Modeling and Simulation of Service Composition in Opportunistic Networks

Umair Sadiq, Mohan Kumar
University of Texas at Arlington
Arlington, TX, USA
umair.sadiq@mavs.uta.edu,
mkumar@uta.edu

Andrea Passarella, Marco Conti
IIT-CNR, Pisa, Italy
{a.passarella,m.conti}@iit.cnr.it

ABSTRACT

Pervasive networks formed by users' mobile devices have the potential to exploit a rich set of distributed service components that can be composed to provide each user with a multitude of application level services. However, mobile and pervasive networks suffer from intermittent connectivity, disconnections and partitions, such that opportunistic networking techniques are required to enable communication. This poses novel challenges to service composition techniques. While several works have discussed middleware and architecture for service composition in well-connected wired networks and in stable MANET environments, the underlying mechanism for selecting and forwarding service requests in the significantly challenging networking environment of opportunistic networks has not been addressed. The problem comprises three stages: i) selecting an appropriate service sequence set out of available services; ii) forwarding service inputs to the device hosting the next service in the composition; and iii) routing final service outcomes back to the requester. The proposed algorithm derives efficiency and effectiveness by taking into account the service load and location of devices providing the services, as well as intermittent connectivity, to select a particular service set. Through extensive simulations on real and synthetic traces, we show that by using only local knowledge collected in a distributed manner, performance close to a real-time centralized system can be achieved.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*store and forward networks, wireless communication*

General Terms

Algorithms, Design, Performance

Keywords

Distance/Location Modeling, Opportunistic Networks, Service Provisioning

MSWiM'11, October 31–November 4, 2011, Miami, Florida, USA.
Copyright 2011 ACM 978-1-4503-0898-4/11/10 ...$10.00.

1. INTRODUCTION

In recent years, the number of multi-functional, personal smart devices has been increasing at a high rate. The possibility of such devices coming within communication range of each other is enhanced by the presence of multiple embedded radios. While, such opportunistic contacts between pairs of such devices have been exploited by the opportunistic networking paradigm [16, 19, 25], exploiting resources to execute remote services is an area of research that has received attention in recent times [17, 18]. The creation of a services-rich environment by effectively making services available on each device, accessible to applications on other devices in opportunistic environments is a major challenge. This paper investigates efficient service composition in opportunistic networks where services are available on devices in physical proximity of a user. Such environments include parks, malls, streets in a city or other social gathering places where citizens interact. In particular, an efficient mechanism for selecting services to complete a request and forward service parameters to corresponding devices in an opportunistic network is proposed.

Heterogeneous resources available on devices can be abstracted as services to simplify the interface and have platform independence. Services from multiple devices can be composed to provide enhanced functionality [13]. For example, to complete a task like '*take a picture* and share it on a *map* in my *current location*', services offered by a camera, GPS receiver and map provider need to be composed [2]. Service composition will be applicable in several areas - pervasive healthcare, intelligent transportation systems, crisis management, etc., [5].

When two devices are within communication range, there exists an opportunity to exploit each other's services. If the contact period is sufficiently long, nodes can utilize each other's resources to execute remote services. On the other hand, if the contact time is insufficient, service parameters can be exchanged during the contact, service can be executed offline and service outputs transmitted to the requester using opportunistic contacts. In Passarella et al. [18], multiple copies of a request are initiated to minimize delay in service execution and to increase robustness but only one-hop communication (i.e. a requester waits until a direct contact with service provider) is used. Also, a few other works propose efficient schemes and use multi-hop paths for service discovery and invocation in opportunistic networks by using a set of proxies [15] or location of the user [24]. However, composition of multiple services is not considered in these works. For composition of services, a number of middleware frameworks and architectures are proposed but these do not discuss the actual forwarding mechanism [2,7,13]. Routing is considered in [8,28] by modeling the expected colocation time between nodes based on past interactions. However, service composition fails if participants do not remain directly connected. In contrast in this paper, service composition is performed using multi-hop paths that can relay information to the service providers and back to the service requestor even when an end-to-end connected path does not exist. Furthermore, alternative paths are considered to increase the success of composition.

The problem of routing requests for service composition is different from traditional routing in two key aspects: i) service load at destination needs to be taken into account; and ii) future forwarding path needs to be considered for the remaining services that need to be composed in a sequence. Our algorithm selects a service sequence set by taking into account the service load and temporal distances between nodes. These values are measured in a distributed manner by using only opportunistic contacts. Temporal distance provides a measure of relative location of other nodes. The elegance of our solution lies in the fact that while a particular service sequence is selected by considering the temporal distance between devices, the actual routing scheme used to forward service request and parameters can be different. For example, to meet a service request of *encrypting and compressing* a file, our algorithm chooses a sequence of services (e.g., an encryption service and then a compression service) that can be composed such that these services are provided at devices in proximity of the service requester. Then, an underlying routing scheme is used to forward service request to the device providing encryption, encrypted result to the device providing compression and then final result back to the service requester.

Extensive simulation results are provided to demonstrate the performance of the proposed method in terms of service composition success, composition length, number of hops, and delays. The performance is analyzed for a range of service density, number of nodes, service request rates, request timeout durations, and routing mechanisms. It is shown that: i) composition is better than searching for the exact single service match; ii) use of multi-hop paths improves performance (service completion rate and delay) significantly as compared to one-hop direct forwarding; and iii) using only local knowledge (collected from opportunistic contacts) about load at and temporal distance from other nodes, performance close to a centralized system (exact load and temporal distances between nodes are known) can be achieved.

In rest of the paper, the terms devices and nodes are used interchangeably. The paper is organized as follows: Section 2 provides the system description. Temporal distances and service loads are modeled in Section 3. Service composition mechanism and algorithm is described in Section 4 followed by simulation results and analysis in Section 5. Section 6 discusses related work and other opportunistic forwarding schemes in literature and Section 7 concludes with directions for future work.

