Minimum User-perceived Interference Routing in Service Composition

Li Xiao and Klara Nahrstedt
Department of Computer Science
University of Illinois at Urbana-Champaign, Urbana, IL 61801

***Abstract*— *Service Composition* is a promising technology for providing on-demand composed services in dynamic and loosely coupled peer-to-peer (P2P) networks. Because of system dynamics, such as the peer leaving from the system, end users may perceive *interference* from service disruptions. How to minimize the user-perceived interference and provide Quality of Service guarantees to the composite services thus becomes important and challenging.**

In this paper, we take a novel approach to study the problem of minimum interference service composition with QoS guarantees. First, we propose a general analytic framework to model the interference and its intensity to the end users due to system dynamics. Based on this framework, the minimum interference routing problem is formulated. Then, we present an optimal solution to the problem through dynamic programming and investigate some optimization simplifications in special cases. We further propose a heuristic measure for fast interference calculation and design efficient routing algorithms by exploiting the local path recovery and reliable service paths. Our analysis and extensive simulations demonstrate that our model and algorithms can achieve much better performance than the traditional methods in finding service paths, with respect to decreasing the interference to end users, especially in the scenarios of stringent QoS requirement, highly dynamic networks, or the type of impatient users.

I. Introduction

Based on the existing Internet infrastructure and development of overlay networks and peer-to-peer (P2P) networks, *Service Composition* is becoming a crucial technology to enable on-demand business process provision, web services, multimedia applications, etc. Quite a few research results have been reported recently, such as [1], [2], [3], and [4]. The essential of service composition is the integration of the loosely coupled distributed services (i.e., *service components*) into a composite service to provide a relatively comprehensive function for end users. For example, a user needs a video on-demand service and wants to compose the end-to-end service from the service components (e.g., transcoding, captioning, and translation) in P2P networks. It means the media content is duplicated and stored at several sites, and moreover, the media stream has to go through service components to transcode, caption, and translate the content before it can be finally viewed. Fig. 1 shows two possible composite services for the end user, one from node A to the user and the other from node H to the user. The individual service components are provided by various nodes in P2P networks. Some fundamental services, such as service discovery, e.g., [5] and [6], are provided by the network infrastructure. The purpose of service composition is thus to compose the dispersed services together to serve end users' requirement. We call the composed result *Service Path*. The process of finding an appropriate service path is called *Service Routing*.

Since the service components are hosted by loosely coupled and managed computers, there are two important issues to address in service composition: the quality of the service path and the failure recovery in service disruptions.

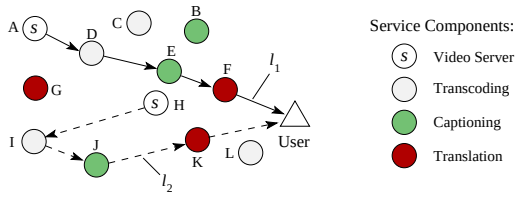


Fig. 1. An example of service composition. Each node holds only one service component. Different types of nodes represent different service components. The lines with arrows represents service paths. The node locations in the figure approximately reflect communication distances between nodes in the network.

Quality of the service path refers to the QoS performance metrics, such as the delay, bandwidth, reliability, etc. Some service paths may satisfy users' QoS requirement and some may not. For example, in Fig. 1, delay of the service path l_2, i.e., (H, I, J, K), may exceed the delay requirement and the path l_1, i.e., (A, D, E, F) is thus more preferable because of its small delay. In the literature, Xu et al. [2] find service paths to optimize the end-to-end resource availability with controlled system overhead. Choi et al. [7] study the least-cost service composition problem for additive QoS metrics. In [8] and [4], multiple QoS criteria are aggregated for service path selection and optimization. The scalable service composition is investigated in [3] and [9] for large scale systems, by employing distributed or hierarchical routing techniques.

Failure recovery is the second critical issue for composed services. In P2P networks, every peer node can potentially be a server or a client. Service components are loosely managed and the composite services more likely fail than in the traditional client-server architecture where dedicated servers are maintained. The reasons of service disruption are due to node leaving (peers exit P2P networks) or node failures. End users may experience service *interference*, if some nodes on the service path exit the system or fail, and the requested service components become unavailable. Therefore, it is necessary to decrease the service interference as much as possible. An effective failure recovery mechanism becomes a crucial factor for ensuring satisfactory composite services to end users. However, so far not much research has been done on failure recovery problems in service composition. Raman et al. [10] present an architecture for quick service path recovery using service replicas and tuning the process of failure detection. Their work mainly focuses on architectural discussion and experimental evaluations.

In this paper, we take a model-based approach to study service interference perceived by end users, service routing, and their relationship. There are two challenges in designing a

service composition system with minimum service interference. First, how can we quantitatively characterize the interference (or the dissatisfaction) to end users? Failure recovery time is a possible metric. However, it is not flexible enough to model users' subjective dissatisfaction. Moreover, it is too dynamic and is determined by many factors such as network conditions, system load, specific implementation of service composition, etc. We need a high-level metric to characterize the service interference, which should be robust and stable, for the purpose of designing routing protocols and algorithms in service composition.

According to this observation, we will model the interference in service composition from a new perspective. A new metric *Interference Intensity* is proposed to characterize the interference experienced by end users in unit time. The measure of interference is derived from *interference functions* that can be flexibly defined according to the relation between the end users' experience and the level of service disruption, i.e., the number of substitutions of service components during a service path recovery. Our new metric closely reflects the impact on the end users caused by service disruption. It is also robust and stable in dynamic P2P networks. Moreover, this metric can be finely tuned to accommodate various types of users (e.g., neutral, patient, and impatient users) and specific applications.

The second challenge is how to find appropriate service paths to minimize the interference. In this paper, we take topology and reliability information on P2P networks as input, provided by link-state protocols, and design optimal routing policies to compute initial service paths and find recoveries if the paths fail.

In P2P networks, different service paths may have different impacts on end users in terms of service interference. There are two major reasons. (1) Different nodes may have different levels of reliability. For instance, some nodes may stay in P2P networks for a long time; other nodes may join and leave more frequently. Intuitively, the service path consisting of more reliable nodes presents less interference to users. For example, if nodes A, D, E, and F are more reliable than other nodes in Fig. 1, the path l_1 is preferred over l_2. (2) If a service path has to satisfy certain QoS requirements, such as delay or bandwidth, not all paths in P2P networks can be used by end users. Thus, some failed service paths can be easily repaired, while some paths may incur larger overhead in recovery. Suppose the delay of service paths is required to be less than d. In Fig. 1, if all nodes have the same reliability and both paths satisfy the delay requirement, the path l_1 is still better than the path l_2. This is because any single-node failure in l_1 can be repaired with only one service component substitution. For instance, if E fails, E can be substituted by B, which provides the same service as E, and other nodes on the path do not change. The delay of the new path just increases a little and can still satisfy the delay requirement. However, if J fails in the path l_2, it can not be simply substituted by E or B, because the delay of the resulted path exceeds the delay requirement d. In this case, multiple node substitution has to be involved in repairing the failure of J.

We call a failure recovery *local recovery*, if it involves only one node substitution, for example, the recovery of E in l_1. On the other hand, if multiple node substitutions have to be involved, we call it *global recovery*, for example, the recovery of J in l_2. A local recovery can be completed quickly. A service path that can be recovered locally is preferred over a path requiring global recovery.

We systematically study the routing and recovery problems in service composition to minimize interference to end users and, at the same time, guarantee the required QoS of the composed service. To our knowledge, this is the first paper in the literature that formally studies the service interference model and applies it to service routing. We will present an optimal solution for this problem and discuss some simplified results in certain special cases.

Moreover, we propose a simplified measure for interference intensity based on one-step lookahead heuristic that allows fast computation, and develop efficient routing and recovery algorithms. The design intuitions behind our heuristics are to take advantage of reliable nodes in P2P networks and to encourage local recovery. We also show that the most reliable service path in traditional concept is not necessarily the optimal choice in terms of decreasing the interference to end users. This observation makes our research different from the conventional most-reliable path routing, such as [11]. Intensive experiments confirm that our proposed routing and recovery algorithms result in much less interference to end users than the existing methods, especially when nodes join and leave the system frequently, the QoS requirement of the service path is stringent, or end users tend to be impatient in service disruption.

The rest of the paper is organized as follows. In Section II, we define network models and a general framework for service path management. In Section III, we propose the metrics for service interference. In Section IV, we define the minimum interference routing problem and present the optimal solution. Furthermore, we discuss the efficient heuristic solutions in Section V. In Section VI, simulation results are presented. Section VII concludes the paper.

II. System Framework

We focus on uni-cast service composition in this paper, i.e., service components are linked in a sequential order and there is only one receiver in a service path. The composed service $\mathcal{S}$ is denoted as $\mathcal{S} = (S_1 \rightarrow S_2 \rightarrow \ldots \rightarrow S_K \rightarrow R)$, where R is the receiver, and K is the number of service components or the number of hops on the service path. S_j stands for the distributed services of type j, which identify a unique function needed by the composed service. In the example of Fig. 1, $S_1, S_2, S_3,$ and S_4 are video server, transcoding, captioning, and translation services, respectively.

A. Service Network Model

In P2P networks, the service S_j may be replicated at multiple nodes, and we denote V_j as the set of nodes that can provide service S_j. For instance, $V_4 = \{F, G, K\}$ in Fig. 1. Specially, $V_{K+1} = R$. *Service Network* is defined as $G_s(V, E)$, where $V = \bigcup_{j=1}^{K+1} V_j$, and $E = \{(v_1, v_2) | v_1 \in V_j, v_2 \in V_{j+1}, 1 \leq j \leq K\}$. Different from general P2P networks, where any two nodes in V can potentially have an overlay link, two nodes in a service network share an overlay link only if they offer two services that are adjacent in the composed service $\mathcal{S}$.

