http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (6): 1098-1106, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.1098.1106

A Proximity-aware Approach for Discovering Adaptation Services

[1]Mohd. Farhan Md. Fudzee, [1]Azizul Azhar Ramli, [1]Mohamad Aizi Salamat,
[1]Mohd. Norasri Ismail and [2]Kamarudin Ambak
[1]Software and Multimedia Centre, Faculty of Computer Science and Information Technology,
[2]Faculty of Civil and Environmental Engineering, Universiti Tun Hussein Onn Malaysia,
Johor, Malaysia

Abstract: It is paramount to provide seamless and ubiquitous access to rich contents available online to interested users via a wide range of devices with varied characteristics. Recently, a service-oriented content adaptation scheme has emerged to address this content-device mismatch problem. In this scheme, content adaptation functions are provided as services by multiple providers. This elevates service discovery as an important problem. As most of the clients are using battery-powered devices, discovering closer providers are essential to reduce the accumulated time of receiving adapted content version. This study proposed a proximity-aware service discovery to address closer service discovery issue. The proposed mechanism uses estimation method to find closer service providers. The performance analysis verifies that our approach guarantees closer providers with a high probability, while at the same time significantly reduced the overhead up to one magnitude compared with the baseline brute force method.

Key words: Content adaptation, service discovery, proximity-aware, network latency, estimation method

INTRODUCTION

There is a substantial trend in the way people communicate through the World Wide Web (WWW); browsing, uploading, downloading or storing contents. This trend is aligned with the growing number of digital contents available on the internet. Most of the existing online contents are originally designed for display on desktop computers (Mohan *et al*., 1999).

On the other hand, the client devices vary in term of their sizes and capabilities (e.g., processing power, input and output facilities). Thus, it is becoming increasingly difficult for direct content delivery to varying devices without layout adjustment or adaptation (Mohan *et al*., 1999). For instance, not every device can play certain media types or formats without installing certain software or plug-in e.g., default iPhone cannot play a flash video file.

One way to address this problem is to use devices that are capable of handling these contents. However, changing client devices to suit the content is not cost effective thus renders it impractical. Another solution is to adapt these contents tailored to the existing client devices. This requires a mechanism called content adaptation. Currently, a Service-oriented Content Adaptation (SOCA) scheme has recently emerged as an efficient, flexible and scalable paradigm to perform content adaptation on the fly (Fudzee and Abawajy, 2011a; Fawaz *et al*., 2008).

A complete content adaptation may require a set of adaptation tasks (e.g., summarizing full text into a short summary and translating the short Spanish summary into English audio) to a set of content objects, e.g., text and audio (Shahidi *et al*., 2008). Each task is performed by a particular content adaptation function that potentially be provided by multiple services located across the wide-area network. On the other hand, a client normally uses a mobile device to browse web content through wireless network. A wireless network is characterized with limited bandwidth and pricey, while mobile devices are having limited battery capacity. As such, reducing time to deliver adapted content is crucial. One fundamental way to achieve this is by having closer providers, thus avoiding inefficient routing. For example, a client in Malaysia has its selected providers in Australia and hence its path to get adapted content may traverse distant providers in Australia. Avoiding such high latency hops improves the routing performance. This necessitates proximity-aware service discovery.

Corresponding Author: Mohd. Farhan Md. Fudzee, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn, Malaysia (UTHM), Johor, 86400, Batu Pahat, Malaysia
Tel: 0060174533796

In this study, the primary practical contribution is an approach endeavors to achieve closer providers with a high probability. Also, it reduces the overhead in term of measurement required.