2. SYSTEM DESCRIPTION

2.1 Service model

We employ a service graph that represents which services can be combined in a sequence based on their input requirements. This can be done by using service ontologies and semantics in a hierarchical manner [13]. For simplicity, we represent a service based on its input and output type to generate a service graph i.e. two services can be composed if output type of one is same as required input to the other. For n_d input/output types, possible services are of type s_{xy} such that input type $x \in [1, n_d - 1]$ and output type $y \in [x + 1, n_d]$. This representation illustrates services with different levels of functionality. A service s_{xy} is of functionality k if $y = x + k$. Suppose, s_{12} and s_{23} represent services *decompress* and *decrypt* respectively, then service s_{13} represents a service of higher functionality that can perform both decryption and decompression. A service with high functionality like s_{14} provides same transformation that is otherwise possible by composition of two or more services with lower functionalities (e.g., $\{s_{12}, s_{23}, s_{34}\}, \{s_{12}, s_{24}\}$ or $\{s_{13}, s_{34}\}$). Services with varying functionalities are used to demonstrate the tradeoff between selecting an exact service (that can be further away in the network) versus composition of smaller services (that might be available in proximity of service requester). To account for task and device heterogeneity, execution time of each service is distributed exponentially on the devices. The performance of service composition is explored by varying the repetition of each service i.e. the number of nodes providing that service.

2.2 Mobility model

It has been shown that real mobility traces show a power law distribution of Inter-Contact Times (ICTs) [3] in contrast to previously assumed exponential distribution. Since random waypoint and random direction models lead to exponential Inter-Contact Times [23], they are not suitable to represent nodes' mobility. In addition, it has been shown that real mobility traces have power-law flight-lengths and pause-time distributions [20]. Therefore, in our analysis and simulation, we generate synthetic traces using levy walk mobility model [20]. Levy walk represents a more realistic scenario in comparison with [18] where each node is equally likely to meet any other node.

Mobility plays a central role in the selection and forwarding of service requests in an opportunistic network. Network environment is made heterogeneous by varying the mobility characteristics of nodes. Based on levy walk, though each node always follows the same step length, pause time distribution and speed for itself, these parameters are varied across the network for other nodes. This is reflected in Table 2 that shows the values for slow and fast moving nodes. In addition to synthetic traces, results on real mobility trace from a State Fair [21] are also analyzed. The corresponding results support applicability of our proposed algorithm in homogenous (nodes with similar mobility characteristics) as well as heterogeneous (nodes with varied mobility characteristics) environments. The focus of this paper is to provide services that are in physical proximity of a user. Therefore, results obtained in schemes [6,12] for social networks, where delays are to the order of hours and days, are not valid for our application scenario.

2.3 Forwarding Schemes

There are several forwarding schemes proposed in the literature on opportunistic networks. These are discussed in Section 6. The following forwarding schemes have been found to work best in dynamic environments: i) use of timer transitivity (TT) [25] or temporal distance between nodes in homogenous (nodes with similar mobility characteristics) environments; and ii) use of encounter rate (EBR) [16, 26] in heterogeneous environments (nodes with varied mobility characteristics). In addition, we have proposed a modified version of timers (MT) [22] that works efficiently over small forwarding paths. In our simulation studies, we compare performance of the proposed selection and forwarding mechanism for service composition under 4 forwarding schemes - direct or one-hop, TT, EBR, and MT.

3. RESOURCE MODELING

3.1 Disconnected paths

In opportunistic networks, nodes providing a service may not be directly connected to the service requested. This leads our investigation into modeling the physical distance between nodes when these are devices carried by people. In [14], Kim et al. show that the magnitude of typical displacement of a mobile node is given by $r = t^{\gamma/2}$ where r^2 (denoted $M(t)$ in [14]) is the mean square displacement of a node at time t from its position at time 0 and $\gamma \in [0.5, 2]$. Then, the distance d between two nodes when they start from same position at time 0 is given by $d = 2r\sin(\theta/2)$ where $\theta \in [0, 2\pi]$ is the angle between their displacements and is uniformly distributed in the range. Therefore, expected displacement is given by

$$E[d] = 2r\frac{1}{2\pi}\int_0^{2\pi}\sin(\theta/2)d\theta = \frac{4r}{\pi} = 4t^{\gamma/2}/\pi. \qquad (1)$$

Interesting result is that the physical distance between nodes is related to the separation in time i.e. time since they moved out of each other's transmission range. For this reason, we use similar timers as proposed in [25] and study properties of the shortest temporal distances between nodes. Several works have used the intuition behind this idea to approximate the physical distance between nodes and characterize the temporal distance [1, 9, 10, 22, 25, 27]. A shortest temporal distance between two disconnected nodes is the minimum time some sequence of contacts can relay information from one node to the other [27].

3.2 Shortest temporal distance

In essence, shortest temporal distance (t_{ij}) provides the minimum time (using epidemic forwarding) it would take for some information to travel from node i to node j. Therefore, to compare efficiency of an underlying forwarding scheme to send requests to different nodes in network, we simply compare this lower bound on the propagation time. For example, node i can select a service provider node j or k based on which of these has smaller distance (t_{ij} or t_{ik}). Since, no centralized infrastructure exists in an opportunistic network to provide t_{ij} and t_{ik}, the approximations t_{ji} and t_{ki} are used as they can be measured at node i in a distributed manner by using simple timers [Section 4.1]. We analyze the shortest temporal distances maintained at nodes in a real mobility trace from State Fair. In this paper, the term temporal distance always refers to the shortest temporal distance. It is found that even though actual nodes in the shortest path may be different, the temporal distance t_{ij} is approximately same as t_{ji}. This is demonstrated in Figure 1. We measure the absolute difference $|t_{ij} - t_{ji}|$ for all node pairs i,j after every 30 sec in a 90 minute duration of State Fair mobility trace [21] with 18 users. Figure 1 shows the CDF of these values. Since real mobility trace is used, different levels of connectivity are investigated by changing the transmission range tr from 100m to 125m. At higher level of connectivity (tr=125m), differences are smaller than at lower level (tr=100m). Also as shown in Figure 1, differences between temporal distances are less than 5 minutes for 50% and less than 10 minutes for around 80% of pairs at all times which justifies the use of approximation t_{ij} for t_{ji}.