The service path $\mathcal{P}$ in service network G_s for the composed service $\mathcal{S}$ is denoted as $\mathcal{P} = (v_1, v_2, \ldots, v_K, R)$, where $v_j \in V_j$. If QoS of service path is not considered, any simple path in G_s from one of the nodes in V_1 to receiver R is a valid service path. The total number of service paths is $\prod_{j=1}^{K} |V_j|$. For example, Fig. 2 shows the service network according to the service composition example in Fig. 1, in which 72 different service paths exist.

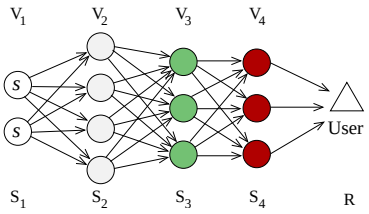


Fig. 2. An example of service network G_s based on the example in Fig. 1.

In service network, some nodes may be unavailable (i.e., failed) to service composition, because some may exit the system, some may have hardware and software errors, or some nodes may not be discovered. In the following analysis, we refer to any type of node unavailability as node failure. We assume every node fails independently of each other and the lifetime of each node follows exponential distribution. Moreover, let r_v denote the failure rate of the node v and $v \in V$. According to this assumption, the failure rate of service path $(v_1, v_2 \ldots, v_K, R)$ is $\sum_{j=1}^{K} r_{v_j}$, if each node only holds one service component. The path with smaller failure rate is more reliable. In Section VI-E, we will discuss the scenarios when the lifetime of nodes is not memoryless.

End users may require some QoS guarantee on the service path. In this paper, we mainly consider additive QoS metric, such as delay and cost. Other types of metrics include multiplicative and concave metrics. The multiplicative metrics, such as reliability, can be transformed into additive metrics by performing logarithm operation. The concave metrics, such as available bandwidth, can be easily handled in our network model and algorithms by deleting the overlay links that do not satisfy the requirement from service networks. In the following sections, we will use delay as an example to explain our framework and algorithms. We denote the computation delay introduced by node v as $d_c(v)$. The network delay introduced by overlay links between nodes v_i and v_j as $d_n(v_i, v_j)$[1]. Thus, the service path $\mathcal{P} = (v_1, v_2, \ldots, v_K, R)$ has delay $d(\mathcal{P}) = \sum_{i=1}^{K-1} [d_c(v_i) + d_n(v_i, v_{i+1})] + d_c(v_K) + d_n(v_K, R)$. As the user requirement, the delay of the service path should be smaller than a given value d^*.

B. Framework of Minimum Interference Service Composition

We demonstrate the framework of our Minimum Interference Service Composition (MISC) system and Minimum Interference Routing (MIR)[2] in Fig. 3. The basic layers include P2P networks and TCP/IP networks, supporting the fundamental communication for service networks. The service discovery layer is in charge of finding peers in P2P networks to provide the needed service components. For each type of service components, a number of replica service nodes from P2P networks are discovered by service discovery protocols, such as [6] or controlled flooding. Let Ψ denote the number of replica nodes that are discovered for each service component (i.e., $|V_j| = \Psi$). Though Ψ is not the focus of this paper, we briefly discuss its impact as follows. A large Ψ means more redundant nodes available for service composition and it is more likely to find a service path that satisfies users' delay requirement even if some nodes fail. However, a large Ψ leads to high system overhead in maintaining service networks and routing. On the other hand, a small Ψ incurs less overhead, but it is possible that no service path can satisfy users' QoS requirement and service composition requests are rejected or are only served for short period of time. In practice, Ψ is chosen as the trade-off between the system overhead and the lifetime of composed services. If some nodes in service network fail, the service discovery is triggered to keep the number of nodes for each service component to be Ψ.

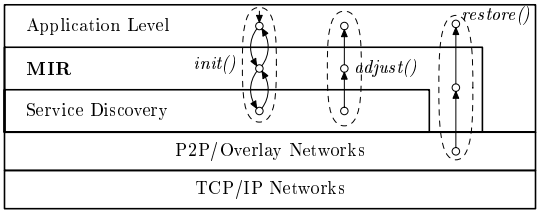


Fig. 3. Framework of Minimum Interference Service Composition (MISC). We focus on Minimum Interference Routing (MIR) layer in this paper.

In service network G_s, node v measures the network delay to node w, where $v \in V_j$ and $w \in V_{j+1}$. Network delay and computation delay information is sent to the receiver or a proxy of the receiver by link-state protocols to construct the information about the service network.

The layer of MIR, in charge of finding and repairing service paths, is the crucial part in designing a low interference service composition system. There are three major processes involved in MIR. (1) $init()$: A service composition request is sent from the application level to MIR, including the delay requirement d^*; MIR triggers service discovery to construct a service network G_s; within G_s, MIR finds a service route, which has delay smaller than d^*, and uses the nodes off the route as backups; finally, the result is returned to application level. (2) $restore()$: if some node in the service path fails, MIR is notified and the service path is repaired by using currently available backup nodes in the service network. In the framework, we do not explicitly keep a backup service path, because backup paths could fail even before the active path. Instead, we maintain the service network G_s that contains replica nodes for each service component, and repair failed service paths with the replica nodes on the fly. (3) $adjust()$: on backup node changes (existing nodes fail or new nodes are discovered), MIR is triggered to update the service network G_s and decide if the current service path needs to be adjusted accordingly.

The optimization objective of MIR is to minimize interfer-

[1]If a node providing multiple types of services appears multiple times in a service network, we have $d_n(v, v) = 0$.

[2]There is a similar but totally different term in MPLS routing literature, which is also called minimum interference routing focusing on the interference between traffic flows. In this paper, we focus on the service interference perceived by end users in P2P networks.

TABLE I
TABLE OF NOTATIONS

K	The number of service components in a service path.
V_j	The set of nodes for service component j, $1 \leq j \leq K$.
$G_s(V, E)$	Service network with node set V and edge set E. $V = \bigcup_{j=1}^{K} V_j$ and $E = \{(v_1, v_2) \mid v_1 \in V_j, v_2 \in V_{j+1}\}$.
$\mathcal{P}$	Service path. $\mathcal{P} = (v_1, v_2, \ldots, v_K, R)$, where $v_j \in V_j$.
Ψ	The number of replica nodes for a service component.
$d_c(v)$	Computation delay incurred at node v. $v \in V$
$d_n(v_i, v_j)$	Network delay from node v_i to node v_j. $v_i, v_j \in V$
d^*	Delay constraint of a service path.
$\mathbb{P}(G_s)$	Set of service paths satisfying delay constraint d^* in G_s.
r_v	Failure rate of node v, $v \in V$.
n_s	Number of node substitutions in a service path recovery.
$n_c(\mathcal{P}, \mathcal{P}')$	Number of node substitutions from service path $\mathcal{P}$ to $\mathcal{P}'$.
$n_r(G_s, v, \mathcal{P})$	Number of node substitutions for repairing v in $\mathcal{P}$ of G_s.
$\mathbf{i}(\cdot)$	Interference function of end users.
$\mathcal{I}$	Summation of interference to end users during service.
$\tilde{\mathcal{I}}$	Interference Intensity to end users during service.
$J(G_s, \mathcal{P})$	Minimum $\mathbf{E}(\mathcal{I})$ with $\mathcal{P}$ as the initial service path in G_s.
$p_1(v)$	Probability that v fails firstly in service network.
ω	Node failure and discovery sequence in service network.
π, Π	Control policy and the set of control policies in MIR.
$\pi_i^*, \pi_r^*, \pi_a^*$	Optimal control policies for $init, restore, adjust$ in MIR.

ence to end users. This objective is achieved by designing appropriate routing control policies in the above three processes. For convenience, we denote Π_i, Π_r and Π_a as the set of policies for $init()$, $restore()$ and $adjust()$, respectively, and $\Pi = \Pi_i \times \Pi_r \times \Pi_a$. The optimal policies for achieving minimum interference are $\pi^* = \{\pi_i^*, \pi_r^*, \pi_a^*\}$.

In this paper, we only focus on how to model service interference perceived by end users and how to find and repair service paths in a given service network with delay and reliability information available. In the MIR layer, there are other auxiliary modules, such as service paths deploying (path signaling) and monitoring, which are out of scope for this paper. Furthermore, for simplicity, we only discuss the case that each node holds one service component, based on which multiple-component case can be extended. We will give brief discussion accordingly.

Table I summaries the notations frequently used in this paper.

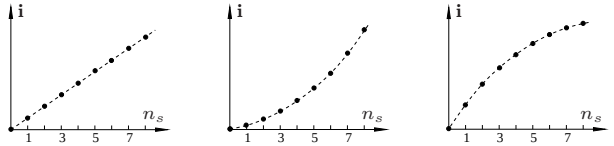
III. MODELING SERVICE INTERFERENCE

To measure the interference to end users is challenging. The interference is about users' dissatisfaction on the composed service. The time percentage of service disruption is one possible measure. However, system failure recovery time is difficult to calculate and it is influenced by many dynamic factors such as system load, network traffic, etc., as well as specific implementations of service composition system. For the purpose of routing design, this type of metrics is too low-level and not robust. Moreover, interference perceived by end users is a subjective measure. The traditional metrics are not flexible enough to accommodate different types of users. For example, some users tend to be more patient to service disruption, while some may be impatient. Thus, we define service *Interference* and *Interference Intensity* based on the frequency of node substitutions in service path recoveries to model the users' dissatisfaction. Notice that the purpose of this metric is not to provide an absolute measure of users' dissatisfaction, but to give a relative value to differentiate the performance of different routing and recovery policies in service composition.