SERVICE DISCOVERY PROBLEM AND RELATED WORK

Content adaptation is a multi-step process involving a number of services, each performing a specific adaptation task. Most of clients requiring content adaptation use mobile devices. These devices connect to the internet through wireless network. A mobile device is characterized with limited battery capacity, while wireless network is relatively more expensive compared to wired network. As such, reducing time to deliver adapted content is an important requirement. Therefore, the service discovery challenge is to discover closer providers (i.e., services) from a large number of available content adaptation services located in many places in the network. Having closer providers will relatively minimize the total time to provide the adapted content (Comer, 2006). Specifically, the service discovery problem of interest can be formulated as follow:

Let $T = \{t_1, t_2, Y, t_n\}$ and $S = \{s_1, s_2, Y, s_m\}$ be a set of adaptation tasks and a set of available services, respectively such that n>>m. Let |cs| represent the network proximity of a path that made of composite services m for serving a series of tasks n. Given a set of S, T and |cs|, the central problem is how to select closer composite services that minimize |cs|.

Service discovery is paramount to any service-oriented system. There are several approaches have been proposed to solve service discovery for the service-oriented content adaptation (Fudzee and Abawajy, 2011b). They can be divided into two dominant approaches: function-based and non-functional-based. Function-based approach utilizes the service's description (i.e., the function's name, its input and output parameters, preconditions and effects) to match the query. One of the well known function-based is keyword matching. It bases matching according to these descriptions, i.e., weighted keywords (Zhou *et al*, 2005). However, the discovery result may return a huge list containing inappropriate services that may not satisfy the requester's intended requirements. To overcome the aforementioned shortcoming, ontology is used. Ontology organizes keywords according to its semantic (Meditskos and Bassiliades, 2010). A reasoning process is then performed by exploiting the information included in the service description to best match the ontology. However, this approach alone is still inadequate as numerous services providing the same content adaptation function are having the identical semantic. A work by Plebani and Pernici (2009) extends this approach by additionally taking into account the service's interface structure. It uses similarity method to find alternative services that descriptively and structurally similar. As the similarity analysis is based on syntactical and structural factors, it can guarantee that similar services are provided semantically exact services; however, they still tend to return a large number of services (Kritikos and Plexousakis, 2009a).

On the other hand, non-function-based approach incorporates the service's Quality of Service (QoS) attributes together with the service's descriptions. QoS is a set of service attributes that encompass performance characteristic such as cost, reliability, availability, rating and reputation (Zaman and Abdullah, 2011). Efforts such as OWL-Q (Song *et al*., 2011) are trying to make QoS description more flexible to describe and present the formal description of a service. Existing methods such in Kritikos and Plexousakis (2009b), Dastjerdi *et al*. (2010) perform matchmaking between the client and the service's QoS. This requires that both the services and the client QoS must be known a priori. Manually discovering services from an Internet-scale list that specifically match the client's QoS requirements is time consuming and tedious.

In a service-oriented content adaptation, an important requirement to design an efficient service discovery is to consider how clients browse online content. Most of these clients are using mobile devices. These devices however, are battery-powered hence having limited power. Moreover, the built-in wireless or network adapter consumes additional power when activated. As such, reducing time to deliver adapted content is crucial. Composite services that are close to each other relatively reduce the transport time (Comer, 2006). Accordingly, Li *et al*. (2009) proposed distance-sensitive service discovery however, it only suitable for the wireless sensor network. Existing approaches (i.e., function-based and non-functional-based) do not take this aspect into account. Even though, there is a study in physic letter does mentions to exchange the function-based method in order to improve meta-GGA (Campo *et al*., 2012). What is required is a service discovery mechanism that utilizes service's attributes and considers closer service providers. One way to find closer providers is using brute force method. It requires each service to measure actual distance to one another and thus rendered this method impractical for internet-scale deployment. Therefore, it is vital to come out with a new solution that could overcome to this problem.