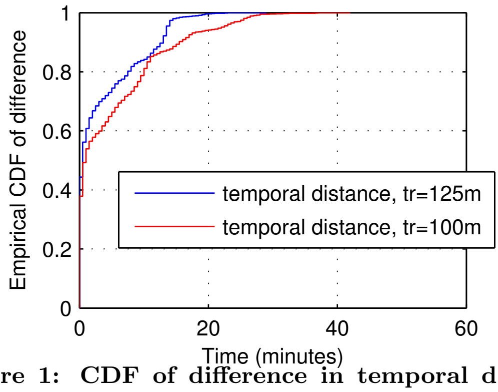


Figure 1: CDF of difference in temporal distance $|t_{ij} - t_{ji}|$ for all node pairs i, j at different transmission ranges (tr)

3.3 Service loads

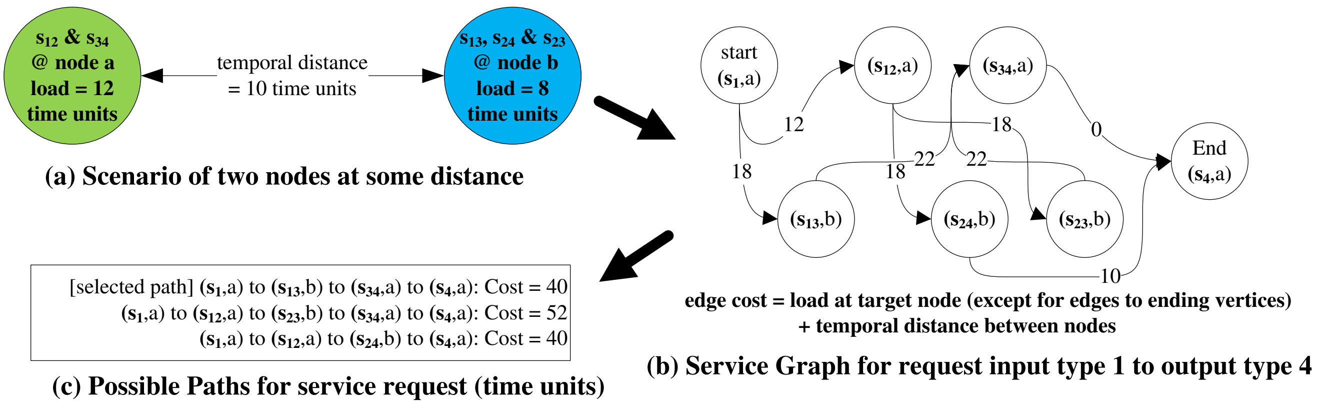
Each node provides one or more services, and maintains a FIFO queue for all service requests. Thus, even though a requester may be connected to a service provider, the request may take a long time to complete because service provider is busy. Therefore, one needs to take into account the current service load at the destination in order to compose services in minimal time. This can be done by sharing service load of all nodes in a distributed manner [Section 4.1]. Our service selection algorithm takes into account the service load at destination together with the temporal distance estimate. Though, statistical mechanisms can be used to predict the future service loads and provide better estimates, our current analysis uses a moving time-window average for the load estimate l at each node with a decaying factor α of 0.5. Load in current window l_{cw} is calculated by the total number of pending requests multiplied by the expected service execution time. Then estimate of load is updated by following rule: $l = \alpha l_{cw} + (1 - \alpha)l_{old}$ where l_{old} is the load value l in previous time window. We use a time window of 30s in our analysis.

4. SERVICE COMPOSITION

In this section, we describe how the information about temporal distance, service loads, and available services can be shared among nodes in the network. The link costs on a service graph are updated based on temporal distance between nodes and their current service loads. A particular service composition is selected based on the shortest path in this graph. Overview of the key algorithm steps is also provided in Figure 2.

4.1 Estimation of temporal distance and load

We are interested in temporal distance between nodes. An estimate of this distance at some time t_0 can be used to


s_{12} & s_{34} @ node a load = 12 time units
temporal distance = 10 time units
s_{13}, s_{24} & s_{23} @ node b load = 8 time units
start (s_1,a)
(s_{12},a)
(s_{34},a)
End (s_4,a)
(s_{13},b)
(s_{24},b)
(s_{23},b)
12
18
18
22
22
18
0
10
edge cost = load at target node (except for edges to ending vertices) + temporal distance between nodes

[selected path] (s_1,a) to (s_{13},b) to (s_{34},a) to (s_4,a): Cost = 40
(s_1,a) to (s_{12},a) to (s_{23},b) to (s_{34},a) to (s_4,a): Cost = 52
(s_1,a) to (s_{12},a) to (s_{24},b) to (s_4,a): Cost = 40

(a) Scenario of two nodes at some distance

(b) Service Graph for request input type 1 to output type 4

(c) Possible Paths for service request (time units)

Figure 2: Overview of Composition Selection Algorithm: (a) an example scenario, (b) construction of service graph, and (c) possible paths for request completion

approximate it for a later time that is not too long after t_0. In order to estimate temporal distance from other nodes in a distributed manner, each node keeps a timer for other nodes in networks with an update rule as defined hereafter. Nodes also share the load values with rest of the network. This is in contrast with centralized estimates of load that are used in [18]. Each node maintains a load value for other nodes in network. Let $t_a(i)$ denote the time elapsed since node a last made contact with node i, where $t_a(a)$ is always set at zero. Also, let $l_a(i)$ denote the estimate for load of node i at node a. Local timer values for each node are incremented after every time unit (e.g., 30s, 60s etc. depending on devices). When node a comes into contact with some other node b, it updates its timers and loads estimates for all nodes from whom node b has a smaller temporal distance. This rule is defined as follows: $\forall i \neq a$ if $t_b(i) < t_a(i) - t_{av}$, then set

$$t_a(i) := t_b(i) + t_{av}$$
$$l_a(i) := l_b(i)$$

where t_{av} is the measure of distance between two nodes when they are within each other's transmission range. Every node performs this update when it comes into contact with another node. Note that $t_a(i)$ is same as temporal distance t_{ia} and is used to approximate t_{ai} at node a as described in Section 3.2. The value of t_{av} is a small constant greater than zero but less than or equal to one time unit (increment by which local timers are updated) i.e. $t_{av} \in (0, 1]$. Note that it's a different definition of t_{av} from that in [25] where the value of average time required to travel between the two nodes is included. A small value of t_{av} is used as we are interested in the time a request will take in forwarding and not the physical distance between nodes. Value of $t_{av} > 0$ creates a gradient in a connected subset of nodes such that forwarding paths with smaller number of hops are selected. Based on this smaller value of t_{av} we have proposed a forwarding scheme modified timer (MT) [22] in which messages are forwarded to nodes with smaller timer from the destination. MT is used as the default forwarding scheme in all simulations and is found to perform efficiently when temporal distances are small (under 15min).