A. Interference Functions

In a service recovery process, the recovery overhead due to a component failure is reflected by the number of service components that has to be substituted in order to repair the service path. Let us denote n_s as the number of component substitutions in a service path recovery. For example, in Fig. 1, suppose node K fails in the service path l_2. If local recovery is viable and node K is substituted by node F, n_s equals one. On the other hand, if global recovery has to be used and l_1 substitutes l_2 completely, n_s equals four. That is, n_s is the number of different components between the new and the original service paths. Compared with recovery time, n_s is a robust measure and can also be easily calculated.

Furthermore, we need a flexible way to map n_s to the service interference perceived by end users. Motivated by the widely applied concept of utility functions in economic theory, we propose to define *Interference Functions* for this purpose. We denote $\mathbf{i}(n_s)$ as the interference to the end users in a service path recovery. $\mathbf{i}(n_s)$ is a scalar function of n_s and is positive and non-decreasing. The specific form of $\mathbf{i}(n_s)$ is mainly determined by the mechanism of setting up service paths and human psychology. Three basic types of interference functions are shown in Fig. 4 as examples. Fig. 4(a) stands for a neutral system in



(a) Linear $\mathbf{i}(n_s)$ – neutral users

(b) Convex $\mathbf{i}(n_s)$ – impatient users

(c) Concave $\mathbf{i}(n_s)$ – patient users

Fig. 4. Three basic types of service Interference Functions.

which the interference grows linearly with n_s, i.e., $\mathbf{i}(n_s) = bn_s$ for a constant b. In this case, the time delay in setting up a service path is proportional to the number of substitutions; the users' impatience is proportional to the recovery overhead. Fig. 4(b) shows a convex interference function in which the users are impatient and their dissatisfaction grows faster as time passes; moreover, the marginal overhead in setting up a longer service path is larger. Fig. 4(c) stands for concave interference. The users are patient in this case and their dissatisfaction grows slower. With respect to setting up a service path, the marginal overhead decreases due to batch processing.

Interference function $\mathbf{i}(n_s)$ can be customized for a specific type of users at run time in MIR, such as the neutral users. In order to address the problem in a broader sense, we do not focus on the detail of constructing $\mathbf{i}(n_s)$ in this paper, but try to optimize the service routing and recovery when an interference

function is given. In addition, when some consecutive service components in the service path are provided by a single node, a part of the recovery overhead incurred by network delay may be avoided. In this case, we can discount the number of substituted components to address this effect.

B. Interference $\mathcal{I}$ and Interference Intensity $\tilde{\mathcal{I}}$

From the beginning of a composed service to the end, the service path may be recovered multiple times. In the k^{th} recovery, the number of component substitution is $n_s(k)$ and the interference is measured as $\mathbf{i}(n_s(k))$. The summation of interferences from all recoveries is denoted as $\mathcal{I}$, and $\mathcal{I} = \sum_{k=1}^{M} \mathbf{i}(n_s(k))$, M is the last recovery before no service path exists or the end user leaves the system. Interference intensity $\tilde{\mathcal{I}}$ is defined as the interference in unit time, i.e.,

$$\tilde{\mathcal{I}} = \frac{\sum_{k=1}^{M} \mathbf{i}(n_s(k))}{\sum_{k=1}^{M} T(k)}, \tag{1}$$

where $T(k)$ is the time distance between $(k-1)^{th}$ recovery and k^{th} recovery. The unit of interference intensity $\tilde{\mathcal{I}}$ is sec^{-1}.

Interference $\mathcal{I}$ and its intensity $\tilde{\mathcal{I}}$ are determined by two major factors. The first factor includes routing polices for setting up and repairing service paths, which is the key part of our minimum interference routing. Let π denote a routing policy and $\pi \in \Pi$. The second one is the dynamic of node joining and leaving the service network, i.e., the failure and discovery *sequence* of nodes, which is the factor we can not control. We denote a sequence of node failure and discovery as ω and the set of all ω's as Ω. For example, in a service network, if A fails, then B fails, and finally C is discovered, we represent this sequence as $\omega = [A^- B^- C^+]$.

In order to evaluate the performance of a control policy π, we need to average $\mathcal{I}$ and $\tilde{\mathcal{I}}$ over all sequences in Ω. The averaged $\mathcal{I}$ and $\tilde{\mathcal{I}}$ for policy π are defined as[3]

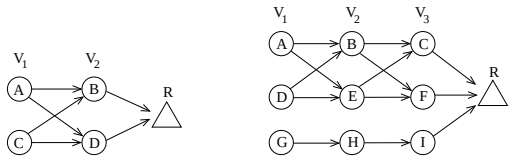
$$\mathbf{E}[\mathcal{I}]_\pi = \mathbf{E}_\omega \left[\mathcal{I}(\pi, \omega)\right] = \sum_{\omega \in \Omega} \left(p_\omega \sum_{k=1}^{M} \mathbf{i}(n_s(k)) \right) \text{ and} \tag{2}$$

$$\mathbf{E}[\tilde{\mathcal{I}}]_\pi = \mathbf{E}_\omega \left[\tilde{\mathcal{I}}(\pi, \omega)\right] = \sum_{\omega \in \Omega} \left(p_\omega \frac{\sum_{k=1}^{M} \mathbf{i}(n_s(k))}{\sum_{k=1}^{M} \mathbf{E}[T(k)]} \right), \tag{3}$$

where p_ω is the probability measure of the sequence ω. In Equations 2 and 3, M, $n_s(k)$ and $T(k)$ are also the functions of ω and π, which is not shown in the Equation for concise presentation. Given ω and π, we can easily compute the number of service component substitutions $n_s(k)$ and the average time between recoveries $\mathbf{E}[T(k)]$. We will use the following example to show the details of calculation.

Example: In Fig. 5(a), every node (except the receiver) has the same failure rate r. There are two types of components (V_1 and V_2) and two nodes exist for each type. All four service paths in the service network satisfy the delay requirement. Due to the symmetric topology, it does not matter which path is used to start the service, and suppose that the initial service path is (A, B, R). We assume that once a node fails, it will not be brought back into the system. If the failed node is on the service path, it is substituted by the other node of the same type if available. Thus, the composed service can be maintained until either V_1 or V_2 becomes empty. Suppose t_j denotes the lifetime of node j. The expected lifetime of the composed service $\mathcal{T}$ in Fig. 5(a) is

$$\mathbf{E}\left[\mathcal{T}\right] = \mathbf{E}\left[\min\left(\max\left(t_A, t_C\right), \max\left(t_B, t_D\right)\right)\right] = \frac{11}{12r}.$$



(a) (b)

Fig. 5. Examples of service composition networks.

There are $4!$ different node failure sequences in the service network of Fig. 5(a). Due to the symmetry, each of sequences happens with probability $1/4!$. We group these sequences into three categories. (1) The service path is repaired twice. For example, if $\omega = [A^- B^- C^- D^-]$, the service path changes from the initial (A, B, R) to (C, B, R), and then to (C, D, R). The service stops when the third node fails. There are 12 sequences that belong to this category. (2) The service path is repaired once, such as $\omega = [A^- C^- B^- D^-]$. When the second node fails, the service stops. 8 sequences are in this category. (3) The service path is not repaired, such as $\omega = [C^- D^- A^- B^-]$. The service stops when the third node fails. Four sequences are in this category. Therefore,

$$\mathbf{E}\left[\mathcal{I}\right] = \frac{1}{4!}\left(12 \times 2\mathbf{i}(1) + 8\mathbf{i}(1) + 0\right) = \frac{4}{3}\mathbf{i}(1), \text{ and} \tag{4}$$

$$\mathbf{E}\left[\tilde{\mathcal{I}}\right] = \frac{1}{4!}\left(12\frac{2\mathbf{i}(1)}{T_3} + 8\frac{\mathbf{i}(1)}{T_2} + 0\right) = \frac{136r}{91}\mathbf{i}(1). \tag{5}$$

where T_2 and T_3 stand for the average time from the beginning to the failure of the second and third nodes, respectively, in a given sequence[4]. As an example, suppose the interfere function to the end users is $\mathbf{i}(1) = 1$ in Fig. 5(a), i.e., the users experience 1 unit of dissatisfaction if the service path fails and it is repaired by using one node substitution. Thus, in the service network we just studied, they are subject to about $1.49r$ unit of dissatisfaction per second in the whole process of service composition, and $1.49r$ components are substituted in unit time.

Please notice that it is easy to compute the interference if the sequence ω of node failure and discovery is known. However, in order to evaluate the performance of a routing control policy over all the possible sequences, the computation cost is intractable, due to the exponential number of sequences to be considered. In practice, we have to use approximation

[3]An alternate definition of $\mathbf{E}\left[\tilde{\mathcal{I}}\right]_\pi$ is $\sum_\omega \frac{\sum_k \mathbf{i}(n_s(k,\omega,\pi))}{\sum_k T(k,\omega,\pi)} p_\omega$. However, the summation in this definition is not converge when T approaches zero. Thus, we use the average time in the definition instead.

