9

Table 4:CPU time of *MCBA* & exhaustive search method with different composite services

Number of vertices	16	35	52	100
Probability to obtain the optimal *SEF* for each simulation	17.84%	14.31%	5.71%	0.33%
Number of simulation times in *MCBA*	20	20	52	925
Probability to obtain the optimal *SEF* for *MCBA*	98.04%	95.45%	95.29%	95.12%
CPU time (seconds) of *MCBA*	0.0695	0.3219	0.8625	34.51
CPU time (seconds) of exhaustive search method	0.2578	17.09	–	–

search method can only search 0.42% of 5.4×10^4 *SEF*s. When taking 1000 times of the *MCBA* CPU time, it can only search approximately 1% of all *SEF*s. We further apply our *MCBA* to a composite service with 100 vertices. It takes 34.51 seconds to finish 925 simulations with a probability of 95.12% to obtain the optimal *SEF*. In contrast, when taking the same time, the exhaustive search method can only search $(9.56 \times 10^{-6})\%$ of 2.92×10^9 *SEF*s. When taking 100 times of the *MCBA* CPU time, it can only search $(1.01 \times 10^{-5})\%$ of all *SEF*s. The above results are listed in Table 4.

In the case of composite service with 100 vertices, the results of *MCBA* are plotted in Fig. 7. When there are $l = 925$ simulation times, *MCBA* can reach the optimal solution with the probability 95.2%. Also it has a great chance to obtain the near-optimal one, even when l is as small as 200. For example, in Fig. 7, when l is 200, the probability for the trust-based *SEF* optimality to be $O_T \geq 0.82$ is about 95.7%.

In summary, our proposed *MCBA* can obtain a near-optimal *SEF* after a certain number of simulations. As the CPU time for a single simulation in *MCBA* is extremely short, our experimental results have illustrated that the overall performance of *MCBA* is good even with complex composite services. In addition, *MCBA* is suitable for parallel computing since each simulation in *MCBA* is independent. This can greatly speed up computations and shorten the overall CPU time. Thus, our proposed *MCBA* is realistic and efficient.

6.2 Experiment on Trust-Oriented Composite Service Selection with QoS Constraints

In this experiment, we compare our proposed *QC_MCBA* with the exhaustive search method by applying it to the composite services listed in Section 6.1. Meanwhile, we adopt the same platform as the one used in Section 6.1 as well.

Table 5: QoS attribute values of each service component in the travel plan example

	root	*Reg*	*Acc*	*Air*	*Online*	*Fax*	*Ha*	*Hb*	*Hc*	*Aa*	*Ab*	*Ac*	*Ccard*	*Taxi*	*Bus*	terminal
cost	0	50	20	50	800	800	1100	1200	1000	2100	2000	2200	50	120	80	0
time	100	80	160	100	30	300	150	160	150	220	200	210	100	180	80	10

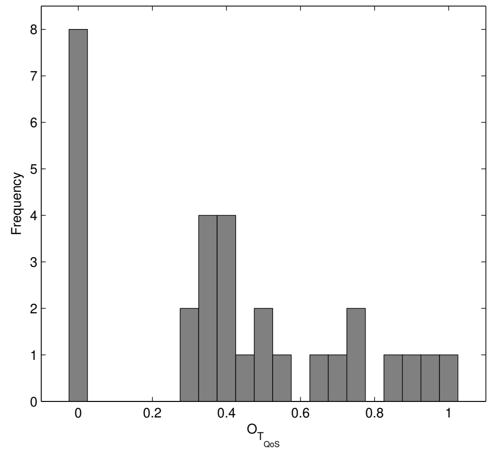

Figure 8: Histograph of $O_{T_{QoS}}$ for each *SEF*

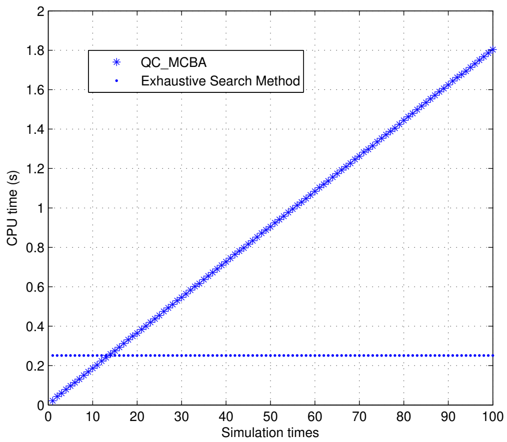

Figure 9: CPU time of *QC_MCBA*