4.1.1 *Service advertisement and controlling distribution radius*

By keeping track of other nodes through use of timers, the service composition mechanism retains information only about nodes that are in close vicinity. This makes our approach scalable for large networks, as nodes have a direct mechanism to filter unnecessary information about devices that are further away. Thus, each time two nodes come into contact, they only update the timers and service load estimates for those remaining nodes in network whose temporal distances (implied proximity of location) are smaller than a certain threshold. Similar timers can directly be used to control epidemic forwarding of description of services that other nodes provide to a limited space. As network size grows, nodes only maintain a fixed set of values to efficiently select and forward service requests.

4.2 Service Graph

Each device maintains a local service graph $G = (V, E)$ based on its view of rest of the network. A simple case is described in Figure 2, where services provided by two devices a and b are used to construct a service graph. The number of devices, services (including repetitions) and input/output types are represented by N, N_s and n_d respectively. The service graph G has two types of vertices $V = \{V_1, V_2\}$, where a vertex $v \in V_1$ is a device and service pair such that $|V_1| = N_s$, and vertex $v \in V_2$ is a device and input/output type pair such that $|V_2| = n_d$. The two types of the vertices are shown in Figure 2b: i) (s_{12}, a) is a vertex of type V_1, represents that service s_{12} is provided at node a; and ii) (s_1, a) is a vertex of type V_2, represents that there is an input/output of type 1 (denoted s_1) entering/exiting an application at node a. Vertices of type V_2 are only maintained for the device constructing the service graph.

A directed edge (u, v) between two vertices u and v exists if (i) input/output type of first vertex $u \in V_2$ is same as service input of second vertex $v \in V_1$ e.g., $(s_1, a) \rightarrow (s_{12}, a)$; (ii) service output of first vertex $u \in V_1$ is same as service input of second vertex $v \in V_1$ e.g., $(s_{12}, a) \rightarrow (s_{24}, b)$; or (iii) service output of first vertex $u \in V_1$ is same as input/output type of second vertex $v \in V_2$ e.g., $(s_{24}, b) \rightarrow (s_4, a)$. The cost of an edge in cases (i) and (ii) is the sum of temporal distance between devices of corresponding vertices and the load at the device of second vertex. For example, cost of $(s_{12}, a) \rightarrow (s_{24}, b)$ is 18 which is sum of temporal distance from a to b (10) and load on b (8). However, the cost of an edge in case (iii) is simply the temporal distance between devices of corresponding vertices as it is the cost of routing final results back to the service requester. Nodes of type V_2 are needed to take into account the temporal distances: (a) from the service requestor to the node providing the first service; and (b) from the node providing the last service to the requestor.

4.3 Algorithm

Our final algorithm is to compute a shortest path on a service graph based on a service request (input type and desired final output type). From the collected information about service loads, temporal distances, and services provided at the neighboring nodes, a device creates the service graph. In Figure 2, a simple case is described for a service request from input type 1 to output type 4 at device a. Using the service load and temporal distance estimate, the edge cost is computed from each service output to compatible service inputs. From the final graph, shortest path is computed from the starting vertex (s_1, a) to ending vertex (s_4, a) by Dijkstra's Algorithm. In this case the shortest path is $(s_1, a) \rightarrow (s_{13}, b) \rightarrow (s_{34}, a) \rightarrow (s_4, a)$ i.e. the next service s_{13} is to be run at device B. This information is given to the forwarding algorithm to route service request from device A to device B. After execution of this service in the sequence, the device hosting the service (device B) finds the next service in the path by re-computing the shortest path for pending request (input type 3 to output type 4) and gives the destination address to the underlying forwarding scheme. This is done until the final results are routed back to the service requestor. The path is re-computed after execution of each service in the sequence as the network topology can change in that duration to provide alternate paths that are more feasible. However, one may choose to follow the same path computed at the service requestor if the network topology does not change very fast.

The service graph has a total of $N_s + n_d$ vertices. Since, the cost of edges from services on one device to services on another is same, service graph requires only $O(N^2)$ updates instead of $O(N_s^2)$. The actual cost of update is even lower, as information about vertices experiencing changes are updated. In a large network, the size of graph can be controlled by retaining information of devices that are in close vicinity. One run of Dijkstra's Algorithm $(O((N_s+n_d)^2))$ finds shortest path from one input type to all possible output types i.e. it finds composition paths of all service requests that have the same input type. Thus, at most, a node only requires to run Dijkstra's Algorithm n_d times to compute composition path for any number of pending service requests.

4.4 Levels of awareness

Note that even though any forwarding scheme can be used to forward service requests efficiently, the selection of service set (and consequently the destination nodes for forwarding) still needs to take into account the likelihood of delivery. Even though our algorithm is independent of any forwarding scheme, selection of a particular service set has a significant impact on performance (service completion rate and delay) of any forwarding algorithm. The selection of a particular service set can be made with different levels of awareness about rest of the network. More specifically, the edge cost between vertices of service graph is based upon the knowledge of temporal distance and service loads. These levels are summarized in Table 1 for node a where $\widehat{l_a(i)}$ represents estimate for load of *node i* at node a and $\widehat{t_i(j)}$ represents estimate for temporal distance $t_i(j)$ at node a. Note that, timers maintained by node a only provide $t_a(i)$, therefore, different levels of awareness are used to estimate $t_i(j)$ for $i, j \neq a$.

Table 1: Settings for different levels of awareness

Level	$\widehat{t_a(i)}, \widehat{t_i(a)}$	$\widehat{t_i(j)} : i, j \neq a$	$l_a(i)$
minimal	1	1	0
local	$t_a(i)$	$t_a(i) + t_a(j)$	$l_a(i)$(delayed)
global	$t_a(i)$	$t_i(j)$(delayed)	$l_a(i)$(delayed)
perfect	$t_a(i)$	$t_i(j)$	$l_a(i)$

At the very basic level it is assumed that a node knows about services being provided in the environment but does not have an estimate of its temporal distances from particular devices. This is reflected as *minimal* in Table 1 as it requires no housekeeping.