[4]Specifically, T_2 is computed based on the joint distribution of the lifetime of the four nodes in a given failure sequence

$$T_2 = \int_0^\infty \int_{t_1}^\infty \int_{t_2}^\infty \int_{t_3}^\infty t_2 \cdot 24 r^4 e^{-r(t_1+t_2+t_3+t_4)}\, \mathrm{d}t_4 \mathrm{d}t_3 \mathrm{d}t_2 \mathrm{d}t_1 = \frac{7}{12r}.$$

Similarly, T_3 can be computed and $T_3 = \frac{13}{12r}$

methods to compute the average interference and its intensity for a control policy, such as Monte Carlo method. For practical routing policies in MIR, a fast evaluation method becomes even more important. In Section V, we will present a simplified measure for this purpose.

IV. Optimal Minimum Interference Routing

In this section, we will describe the Minimum Interference Routing (MIR) problem and present a solution based on dynamic programming to minimize the service interference.

A. Routing Problem Description and Intuitions

In service network G_s, nodes join and leave as time passes. The delay of the composite service path $\mathcal{P}$ is required to be smaller than d^*. The composite service is maintained as long as a service path satisfying the delay requirement d^* exists[5].

From the service composition framework shown in Fig. 3, we can see that the quality of applications is influenced by the underling layers: MIR, service discover, P2P networks and TCP/IP networks. Generally speaking, we have no control over processes of node joining and leaving. The result of service discovery also varies depending on different system implementations, which is out of scope for this paper.

Our focus is to decrease interference to end users by designing appropriate routing and recovery policies in MIR. Specifically, the *Minimum Interference Routing Problem* is to find optimal policy $\pi^* = \{\pi_i^*, \pi_r^*, \pi_a^*\}$ for processes $init()$, $restore()$ and $adjust()$, such that the average interference $\mathbf{E}[\mathcal{I}]_\pi$ (see Equation 2) during the service period is minimized.

Note that MIR does not influence the lifetime of a composite service. The composite service is kept as long as there is a path satisfying the delay requirement. Thus, the lifetime of the composite service is determined by the node joining and leaving processes and service discovery. Since the lifetime is the same whatever routing policies are used, minimizing the average summation of interference $\mathbf{E}[\mathcal{I}]_\pi$ is approximately the same as minimizing the interference intensity $\mathbf{E}[\tilde{\mathcal{I}}]_\pi$.

There are two important intuitions behind designing MIR algorithms. First, we can take advantage of reliable nodes to compose service paths. For instance, in the example of Fig. 5(a), the interference intensity decreases linearly with node failure rate according to Equation 5. This shows that more reliable paths (with less failure rates) may result in less interference intensity and are thus more preferable. The second intuition is to favor service paths that can be recovered locally. The local recovery leads to the fewer number of node substitutions than the global recovery, which means the interference to users is lower.

These two intuitions do not always agree with each other. In some cases, a trade-off has to be made between the two. We will show this fact in the following example.

Example: In the service network shown in Fig. 5(b), let us compare the performance of different service paths. Suppose the communication delay from $\{G, H, I\}$ to the other part of the network is larger than the delay requirement d^*, and the related links are removed from the figure for clarity. All paths shown in the figure satisfy the delay requirement. The recovery policy (Π_r) is to try local recovery first, then try global recovery, and if no path is available, the composite service terminates. The failure rate of node G, H and I is r_1. All other nodes have failure rate r_0 and $r_0 = 0.01$. Once a node fails, it will not be brought back to the system.

There are two options in Π_i policy for choosing an initial service path[6]: $\mathcal{P}_1 = (A, B, C, R)$ and $\mathcal{P}_2 = (G, H, I, R)$. By enumerating all $9!$ node failure sequences[7] and following the same approach as the example in Fig. 5(a), we can calculate the interference intensity when $\mathcal{P}_1$ or $\mathcal{P}_2$ are chosen as the initial service path, respectively. The results, with respect to different values of r_1, are shown in Fig. 6.

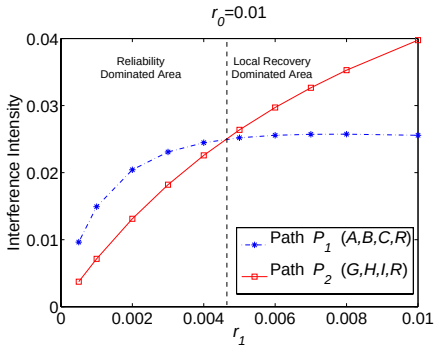


Fig. 6. Interference intensity of the composite service in Fig. 5(b) using $\mathcal{P}_1$ or $\mathcal{P}_2$ as the initial service path. $\mathbf{i}(n_s) = n_s$. r_1 is the failure rate of nodes G, H and I; r_0 is the failure rate of other nodes and $r_0 = 0.01$.

The balance between preferring reliable path and local recovery is clearly demonstrated in Fig. 6. In the service network (Fig. 5(b)), when r_1 is smaller than 0.01, path $\mathcal{P}_2$ (G, H, I, R) is more reliable but it can not be repaired locally; path $\mathcal{P}_1$ (A, B, C, R) has higher failure rate but it can be locally repaired. According to the results in Fig. 6, when r_1 is much smaller than r_0, the reliability of service path is the major factor in MIR. Using path $\mathcal{P}_2$ as the initial path incurs lower interference intensity than using $\mathcal{P}_1$ and thus $\mathcal{P}_2$ is preferred. On the other hand, if r_1 is close to r_0, local recovery dominates reliability factor, and thus $\mathcal{P}_1$ is better than $\mathcal{P}_2$. At a middle point where r_1 is around 0.005, two paths are about the same.

This example also demonstrates that the most reliable service path does not necessarily result in minimum interference. For example, when $r_1 = 0.006$, path $\mathcal{P}_2$ is more reliable than the path $\mathcal{P}_1$, but $\mathcal{P}_2$ still leads to larger interference intensity. Therefore, we should consider the reliability and local recovery jointly in MIR algorithms.

B. Optimal MIR Algorithms

We only use the current information of service networks to make routing decisions. The future information, such as the coming events of node discovery or node leaving, is unavailable and is thus not considered in service routing.

As an important observation, in the MIR problem, an optimal policy π^* for the service composition beginning at time t is also the optimal policy for the same service composition beginning at

[5]We assume that end users use the composite service for very long time.

[6]Other paths are identical to $\mathcal{P}_1$ due to the topology symmetry in Fig. 5(b).

[7]The probability of each failure sequence is given by Lemma 1.

time $t+\Delta t$, where $\Delta t \geq 0$. Otherwise, we can use the policy of the later case (the tail problem) that begins at $t+\Delta t$ to improve the optimal policy of the former case that begins at t. That is, the optimal solution is also optimal in the tail problem. Based on this observation, we have the following optimal solution derived from dynamic programming for the MIR problem.

We denote $J(G_s, \mathcal{P})$ as the minimum average interference $\min_\pi \mathbf{E}[\mathcal{I}]_\pi$ in service network G_s, given that $\mathcal{P}$ is the initial service path and no new nodes are further added into G_s. Moreover, let $\mathbb{P}(G_s)$ denote the set of all routes that satisfy the delay requirement d^* in G_s. Thus, the optimal policy of initialization $init()$ is

$$\pi_i^* : \quad \min_{\mathcal{P}\in\mathbb{P}(G_s)} J(G_s, \mathcal{P}). \tag{6}$$

The basic idea of π_i^* is to set up the initial route that incurs the minimal expected interference given that no new nodes join G_s. The calculation of $J(G_s, \mathcal{P})$ will be shown shortly (Equation 8).

In G_s, suppose the currently used service path is $\mathcal{P}$. When a node fails or a new node is discovered, G_s will be changed into G_s'. The new path $\mathcal{P}'$ to be used in G_s' is obtained by $restore()$ or $adjust()$, and the optimal policies are:

$$\pi_r^*, \pi_a^* : \quad \min_{\mathcal{P}'\in\mathbb{P}(G_s')} \left[\mathbf{i}(n_c(\mathcal{P}, \mathcal{P}')) + J(G_s', \mathcal{P}')\right], \tag{7}$$

where $n_c(\mathcal{P}, \mathcal{P}')$ is the number of node substitutions from the currently used path $\mathcal{P}$ in G_s to path $\mathcal{P}'$ in G_s'. For example, in Fig. 5(b), $n_c((A,B,C,R),(A,E,F,R)) = 2$. π_a^* and π_r^* reconstruct the service path by taking into account the interference due to the path change and the result of the new network.

The calculation of $J(G_s, \mathcal{P})$ is based on dynamic programming from the scenario of network G_s and initial path $\mathcal{P}$ without new nodes discovered. That is,

$$J(G_s, \mathcal{P}) = \sum_{v\in G_s.V} p_1(v) \min_{\mathcal{P}'\in\mathbb{P}(G_s\backslash v)} \left[\mathbf{i}(n_c(\mathcal{P}, \mathcal{P}')) + J(G_s\backslash v, \mathcal{P}')\right]. \tag{8}$$

In Equation 8, $p_1(v)$ is the probability that node v fails first among the nodes in G_s. Under our exponential distribution assumption of node lifetime, we have

$$p_1(v) = \frac{r_v}{\sum_{w\in G_s.V} r_w}, \tag{9}$$

which is supported by the following basic lemma.

Lemma 1: In service network, n nodes are denoted as $v_1, v_2, \ldots, v_n$. The lifetime of each node is exponentially distributed and the rate of v_j is r_j. Then, the probability that node v_l fails first among the n nodes is $p_1(v_l) = r_l / \sum_{j=1}^n r_j$; moreover, the probability of node failure sequence $v_{s_1}, v_{s_2}, \ldots, v_{s_n}$ is $P(v_{s_1}, v_{s_2}, \ldots, v_{s_n}) = \prod_{k=1}^{n-1} (r_{s_k} / \sum_{l=k}^n r_{s_l})$.

Proof: For the first part, let us assume l equals 1 without loss of generality. Because the first failure time T of any node in $\{v_2, \ldots, v_n\}$ follows an exponential distribution with rate $R = \sum_{j=2}^n r_j$, p_{v_1} is the probability that v_1's lifetime T_1 is smaller than T. Thus, $p_1(v_1) = \int_0^\infty \int_{T_1}^\infty R r_1 e^{-(R+r_1)T_1 T} \, \mathrm{d}T \mathrm{d}T_1 = r_1/(R+r_1)$.

For the second part,

$$\begin{aligned} P(v_{s_1}, v_{s_2}, \ldots, v_{s_n}) &= p_1(v_{s_1}) P(v_{s_2}, \ldots, v_{s_n}) \\ &= \frac{r_{s_1}}{\sum_{l=1}^n r_l} P(v_{s_2}, \ldots, v_{s_n}) \\ &= \ldots = \prod_{k=1}^{n-1} \left(r_{s_k} / \sum_{l=k}^n r_{s_l} \right). \end{aligned}$$

■

$J(G_s, \mathcal{P})$ can be calculated in a recursive way using Equation 8. Basically, all failure sequences of nodes in G_s have to be enumerated, which will result in huge amount of computing overhead if the network size is large. In Section V, we will show an approximate method.

In summary, we have the following result.

Lemma 2: The optimal solution of minimum interference routing problem is given by Equations 6, 7 and 8.

C. Optimization Simplification in Special Cases

In some special cases, the MIR problem can be largely simplified. We will show this by the following two lemmas.

The first lemma is about whether a service path needs change proactively, if the nodes on the path do not fail.

Lemma 3: (*Lazy-Adjust Strategy*) If interference function $\mathbf{i}(n_s)$ is linear or concave, in order to achieve minimum interference routing, a service path is changed if and only if some node on the path fails.

Proof: The 'if' part is obvious. We prove the 'only if' part next, i.e., the service path remains the same on the discovery of new nodes and the failures of nodes that are not on the service path. We denote G_s^0 as the original network and the service path is $\mathcal{P}^0$. The series of service networks are $G_s^1, G_s^2, \ldots, G_s^{L-1}$, when the service networks change but no node on $\mathcal{P}^0$ fails. G_s^L is the first service network that a node on $\mathcal{P}^0$ fails.

We consider two strategies for designing routing policies: in strategy A, the service path is adjusted according to Equation 7 whenever the service network changes and we get service paths $\mathcal{P}^l$ in network G_s^l ($1 \leq l \leq L$); in strategy B, service path is lazily adjusted, that is, it is recomputed only in G_s^L and the network changes in the middle are ignored. Let us compare the service interference involved in these two strategies:

$$\begin{aligned} J_A &= \min_{\mathcal{P}'\in\mathbb{P}(G_s^L)} \left[\sum_{j=1}^{L-1} \mathbf{i}(n_c(\mathcal{P}^{j-1}, \mathcal{P}^j)) + \mathbf{i}(n_c(\mathcal{P}^{L-1}, \mathcal{P}')) \right. \\ &\quad \left. + J(G_s^L, \mathcal{P}') \right] \equiv \min_{\mathcal{P}'\in\mathbb{P}(G_s^L)} \left[f_A(\mathcal{P}') \right], \text{ and} \\ J_B &= \min_{\mathcal{P}'\in\mathbb{P}(G_s^L)} \left[\mathbf{i}(n_c(\mathcal{P}^0, \mathcal{P}')) + J(G_s^L, \mathcal{P}') \right] \\ &\equiv \min_{\mathcal{P}'\in\mathbb{P}(G_s^L)} \left[f_B(\mathcal{P}') \right]. \end{aligned}$$

Because $\mathbf{i}(\cdot)$ is a linear or concave function and

$$n_c(\mathcal{P}^0, \mathcal{P}') \leq \sum_{j=1}^{L-1} n_c(\mathcal{P}^{j-1}, \mathcal{P}^j) + n_c(\mathcal{P}^{L-1}, \mathcal{P}'),$$

we have $f_B(\mathcal{P}') \leq f_A(\mathcal{P}')$. Thus, $J_B \leq J_A$. Since strategy B is better than A, we proved this lemma. ■

By the lazy-adjust strategy, the unnecessary computation in MIR can be avoided. Specifically, when the interference

function is linear or concave, MIR will not be triggered to adjust the service path until the currently used service path fails and Π_a control policy is thus not needed.

Lemma 4: In service network $G_s(V, E)$, where $V = \cup_{j=1}^{K+1} V_j$, if the delay requirement d^* is large enough, MIR problem is equivalent to the problem of the most reliable path routing, i.e., $J(G_s, \mathcal{P}^*) \leq J(G_s, \mathcal{P})$, where $\mathcal{P}^*$ is path $(v_1, v_2, \ldots, v_K, R)$ satisfying $v_l = \arg\min_{v \in V_l} r_v$; $\mathcal{P}$ is any other service path.

Proof: Because the delay requirement d^* is large enough, the nodes of different service components can be selected independently in service routing and recovery. Thus, we only need to show that, within nodes of one service component, choosing the most reliable node leads to the least interference. We will use induction to prove this fact as follows.

Without loss of generality, let's focus on the first service component V_1 and suppose $V_1 = \{v_1, \ldots, v_L\}$, which means the nodes in V_1 can provide the same service. The failure rate of v_l is r_l. Suppose $r_1 \leq r_2 \ldots \leq r_L$, without loss of generality.

When $L = 2$, the expected service interference is $J(V_1, v_1) = \frac{r_1}{r_1+r_2}\mathbf{i}(1)$, if v_1 is chosen; similarly, if v_2 is chosen, $J(V_1, v_2) = \frac{r_2}{r_1+r_2}\mathbf{i}(1)$. Because $r_1 \leq r_2$, $J(V_1, v_1) \leq J(V_1, v_2)$. Next, let us assume that when $L \leq n$, we have $J(V_1, v_1) \leq J(V_1, v_l)$ for any $l \leq L$. When $L = n + 1$, we will prove that $J(V_1, v_1) \leq J(V_1, v_l)$, for any $1 < l \leq L$. Let R denote $\sum_{j=1}^{L} r_j$. Based on Equation 8 and the induction assumption, we have

$$J(V_1, v_1) = \frac{r_1}{R}\left[\mathbf{i}(1) + J(V_1 \backslash v_1, v_2)\right] + \sum_{j=2}^{L} \frac{r_j}{R} J(V_1 \backslash v_j, v_1) \text{ and}$$

$$J(V_1, v_l) = \frac{r_l}{R}\left[\mathbf{i}(1) + J(V_1 \backslash v_l, v_1)\right] + \sum_{\substack{1 \leq j \leq L \\ j \neq l}} \frac{r_j}{R} J(V_1 \backslash v_j, v_l).$$

Therefore,

$$\begin{aligned} J(V_1, v_1) - J(V_1, v_l) &= \frac{r_1 - r_l}{R}\mathbf{i}(1) \\ &+ \frac{r_1}{R}\left(J(V_1 \backslash v_1, v_2) - J(V_1 \backslash v_1, v_l)\right) \\ &+ \sum_{j \neq 1, l} \frac{r_j}{R}\left(J(V_1 \backslash v_j, v_1) - J(V_1 \backslash v_j, v_l)\right) \\ &\leq 0. \end{aligned}$$

The last inequality is due to the induction assumption and $r_1 \leq r_l$. Finally, we proved this lemma. ∎

The intuition of Lemma 4 is that we choose the node substitution sequence according to the failure sequence that least likely happens. It is easy to obtain from Lemma 1 that $P(v_{s_1}, v_{s_2}, \ldots, v_{s_n})$ is minimized when $r_{s_1} \leq r_{s_2} \ldots \leq r_{s_n}$, which leads to fact that the maximum reliability routing is equivalent to the minimum interference routing under the condition that the delay requirement d^* is relaxed.

V. Heuristic Algorithms in MIR

Two major computation overhead exist in the optimal MIR solution in Section IV: calculating $J(G_s, \mathcal{P})$ and searching $\mathbb{P}(G_s)$. The first one requires to enumerate all node failure sequences and the second one needs to search all paths in service network. It is not surprising to see that, in order to obtain the exact solutions, the computation complexity is an exponential function of the network size. In this section, we propose some heuristics within the framework of the optimal MIR algorithms to solve the problem efficiently with satisfying performance.

A. One-step Lookahead Estimation $\hat{J}(G_s, \mathcal{P})$

$J(G_s, \mathcal{P})$ is the minimum average interference in service network G_s with $\mathcal{P}$ as the initial path. In order to avoid the heavyweight computation of $J(G_s, \mathcal{P})$, we define an alternate metric $\hat{J}(G_s, \mathcal{P})$, the interference intensity $\tilde{\mathcal{I}}$ measured in the time period until the first failure of $\mathcal{P}$ happens. Specifically,

$$\hat{J}(G_s, \mathcal{P}) = \sum_{v \in \mathcal{P}} p_1(v)\mathbf{i}(n_r(G_s, v, \mathcal{P}))\frac{1}{\mathbf{E}[T_1(v)]}, \tag{10}$$

where $n_r(G_s, v, \mathcal{P})$ is the number of node substitutions resulted by repairing node v in path $\mathcal{P}$ of network G_s. $\mathbf{E}[T_1(v)]$ is the expected time in which node v firstly fails.

$$\begin{aligned} \mathbf{E}[T_1(v)] &= \mathbf{E}[T_v | T_v \leq T_j, j \in \mathcal{P}] \\ &= \int_0^\infty \int_{t_v}^\infty t_v r_v R e^{-r_v t_v - Rt}\, \mathrm{d}t \mathrm{d}t_v / p_1(v) \\ &= 1/\sum_{j \in \mathcal{P}} r_j, \end{aligned}$$

where $R = \sum_{j \neq v, j \in \mathcal{P}} r_j$ and T_v is the lifetime of node v. By inserting this result into Equation 10, we have

$$\hat{J}(G_s, \mathcal{P}) = \sum_{v \in \mathcal{P}} r_v \mathbf{i}(n_r(G_s, v, \mathcal{P})). \tag{11}$$

Intuitively, $\hat{J}$ estimates interference intensity by considering all possible first failures in the service path. It is also called *one-step lookahead estimation* of interference intensity.