The next level of *local* knowledge is by use of timers as described in Section 4.1. A node has knowledge of its temporal distance from other nodes in the network (e.g., $t_a(i)$ for node a). A node also knows about the latest service load at other nodes, which is forwarded to it in a distributed manner [Section 4.1] and therefore incurs some delay. This level requires exchanging $O(N)$ values $(t_a(i), l_a(i)\ \forall i$ at node $a)$ per contact. In order to update link cost between services provided at other nodes, a node needs to be aware of temporal distances between other nodes (i.e. $t_i(j) : i, j \neq a$). However, this information is not available in *local knowledge*. Therefore, a node uses its own timer values for approximation as shown in Table 1. For example, node a uses $(t_a(i) + t_a(j))$ as an approximation for temporal distance $t_i(j)$ because $|t_a(i) - t_a(j)| < t_i(j) < t_a(i) + t_a(j)$. Even though this approximation (an upper bound on $t_i(j)$) is not accurate, it still gives an idea of proximity of a device. Hence, the performance results using local knowledge are fairly close to ones achieved by perfect knowledge.

The next higher level of *distributed global* knowledge is when a node also receives timers of other nodes in a distributed manner i.e. the estimates of temporal distances between all nodes which requires exchanging $O(N^2)$ values $(t_i(j), l_a(i)\ \forall i, j$ at node $a)$ per contact. However, this information about temporal distance between other nodes is still not up-to-date because of propagation delay. Therefore, we further make comparison with a *perfect* system which assumes centralized knowledge of temporal distances and service loads i.e. all information is available to nodes without any delay.

5. SIMULATION AND ANALYSIS

Performance of service selection algorithm is evaluated for types for levels of awareness, forwarding schemes, repetition of services, request rates, and node densities. Extensive simulations are run on real as well as synthetic mobility traces. The real mobility trace has been collected from participants that carry GPS receivers which log position at 30 second intervals. These traces are collected in five different environments, but we show representative results from a State Fair [21]. In order to make a comparison with suitable number of nodes, track logs from each day are considered to be a separate user. These logs have durations from around one to ten hours each day. We truncate all logs to 90 minutes during which 18 users record their location. Synthetic mobility traces are generated using Levy Walk mobility model [20] with 20 (10 slow, 10 fast moving) and 40 (30 slow, 10 fast

Table 2: Levy mobility trace used for simulation

Description	slow	fast
Exponent of step length distribution	1.6	
Exponent of pause time distribution	0.6	
Velocity of node	1m/s	5m/s
Maximum step length	5m	25m
Minimum pause duration	30s	7.5s
Maximum pause duration	600s	60s
Simulation area	500×500m	

Table 3: Simulation Parameters

Description	Range [**default value**]
Number of services	[**20**]
Number of service providers	[**20**], 40
Repetition of each service	1, [**2**], 3, 4
Request timeout	10, [**15**], 20, 30
Request rate per node	0.2, [**0.4**], 0.67, 1/min
Trace duration	[**10 hours**]
Transmission range of device	[**100m**]

moving) nodes. Details about trace settings are provided in Table 2.

Simulation setup: All plots show the average of five simulation runs and the error bars are plotted on the minimum and maximum values in those runs. Table 3 summarizes the generic simulation parameters. In all following plots, default values of these parameters are used unless it is mentioned otherwise under the plot. Parameters used for forwarding schemes are EBR-Encounter rate window=10 minutes, t_{av}=10min for TT and 0.5min for MT. A total of seven input/output types are used to create $C^7_2 = 21$ unique services. Service s_{17} is dropped to leave 20 unique services that are provided at 20 nodes in the network. Note that for a service repetition level of 3, all services are provided by 3 unique nodes (randomly selected) such that every node provides exactly 3 services. Service requests are only generated for services with $k \geq 4$ and the average execution time of service is set to 30s. Each service request times out after 15 minutes so that previous service requests do not overload the system. In all simulation results, service requests are not generated in the last 15 minutes (duration of request timeout) of the trace duration. We consider a simulation analysis including both a transient and a steady regime phase. The transient regime lasts for approximately the initial 30 minutes, after which all nodes acquire information about network resources and system enters the steady regime.

5.1 Direct and multi-hop forwarding

Figure 3 shows the service completion rate in the Levy Walk mobility trace. In all these cases *local* level of awareness is used to construct service graph. The search for a single service that matches the request exactly only fulfills 40% of the requests. However, when the condition is relaxed to compose multiple services to meet the requirements, about 50% of service requests are completed. As a further improvement, when multi-hop forwarding paths are used to forward requests and service outputs, 70% requests are completed. This shows the significant improvement that service composition with multi-hop forwarding paths can have over other mechanisms. While such mechanisms are discussed in [8, 28], our algorithm tolerates disconnections and utilizes paths that are formed over time. As a result, in addition to utilizing services at devices with directly connected paths to a node, ones that are present in the nearby vicinity can be invoked at the expense of slightly higher delays. However, in opportunistic networks, multi-hop forwarding still yields lower delays as compared to direct one-hop forwarding. This is shown in Figure 4 where 50% of compositions are made within 5 minutes using multi-hop forwarding, whereas it takes 12 minutes in direct forwarding. In case when only exact match is searched in the network, only 40% requests are completed after 15 minutes. The CDF plot (proportion of samples less than a value) is based on the aggregate delay per service from five simulation runs. As seen from Figure 3, the percentage of completed services increases sharply in the first 30 minutes (transient regime) and then becomes steady. In the transient regime the system starts with no initial service loads. Therefore, delays of only those compositions are considered that are generated once the system is well inside the steady state regime. Figure 4 and later plots show delays of only those service requests that are generated after first two hours of simulation.