To precisely calculate $n_r(G_s, v, \mathcal{P})$, we need to know the exact recovery result for the failure of v. However, this information is difficult to get, because the recovery policy itself recursively depends on $\hat{J}$. Thus, we assign $n_r(G_s, v, \mathcal{P})$ coarsely-grained values, depending on if the local recovery is viable. If $\mathcal{P}$ can be locally recovered, n_r equals 1; otherwise, it is αK, where K is the number of service components in the service path[8]. For instance, in the example of Section IV-A and Fig. 5(b), $\hat{J}(G_s, (A, B, C, R)) = 3r_0\mathbf{i}(1)$ and $\hat{J}(G_s, (G, H, I, R)) = 3r_1\mathbf{i}(3\alpha)$. α is a number between 0 and 1. In this paper, we find that $[0.4, 0.6]$ is a good range for α in most cases, i.e., we assume that in a global recovery, about half of the components are substituted.

Based on $\hat{J}$, the optimal initialization policy is the same as in previous section, i.e., finding the path with the minimal estimated interference intensity

$$\pi_i^* : \quad \min_{\mathcal{P} \in \mathbb{P}(G_s)} \hat{J}(G_s, \mathcal{P}). \tag{12}$$

The recovery and adjust policy have to be reformed by considering the interference intensity caused by the current path change which is $\mathbf{i}\left(n_c(\mathcal{P}, \mathcal{P}')\right)/\mathbf{E}[T_1(\cdot)]$. Thus, we have:

$$\pi_r^*, \pi_a^* : \quad \min_{\mathcal{P}' \in \mathbb{P}(G_s')} \left[\mathbf{i}\left(n_c(\mathcal{P}, \mathcal{P}')\right) \sum_{j \in \mathcal{P}'} r_j + \hat{J}(G_s', \mathcal{P}')\right]. \tag{13}$$

[8]There are other treatments of n_r. For example, we can separate the cases in which n_r equals 2 to make the granularity of $n_r(G_s, v, \mathcal{P})$ finer. We can also define n_r as the minimum number of substitutions to repair $\mathcal{P}$ in G_s. In this paper, we only take the simplest approach to differentiate local recovery and global recovery

In summary, the time complexity in computing $\hat{J}$ is $O(K\Psi)$, where K is the number of components in the service path and Ψ is the number of redundant nodes for a component. Different from the traditional reliability or delay metrics, $\hat{J}$ combines the concepts of node reliability and local recovery together.

B. Routing Heuristics in MIR

Besides estimating interference intensity, we still need to design efficient path selection and recovery algorithms in MIR. A direct enumeration approach by searching through $\mathbb{P}(G_s)$ has time complexity $O(\Psi^K)$, which causes huge amount of overhead in large service networks. In this section, we focus on the heuristics to select service paths efficiently.

1) Service Path Initialization: Based on Equation 12, the service path initialization problem is a delay constrained and least $\hat{J}$ routing problem in a service network. In general, this is a hard problem due to two independent routing metrics involved (delay and interference metric). As a special case, when each node can be recovered locally, the objective $\hat{J}(G_s, \mathcal{P})$ is simplified as $\mathbf{i}(1)\sum_{j\in\mathcal{P}} r_j$, which makes this problem closely related to the traditional problem of *Delay Constrained and Least Cost routing* (***DCLC*** problem) shown to be NP-hard in [12]. Specifically, by adding an auxiliary node v_a, connecting v_a to all nodes in $G_s.V_1$, and letting the failure rate and delay related to v_a to be zero, the special case of service path initialization problem is then equivalent to the DCLC problem in the obtained network from v_a to the receiver.

Moreover, in calculating $\hat{J}(G_s, \mathcal{P})$, we have to decide if each node in $\mathcal{P}$ can be recovered locally if it fails, which makes the cost contributed by a node to $\hat{J}$ depend on other nodes in the path. This fact invalidates the Markovian property assumed by most routing algorithms and thus the problem becomes even more difficult to solve. For example, in the service network shown in Figure 5(b), the delay requirement is 4. Suppose all links have delay 1 except $d_n(D, E) = 3$, and all computation delay is zero. In service path (A, B, C, R), the cost contributed by B to $\hat{J}$ is $r_B\mathbf{i}(1)$, because B can be locally repaired with E. On the other hand, in service path (D, B, F, R), if B fails, it can not just be substituted by E due to the violation of delay requirement and thus the cost contributed by B is $r_B\mathbf{i}(3)$. In the above two service paths, B incurs different costs to $\hat{J}$. While, in terms of traditional routing metrics, a node usually contributes the same amount of cost in all paths.

From the example in Section IV-A (Fig. 6), two factors are important for solving the problem: node reliability and local recovery. Based on this intuition, we propose a heuristic for the service path initialization problem by solving a sequence of DCLC problems. The basic idea is to first find the most reliable path satisfying the delay requirement and then check if each node can be locally recovered. After removing some nodes that obstruct local recovery from the service network, we iterate the previous process until satisfying service paths are found. The nodes that have the largest delay on the service path are the candidates to be removed. For convenience, we denote $D(v, \mathcal{P})$ as the delay contributed by node v in path $\mathcal{P}$, which consists of the computation delay of v and the network delay between v and its previous and next nodes in $\mathcal{P}$.

The detail algorithm is described in Algorithm 1. θ is in $(0, 1)$ to control the percentage of nodes that can not be locally recovered. In this paper, we choose θ from $[0, \frac{2}{K}]$. We use the Lagrange relaxation method in [13] to solve the DCLC problem. The algorithm terminates in at most $(K-1)\Psi$ iterations. In each iteration, solving DCLC problems dominates other computation. In total, the time complexity is $K^2\Psi^3\log^3(K\Psi^2)$, if the method in [13] is applied. In practice, K and Ψ are not large numbers.

Algorithm 1: Heuristic Algorithm for Delay Constrained and Least $\hat{J}$ Routing

input : Service Network G_s, Receiver v_r, Delay Constraint d^*
output: Service Path $\mathcal{P}_0$

$G'_s \leftarrow G_s$;
$\mathcal{P}_0 \leftarrow null$;
$n_l \leftarrow K$; *//K – the length of the service path*
$\mathcal{P} \leftarrow \texttt{DCLC}(G'_s, v_r, d^*)$;
while $\frac{n_l}{K} > \theta$ *and* $\mathcal{P} \neq null$ **do**
  if $\mathcal{P}_0 = null$ *or* $\hat{J}(G_s, \mathcal{P}_0) > \hat{J}(G_s, \mathcal{P})$ **then**
    $\mathcal{P}_0 \leftarrow \mathcal{P}$;
  end
  $V_d \leftarrow \{v \in \mathcal{P} | v$ has backup nodes in $G'_s\}$;
  if $V_d = \emptyset$ **then** break;
  $v_d \leftarrow \arg\max_{v\in V_d} D(v, \mathcal{P})$;
  delete v_d from G'_s;
  $\mathcal{P} \leftarrow \texttt{DCLC}(G'_s, v_r, d^*)$;
  $n_l \leftarrow$ the number of nodes in $\mathcal{P}$ which can not be locally repaired in G'_s;
end

2) Service Path Recovery: When the service path $\mathcal{P}$ fails, we need to find a new path $\mathcal{P}'$ in the updated service network G'_s. According to Equation 13, the objective of service path recovery is to minimize the following formula by finding an appropriate $\mathcal{P}'$ from $\mathbb{P}(G'_s)$:

$$\hat{J}_r(G'_s, \mathcal{P}, \mathcal{P}') \equiv \sum_{v\in\mathcal{P}'} r_v[\underbrace{\mathbf{i}\left(n_c(\mathcal{P}, \mathcal{P}')\right)}_{\text{(a)}} + \underbrace{\mathbf{i}\left(n_r(G'_s, v, \mathcal{P}')\right)}_{\text{(b)}}]. \quad (14)$$

Our basic idea is as follows. We first assume that part (b) is $\mathbf{i}(1)$, i.e, all nodes in the new path $\mathcal{P}'$ can be locally repaired, and find routes only by part (a). Then, we perform iterative optimization by removing the nodes that obstruct local recovery.

Especially, if $\mathbf{i}(\cdot)$ is linear,

$$\begin{aligned}\hat{J}_r(G'_s, \mathcal{P}, \mathcal{P}') &\simeq \sum_{v\in\mathcal{P}'} r_v\left[n_c(\mathcal{P}, \mathcal{P}')\mathbf{i}(1) + \mathbf{i}(1)\right] \\ &\simeq \sum_{\substack{v\in\mathcal{P}'\\ v\notin\mathcal{P}}} r_v\mathbf{i}(1)(K+1) + \sum_{\substack{v\in\mathcal{P}'\\ v\in\mathcal{P}}} r_v\mathbf{i}(1). \quad (15)\end{aligned}$$

The last formula is obtained by using the failure rates of the nodes off $\mathcal{P}$ to approximate the rates of nodes on $\mathcal{P}$. This is equivalent to scaling the failure rate of nodes off $\mathcal{P}$ by $(K+1)$, which makes the nodes on $\mathcal{P}$ more preferable to be used in the new path, and then carrying on the DCLC routing.