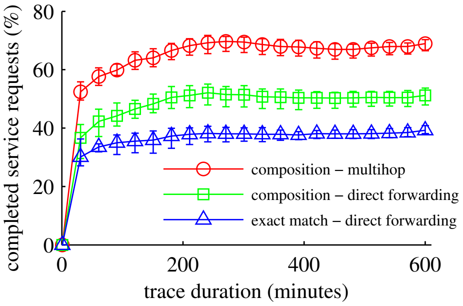


Figure 3: Higher service completion in multi-hop composition

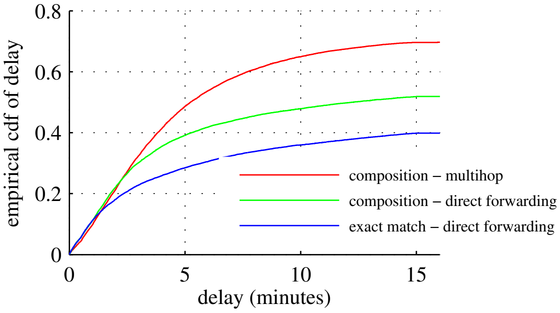
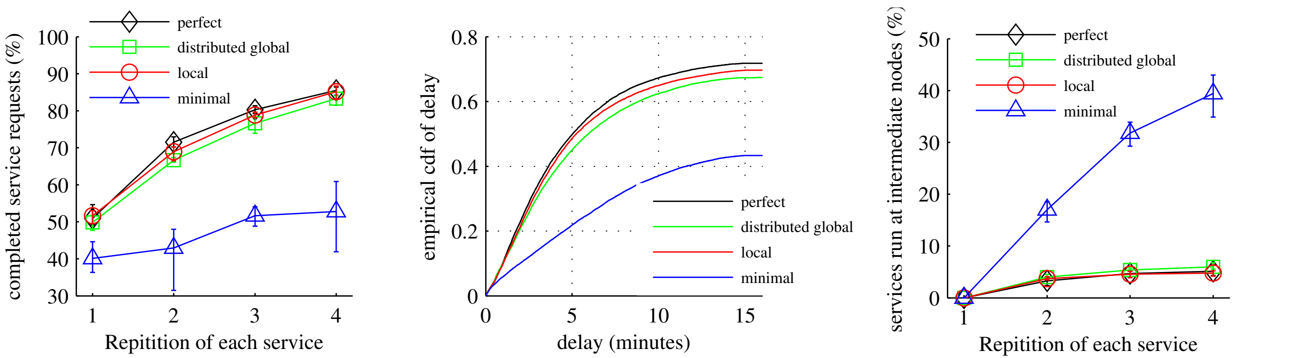


Figure 4: Lower delay in multi-hop composition

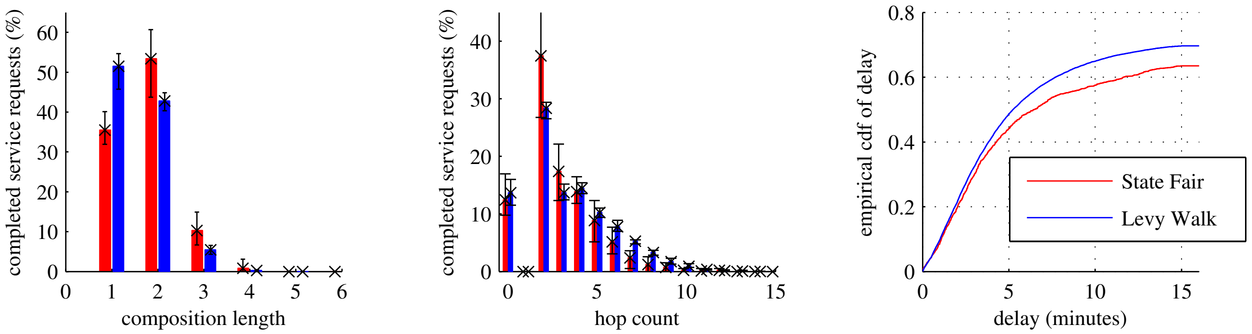
5.2 Levels of awareness

While there is significant improvement in service completion rate and delays when some knowledge about temporal distances from other nodes is used, there is not much difference in the performances of perfect, globally aware and local schemes. This is clear from Figure 5a where in a range of service densities, completion rate of perfect, distributed global and local levels of awareness is within 3% of each other. Figure 5b shows the delays when each service is repeated twice i.e. two different nodes provide that service. The local scheme has delays between the globally aware and



(a) (b) (c)

Figure 5: Levy Walk: With different levels of awareness (a) service completion, (b) delay, and (c) services run at intermediate nodes



(a) (b) (c)

Figure 6: State Fair [red] and Levy Walk [blue] mobility trace: (a) composition length, (b) hop count, and (c) delay profiles for multi-hop service composition

perfect scheme as shown in Figure 5b. One explanation for this behavior is that approximation for temporal distance between other nodes is better than delayed information of actual temporal distance between those nodes. Also, by the time a service is forwarded and executed on a device, the network topology changes so much that it does not make much difference if local knowledge is used or perfect. Since, all three schemes have similar performance; we select the scheme with local knowledge for further analysis as it is more light weight (requires exchanging $O(N)$ temporal distance and load values per contact instead of $O(N^2)$).

In Figure 5c, we validate the selection of correct devices to compose all services. At different service densities, percentage of services that are run at intermediate nodes while they were in route to some other device is considered (i.e. when required service is opportunistically found at a device different from one selected by the algorithm). While it is quite high around 40% when minimal knowledge about network is available, it is reduced to under 5% when some knowledge about network topology is used. This validates that our service selection algorithm selects the correct devices to execute services in a composition.

5.3 Composition length, delay, and hop profiles

Figure 6 shows the detailed profile of multi-hop service composition for State Fair and Levy Walk mobility trace. Most compositions are comprised of 1 or 2 services and then there are some of three, and four services as shown in Figure 6a. In our analysis, services are distributed such that every service is provided by at least one node in the network. However, as described earlier, restricting the completion to the exact match degrades performance. Therefore, all compositions (around 60%) of length more than 1 that are shown in Figure 6a are optional, and are made to achieve higher service completion rates and minimal delays. However, these compositions come at the expense of a few extra hops as shown in Figure 6b. Most of the times (about 80%) complete services are composed by using less than 5 hops. Around 15% of requested services are found at the device itself. This causes a hop count of zero as shown in Figure 6b. If the service is not found at the requesting devices, service completion incurs at least 2 hops (one to send request and one to receive results). Figure 6c shows the empirical CDF of the time taken from service request to receiving composed results, routed back from the last node in the composition path. Around 60% of completed service requests have a delay of less than 10 minutes whereas close to 50% of these are completed in less than 5 minutes. Each service is repeated twice in this run. For higher number of repetitions, delays are lower and percentage of completed services reaches close to 100% (85% for four repetitions, see Figure 5a). This means that for, reasonable service densities, applications that can tolerate delays to the order of 5 to 10 minutes, opportunistic networks can provide a reasonable service completion rate in an environment similar to State Fair. Results in State Fair and Levy Walk are fairly similar which means that evaluations based on Levy Walk traces are close to real mobility traces.

5.4 Effect of forwarding scheme, request load, request timeout, node and service density

Figures 8 shows the service completion rate of 3 different underlying forwarding schemes used in the service selection

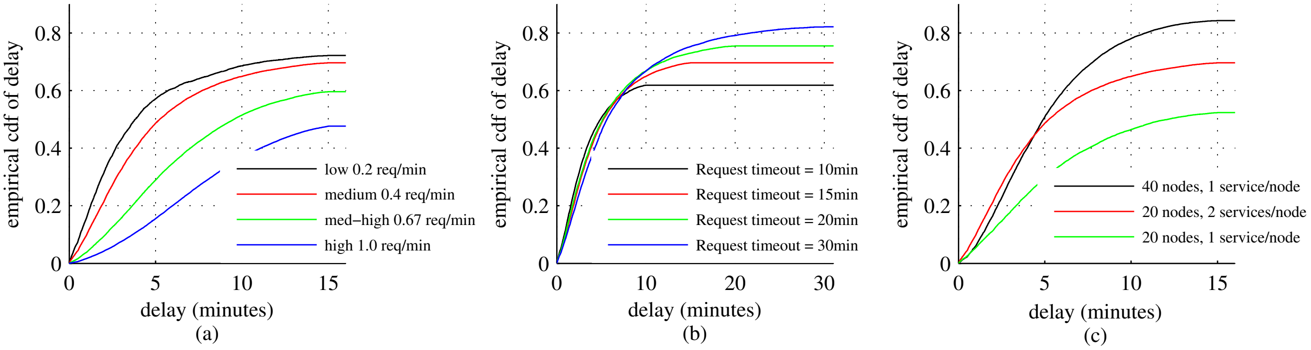


Figure 7: Levy Walk: Higher service completion and lower delay at (a) low loads, (b) long request timeout, and (c) high node, service density for multi-hop service composition

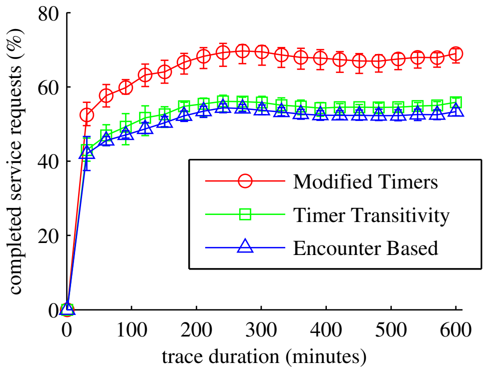


Figure 8: MT with better service completion

algorithm. Only single copy of each request is forwarded. MT shows better delivery rate (about 30-40% higher than other schemes). Network performance is also analyzed under different service load conditions and levels of connectivity. Requests rates are varied from 0.2 to 1 request per minute per node. Figure 7a shows that service completion rate decreases under high loads. At high node/service density Figure 7c shows that almost 85% of all requests can be composed. Also, when a service request does not timeout for a longer time, more requests can be completed [Figure 7b]. Thus, while service compositions can be made in a moderately connected network, our algorithm adapts well in sparse scenarios with higher service request loads and short request timeout duration. Note that variation in load greatly changes the delay profile under 5 min as shown in Figure 7a. This is because higher loads result in longer service queues and services take longer to execute even when a connected path to service provider exists. Whereas variation in request timeout changes the delay profile at over 10 min as shows in Figure 7b. This is because longer request timeouts keep the service requests in the queue which are otherwise dropped without completion. As a side effect, average service load at nodes increases slightly. For this reason, 40% of compositions are made in 4.5 min with request timeout of 30min but only in 3.5min with timeout of 10 min due to change in average load at nodes. Figure 7c shows that average delays are lower when 40 services are distributed in 40 nodes instead of 20 nodes even though the number of generated requests is double in a network with 40 nodes. This is because the network with 40 nodes is more dense and connected.

6. RELATED WORK

Brief overview of related works in single service provisioning [15, 17, 18, 24] and service composition [2, 7, 8, 13, 28] was presented in the introduction. In this section, we discuss the highlights of these works in greater detail. There has been a significant research on semantics and ontologies to describe services and compositions (see [4] for survey and issues of service composition in mobile environments). Services to be composed can either be strictly defined in the request (static composition) [11], or can be computed at run-time (dynamic composition) based on the request [13]. Dynamic composition looks for alternate sets of services in the current environment that can be composed to provide the required functionality. Services are modeled as directed attributed graphs [13], to find possible compositions. We leverage this work for service representation to find possible compositions and explore mechanism to execute such compositions in an opportunistic network. Passarella et al., [17, 18] investigated optimal policy for service execution in opportunistic networks. An optimal policy is derived for the level of replication to receive service results in minimal time but only a single service is requested and the service requester waits until a direct contact with service provider. In contrast, the algorithm presented in this paper uses a single instance of service request to compose multiple services, collects all information in a distributed manner, and uses multi-hop forwarding paths to upload requests to and download results from service provider. Since, multihop forwarding paths are used, next we describe the existing schemes to forward data in opportunistic networks that are applicable to our analysis.

Several schemes for forwarding messages in opportunistic networks have been proposed in the literature. Two key metrics used in open environments where participating nodes may not have social interaction history include, *time since last encounter* with destination [9] and *rate of encounter* [16, 26] . Time since last encounter provides good location approximation in dense mobile networks [10] but has a slow warm up time (i.e. it takes a long period of time before every node has made at least one contact with every other to make an estimate of its location). To reduce warm up time, Spyropoulos et al. [25] presented a timer transitivity (nodes exchange timer values for all destinations). Whereas, use of encounter rate provides good delivery rates in heterogeneous environments by identifying nodes that are highly mobile.

7. CONCLUSIONS

In this paper, we proposed a novel algorithm for service composition in opportunistic networks. With the proposed algorithm, mobile users can benefit from a larger set of services available locally in an environment. Proposed service composition makes efficient service selections from devices that are in close proximity. Key insight used in the solution is that temporal distance between devices provides a direct measure for reachability of nodes when an end-to-end connected path does not exist. This way, based on a service graph, a composition sequence can be selected to meet the requirements of service request while any routing scheme can be used to forward service parameters.