The detailed process is described in Algorithm 2. Local recovery is tried first, and then we apply the method similar to Algorithm 1 but $\hat{J}_r$ is the optimization objective. The node failure rates are adjusted to favor the nodes on the old service path $\mathcal{P}$ according to Equation 15.

Algorithm 2: Heuristic Algorithm for Service Path Recovery

input : Updated Service Network G_s, Receiver v_r, Delay Constraint d^*, Service Path $\mathcal{P}$ with the failed node v_f
output: New Service Path $\mathcal{P}_n$

$\mathcal{P}_n \leftarrow null$;
// Try to recover v_f locally.
forall *node* v *in* G_s *that can substitute* v_f *and satisfy the delay constraint* d^* **do**
  $\mathcal{P}' \leftarrow \mathcal{P} \backslash v_f + v$; // *substitute* v_f *with* v *in* $\mathcal{P}$.
  if $\mathcal{P}_n = null$ *or* $\hat{J}_r(G_s, \mathcal{P}, \mathcal{P}') < \hat{J}_r(G_s, \mathcal{P}, \mathcal{P}_n)$ **then**
    $\mathcal{P}_n \leftarrow \mathcal{P}'$;
  end
end
if $\mathcal{P}_n \neq null$ **then** return;
// Try to recover v_f globally.
$G'_s \leftarrow G_s$;
forall $v \in \mathcal{P} \bigcap G'_s$ **do** $G'_s.r_v \leftarrow G'_s.r_v/(K+1)$;
$n_l \leftarrow K$;
$\mathcal{P}' \leftarrow \texttt{DCLC}(G'_s, v_r, d^*)$;
while $\frac{n_l}{K} > \theta$ *and* $\mathcal{P}' \neq null$ **do**
  if $\mathcal{P}_n = null$ *or* $\hat{J}_r(G_s, \mathcal{P}, \mathcal{P}') < \hat{J}_r(G_s, \mathcal{P}, \mathcal{P}_n)$ **then**
    $\mathcal{P}_n \leftarrow \mathcal{P}'$;
  end
  $V_d \leftarrow \{v \in \mathcal{P}' | v$ has backup nodes in $G'_s\}$;
  if $V_d = \emptyset$ **then** break;
  $v_d \leftarrow \arg\max_{v \in V_d} D(v, \mathcal{P}')$;
  delete v_d from G'_s;
  $\mathcal{P}' \leftarrow \texttt{DCLC}(G'_s, v_r, d^*)$;
  $n_l \leftarrow$ the number of nodes in $\mathcal{P}'$ which can not be locally repaired in G'_s;
end

VI. Simulations and Discussions

In this section, we validate our MIR heuristic algorithms and their performance by comparing them with the traditional routing based on reliability and delay QoS metrics. The heuristic algorithms presented in Section V are implemented. For convenience, we refer to the optimal routing policy using the one-step lookahead estimation (i.e., Equations 12 and 13) as ***HMIR-I***; the routing policy based on Algorithms 1 and 2 as ***HMIR-II***. The routing policies that are used for comparison in our simulations are: (1) ***minDelay*** – the path with the minimum delay is used as the service path; (2) ***maxReliability*** – the most reliable path is chosen in service composition. For fairness, in the process of service path recovery, all four policies try to recover the failed path locally and then invoke the global path recovery. Please notice that HMIR-I and maxReliability use the exhaustive path enumeration to find the best path according to their own optimization objectives. HMIR-II uses the method based on Lagrange relaxation to solve a sequence of DCLC problems; minDelay uses method based on Dijkstra's shortest path routing algorithm.

Our simulation is based on two IP networks: *WM40* and *WM80* that have 40 and 80 nodes, respectively. The network topologies are generated by using BRITE topology generator [14] under Waxman model and the ratio between the number of edges to the number of nodes is two. The link delay and computation delay are randomly generated from $[0.01, 0.1]$ seconds. At the IP level, all routes between nodes are determined by shortest hop-count routing. The failure rates of nodes are taken from $[0.01, 0.04]$ randomly; the node recovery rate is from $[0.02, 0.08]$. Only the alive nodes can be used to in the composite services.

We do not simulate the application layer, and mainly focus on the performance of the MIR layer. Every node in networks is randomly assigned a service type from $1, 2, \ldots, K$. A service path can be composed by sequentially connecting the nodes, one from each service type, and finally connecting to the receiver. In our experiment, the service discovery process scans the network randomly, until the required service components are found. That is, all unknown nodes are discovered with the same probability if they are needed.

Our major purpose of the experiment is to test the performance of various service routing policies, with respective to service interference intensity. For any sequence of node failures and recoveries, all four policies (HMIR-I, HMIR-II, minDelay, and maxReliability) are executed in parallel and independently, the incurred service interference and the lifetime of composite services are recorded, and final the interference intensity $\tilde{\mathcal{I}}$ is computed according to Equation 1. In WM40, every node has a chance to be a receiver; in WM80, 40 nodes are selected randomly to be receivers. For each receiver, 150 iterations are simulated. In each iteration, the simulation stops in 3600 seconds or until no service path satisfies the delay constraint. The average interference intensity is calculated based on all iterations and all receivers. The result is used as the metric to evaluate the performance of routing in service composition regarding to service interference.

In the following text, we will discuss five scenarios to understand the performance and the applicability of our proposed algorithms. Please notice that all values of interference intensity shown as follows are average values.

A. Impact of Delay Requirement

We first study the interference intensity incurred by different policies with respect to various delay requirements in service composition. The results are shown in Fig. 7. In network WM40, all nodes participate in the service network and $\mathbf{i}(n_s) = n_s$. It is not surprising to see that HMIR-I and HMIR-II, which use service interference as the minimization objective, perform better than the other two policies, especially when the delay requirement d^* is relatively small. When d^* is large, the performance of maxReliability catches up. This also confirms our Lemma 4, i.e., the most reliable path routing is the optimal routing policy when d^* is large enough. Thus, our minimum interference routing algorithms are especially useful when the delay requirement is not very large.

Moreover, HMIR-I that searches paths exhaustively just performs a little better than HMIR-II. This result demonstrates that our heuristic algorithms based on solving DCLC problems can achieve satisfying performance but is much more efficient than the exhaustive search in large service networks.

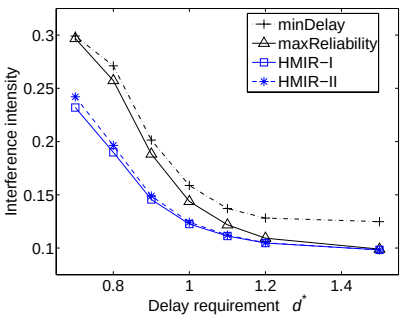


Fig. 7. Average $\tilde{\mathcal{I}}$ in Network WM40 (K=6). $\mathbf{i}(n_s) = n_s$.

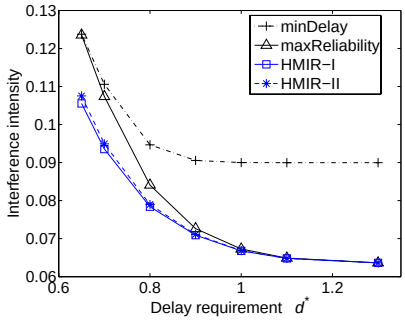


Fig. 8. Average $\tilde{\mathcal{I}}$ in Network WM80 (K=4, $\Psi = 5$). $\mathbf{i}(n_s) = n_s$.

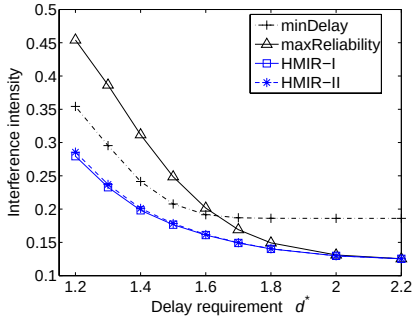


Fig. 9. Average $\tilde{\mathcal{I}}$ in Network WM80 (K=8, $\Psi = 5$). $\mathbf{i}(n_s) = n_s$.

B. Impact of Length of Service Path

We demonstrate the impact of the length of service path (K) is Fig. 8 and Fig. 9. In network WM80, two types of service paths are tested, which have 4 and 8 components on the paths, respectively. The improvement of our MIR heuristic algorithms over other policies is more prominent when the service path is longer (in Fig. 9). If a composite service uses more components, the service path more likely fails because of more unreliable components. Thus, the interference intensity is higher. It is crucial to arrange the path recovery appropriately for long service paths, such that the interference can be decreased.

It also interesting to notice that when the service path is long, the delay of the service path plays an important role for decreasing inteference. In Fig. 9, the performance of minDelay actually exceeds maxReliability, when d^* is small. While, HMIR-I and HMIR-II always performs the best among the four.

C. Impact of System Dynamics

Intuitively, if service networks are more dynamic, i.e., nodes join and leave the networks more frequently, service interference intensity increases and our MIR policies should perform much better than other two routing policies which do not explicitly encourage local recovery. These facts are confirmed in Fig. 10 of the simulation in network WM80. As the average failure rate of node increases, the performance gaps between different routing polices become larger.

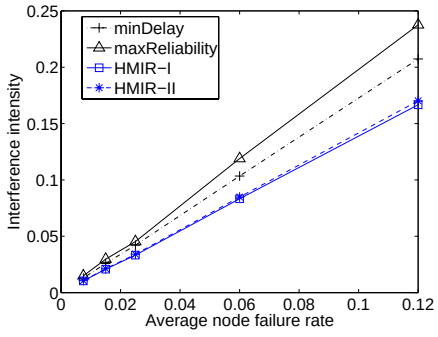


Fig. 10. Average $\tilde{\mathcal{I}}$ in WM80. $\mathbf{i}(n_s) = n_s/K$, $K = 6$, $d^* = 0.9$ and $\Psi = 5$.

D. Impact of $\mathbf{i}(\cdot)$ function

The interference function $\mathbf{i}(n_s)$ may take different forms and the shapes of the functions can influence the performance of MIR. Fig. 11 shows various interference functions and the related service interference intensity of the four routing policies.

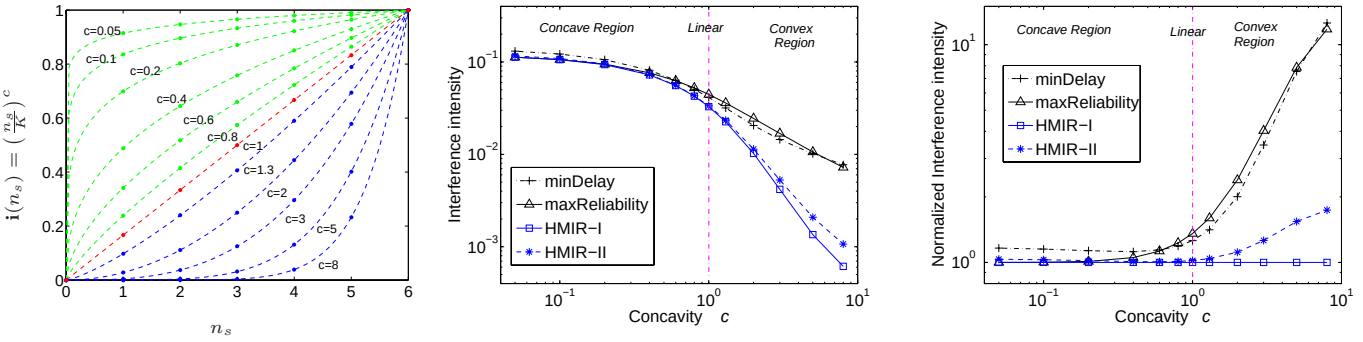
In network WM80, we tested several interference functions in the form of $\mathbf{i}(n_s) = \left(\frac{n_s}{K}\right)^c$, where service path length K equals 6 and c is a positive number. These functions are drawn in Fig. 11(a). c means the concavity factor of the interference function. When $c \in (0, 1)$, we have concave curves; when $c > 1$, $\mathbf{i}(n_s)$ is convex; if $c = 1$, the interference function is linear. According to these functions, the performance of the four routing policies is pictured in Fig. 11(b). In the region of convex interference functions (i.e., end users are of impatient type), HMIR-I and HMIR-II perform much better than the other two polices, when compared with the results in the concave region. We can see this more clearly in Fig. 11(c), where the normalized interference intensity is displayed. Specifically, the $\tilde{\mathcal{I}}$'s from minDelay, maxReliability and HMIR-II are divided by the result from HMIR-I.

We explain the influence of the shapes of interference functions as follows. If convex interference functions are used, local recovery incurs much less interference penalty than global recovery due to the convex shape. Our MIR heuristic algorithms aggressively encourage local recovery and thus obtains much less $\tilde{\mathcal{I}}$ than the other policies. On the other hand, in the concave region, the benefits of local recovery are not significant especially for small c, and the advantage of HMIR-I and HMIR-II is not very prominent. However, we still believe there is room for improvement for small c and this will be our future research.

E. Other Distributions of Node Lifetime

In this paper, our theory and algorithms are developed based on an assumption that the events of node failures and recoveries follow Poisson process, i.e., the lifetime and the recovery time of a node are independently and exponentially distributed. However, because our MIR heuristic algorithms (HMIR-I and HMIR-II) mainly take advantages of local recovery, they can still be robustly applied into the scenarios of non-exponential distributions with satisfactory performance.

We tested the four polices in two other distributions: uniform and Weibull distributions. In the uniform distribution, the node lifetime is randomly generated from interval $[\frac{1}{r_v}(1-a), \frac{1}{r_v}(1+a)]$, where r_v is failure rate of node v and a is a number between 0 and 1. In Weibull distribution, the node lifetime is generated according to CDF, $F(t) = 1 - e^{r_v t^a}$, where a is the shape parameter and $a > 0$. When $a = 1$, it becomes the exponential



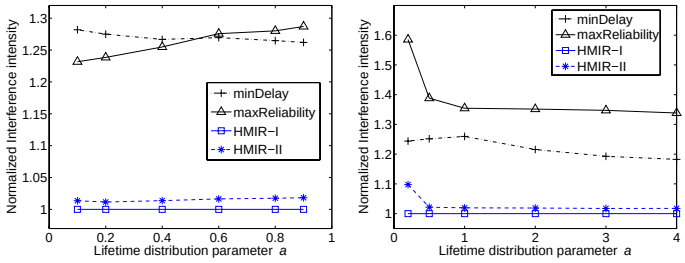
(a) Shapes of $\mathbf{i}(n_s)$.

(b) Average $\tilde{\mathcal{I}}$ in WM80. $K = 6$, $d^* = 0.9$, and $\Psi = 5$.

(c) Normalized $\tilde{\mathcal{I}}$ in WM80. $K = 6$, $d^* = 0.9$, and $\Psi = 5$. HMIR-I is the normalization base.

Fig. 11. Impact of the shapes of the interference function $\mathbf{i}(n_s)$.

distribution. The normalized interference intensity of these four routing policies is shown in Fig. 12. Again, we use the results from HMIR-I to normalize the results of the other three polices.



(a) Node lifetime follows uniform distribution.

(b) Node lifetime follows Weibull distribution.

Fig. 12. Performance of MIR in non-exponential distributions of node lifetime. Normalized $\tilde{\mathcal{I}}$ in WM80. $K = 6$, $d^* = 0.9$, and $\Psi = 5$.

From the figure, we can conclude that HMIR-I and HMIR-II have similar performance; other two polices is at least 20% worse than them.

VII. Conclusion

In this paper, we have systematically studied the interference to end users in service composition and designed the routing and recovery polices for decreasing this interference. We take model-based approach to quantify the interference and develop optimal and heuristic solutions for the minimum interference service composition problem based on the dynamic programming optimization framework. Our model and algorithms can achieve much less interference to end users than the traditional methods, especially in the scenarios of stringent QoS requirement, highly dynamic networks, or the type of impatient users.

In the future work, we will further improve the performance of our heuristics for MIR. Especially, we will consider to deploy our policies in a distributed fashion. The adjustment policy Π_a in MIR is not studied in great detail in this paper and we are going to investigate its applicability to convex interference functions in the future.

References

[1] S. D. Gribble, M. Welsh, J. R. von Behren, E. A. Brewer, D. E. Culler, N. Borisov, S. E. Czerwinski, R. Gummadi, J. R. Hill, A. D. Joseph, R. H. Katz, Z. M. Mao, S. Ross, and B. Y. Zhao, "The ninja architecture for robust internet-scale systems and services," *Computer Networks*, vol. 35, no. 4, pp. 473–497, 2001.

[2] D. Xu and K. Nahrstedt, "Finding service paths in a media service proxy network," in *Proceedings of SPIE/ACM Multimedia Computing and Networking Conference*, 2002.

[3] J. Jin and K. Nahrstedt, "On exploring performance optimizations in web service composition," in *Proceedings of IEEE International Symposium on High Performance Distributed Computing*, 2004.

[4] Y. Liu, A. H. Ngu, and L. Zeng, "QoS computation and policing in dynamic web service selection," in *Proceedings of the International Conference on World Wide Web*, 2004.

[5] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 17–32, February 2003.

[6] D. Xu, K. Nahrstedt, and D. Wichadakul, "QoS-aware discovery of wide-area distributed services," in *Proceedings of IEEE/ACM International Symposium on Cluster Computing and the Grid*, 2001.

[7] S. Y. Choi, J. Turner, and T. Wolf, "Configuring sessions in programmable networks," in *Proceedings of IEEE INFOCOM*, 2001.

[8] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng, "Quality driven web services composition," in *Proceedings of the International Conference on World Wide Web*, 2003.

[9] X. Gu and K. Nahrstedt, "A scalable QoS-aware service aggregation model for peer-to-peer computing grids," in *Proceedings of IEEE International Symposium on High Performance Distributed Computing*, 2002.

[10] B. Raman and R. H. Katz, "An architecture for highly available wide-area service composition," *Computer Communications Journal*, vol. 26, no. 15, pp. 1727–1740, 2003.

[11] G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, A. Orda, T. Przygienda, and S. Systems, *QoS Routing Mechanisms and OSPF Extensions. RFC 2676.* Network Working Group, August 1999.

[12] M. Garey and D. Johnson, *Computers and intractability: a guide to the theory of NP-completeness.* New York: W.H. Freeman, 1979.

[13] A. Jüttner, B. Szviatovszki, I. Mécs, and Z. Rajkó, "Lagrange relaxation based method for the QoS routing problem," in *Proceedings of IEEE INFOCOM*, 2001.

[14] A. Medina, A. Lakhina, I. Matta, and J. Byers, "Boston university representative internet topology generator," in *http://cs-www.bu.edu/brite/*.