Through extensive simulations on real and synthetic mobility traces, we conclude that using only local knowledge about temporal distances and service loads, a large percentage of services can be composed in an opportunistic network using multi-hop paths between devices. In order to complete a request, searching for possible compositions is better than looking for a single service that matches the request exactly. Also, use of multi-hop forwarding paths is more efficient than one-hop direct forwarding. Performance of service composition improves (more requests are completed with lower delays) under high service density, high node density, long request timeout duration and low request rate. Future work will explore how additional information about a node's mobility and context can be combined with temporal distances to improve successful compositions in minimal time. Also, analysis of replicated service requests at different stages of composition to increase robustness will be investigated. Replication strategies can be devised based on proximity and number of devices providing the required service.

8. ACKNOWLEDGMENTS

The research work presented in this paper was carried out with support from the US National Foundations Grants ECCS-0824120, CSR 0834493 and European Commission under the FIRE SCAMPI (FP7-IST-258414), FET-AWARE RECOGNITION (FP7-IST-257756) Projects. The authors would like to thank Sidra Bashir for help in running simulations.

9. REFERENCES

[1] H. Cai and D. Y. Eun. Aging rules: what does the past tell about the future in mobile ad-hoc networks? In *Proc. ACM MobiHoc*, 2009.

[2] I. Carreras, L. Bassbouss, D. Linner, H. Pfeffer, V. Simon, E. Varga, D. Schreckling, J. Huusko, and H. Rivas. Bionets: Self evolving services in opportunistic networking environments. In *Bioinspired Models of Network, Information, and Computing Systems*, pages 88–94. 2010.

[3] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Trans. Mobile Comput.*, 6:606–620, 2007.

[4] D. Chakraborty, A. Joshi, T. Finin, and Y. Yesha. Service composition for mobile environments. *Mobile Networks and Applications*, 10:435–451, 2005.

[5] M. Conti and M. Kumar. Opportunities in opportunistic computing. *Computer*, 43(1):42–50, 2010.

[6] E. M. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *Proc. ACM MobiHoc*, 2007.

[7] O. Davidyuk, I. Sanchez, and J. Riekki. CADEAU: Supporting Autonomic and User-Controlled Application Composition in Ubiquitous Environments. In *Pervasive Computing and Communications Design and Deployment: Technologies, Trends, and Applications*. 2010.

[8] L. Del Prete and L. Capra. Reliable discovery and selection of composite services in mobile environments. In *Proc. IEEE EDOC*, 2008.

[9] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli. Age matters: efficient route discovery in mobile ad hoc networks using encounter ages. In *Proc. ACM MobiHoc*, 2003.

[10] M. Grossglauser and M. Vetterli. Locating nodes with ease: last encounter routing in ad hoc networks through mobility diffusion. In *Proc. IEEE INFOCOM*, 2003.

[11] X. Gu, K. Nahrstedt, and B. Yu. Spidernet: an integrated peer-to-peer service composition framework. In *Proc. IEEE HPDC*, pages 110–119, 2004.

[12] P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: social-based forwarding in delay tolerant networks. In *Proc. ACM MobiHoc*, 2008.

[13] S. Kalasapur, M. Kumar, and B. A. Shirazi. Dynamic service composition in pervasive computing. *IEEE Trans. Parallel Distrib. Syst.*, 18:907–918, 2007.

[14] S. Kim, C.-H. Lee, and D. Y. Eun. Superdiffusive behavior of mobile nodes and its impact on routing protocol performance. *IEEE Trans. Mobile Comput.*, 9:288–304, 2010.

[15] N. Le Sommer, R. Said, and Y. Mahéo. A proxy-based model for service provision in opportunistic networks. In *Proc. MPAC*. ACM, 2008.

[16] S. Nelson, M. Bakht, and R. Kravets. Encounter-based routing in dtns. In *Proc. IEEE INFOCOM*, 2009.

[17] A. Passarella, M. Conti, E. Borgia, and M. Kumar. Performance evaluation of service execution in opportunistic computing. In *Proc. 13th ACM MSWIM*, 2010.

[18] A. Passarella, M. Kumar, M. Conti, and E. Borgia. Minimum-delay service provisioning in opportunistic networks. *IEEE Trans. Parallel Distrib. Syst.*, PP(99), 2011.

[19] L. Pelusi, A. Passarella, and M. Conti. Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. *IEEE Communications Magazine*, 44(11):134–141, 2006.

[20] I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong. On the levy-walk nature of human mobility. In *Proc. IEEE INFOCOM*, 2008.

[21] I. Rhee, M. Shin, S. Hong, K. Lee, S. Kim, and S. Chong. CRAWDAD trace. `http://crawdad.cs.dartmouth.edu/ncsu/mobilitymodels/GPS/NC_State_Fair`, July 2009.

[22] U. Sadiq and M. Kumar. ProxiMol: Proximity and mobility estimation for efficient forwarding in opportunistic networks. In *Proc. IEEE MASS*, 2011.

[23] G. Sharma and R. Mazumdar. Scaling laws for capacity and delay in wireless ad hoc networks with random mobility. In *Proc. IEEE ICC*, 2004.

[24] N. L. Sommer and S. B. Sassi. Location-based service discovery and delivery in opportunistic networks. In *Proc. ICN*, 2010.

[25] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Efficient routing in intermittently connected mobile networks: the single-copy case. *IEEE/ACM Trans. Netw.*, 16(1):63–76, 2008.

[26] T. Spyropoulos, T. Turletti, and K. Obraczka. Routing in delay-tolerant networks comprising heterogeneous node populations. *IEEE Trans. Mobile Comput.*, 8(8):1132–1147, 2009.

[27] J. Tang, M. Musolesi, C. Mascolo, and V. Latora. Characterising temporal distance and reachability in mobile and online social networks. *SIGCOMM Comput. Commun. Rev.*, 40(1):118–124, 2010.

[28] J. Wang. Exploiting mobility prediction for dependable service composition in wireless mobile ad hoc networks. *IEEE Trans. on Services Comput.*, 4:44–55, 2011.