SERVICE DISCOVERY ARCHITECTURE

Proposed architecture: As depicted in Fig. 1 the service discovery for service-oriented content adaptation platform is composed of two interrelated layers: Information and look up. This architecture can be deployed in a distributed manner. As such, we can have multiple service registries and corresponding information tables. Such infrastructure deployment can avoid single point of failure and scalability issue. In this study however, distributed registries and queries are left for future work. Interested readers can refer to Fonseca *et al*. (2005). Thus, it is assumed that the local participant registry is known a priori. Therefore, it is recommended to preserve locality, since the attempt is to find closer services of ice discovery for service-oriented content adaptation Information layer contains the service registries and proximity information tables. A service registry lists services advertised by service providers. Proximity information table stores the network proximity information for each service. With the growing number of services in the large Internet-scale application, accurately estimating network proximity with minimal probing overhead becomes essential for scalable deployment (Sharma *et al*., 2006). There are a number of network proximity estimation methods such as Netvigator (Sharma *et al*., 2006) and Vivaldi (Dabek *et al*., 2004) proposed in recent years. These methods utilize either the IP address or landmark to retrieve geolocation information. Among these methods, Netvigator is demonstrated to have high accuracy of distance. We can use the same approach for conducting the landmark probing, so that the network distance between pair nodes can be estimated.

Each provider node measure distances to a given set of landmarks. During landmarks probing, it records the distances to the encountered milestones. In this scheme, a small number of landmarks are used for bootstrapping and a large number of milestones are used for refinement. No milestones deployment is required. Each node N*L is the total number of traceroutes being conducted asynchronously. This requires only a very minimal extra overhead of ACK packet transmission from the milestones. The details of proximity information table construction are described below (Sharma *et al*., 2006):

- Each node sends probe packets to the landmarks for round-trip time measurement. Each node is assumed to have landmark information known a priori. These packets may encounter milestones while routing to the landmarks. When a milestone node receives a probe packet, it sends an acknowledgment packet back to the node that originated the probe packet
- After the node receives all acknowledgement (ACK) packets from the landmarks and milestones, it constructs a landmark vector that includes the distances to all the landmarks and encountered milestones
- Each node submits its landmark vector V to a repository called proximity information table. Local service registry can link this information for services advertised in their list

In the look up layer, the service broker looks ups the service registry for suitable services that match the required adaptation tasks. Then, it uses discovery algorithm to find closer providers. The discovery algorithm takes the proximity information from the information table that corresponded to the suitable services. Table 1 describes the commonly used notations in this section.

Figure 2 outlines the pseudo-code of the service discovery algorithm. The inputs to the algorithm are the set of adaptation tasks T (i.e, T = (t_1, t_2,Y, t_n)), set of available services for each task S (i.e, S = (s_1, s_2,Y,s_m)) and set of landmark vector V for each service belongs to T.

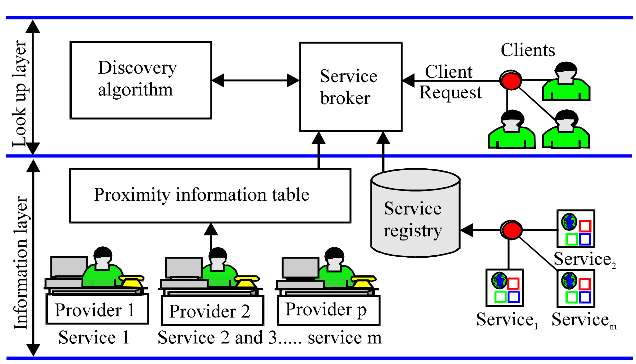

Fig. 1: Service discovery for service-oriented content adaptation

Table 1: List of notations

Notation	Description
S	Set of available services for the tasks T
T	Set of adaptation tasks
S_{ij}	Service j for task i
V	Landmark vector for each service s_{ij}
P	A path of composite services serving T
|cs2|	Network proximity between two nodes s_{ij} and si+1,j
|cs|	Summation network proximity for all residing pairs, sum |cs2|
min|cs|	Closest providers for a series of tasks

```
Algorithm 1: Service discovery
INPUT: T, S, V |cs2|
BEGIN
Construct composite services paths (Path: Tasks, Services, |cs2|)
FOR each path DO
Estimate |cs2| for all residing pairs using V
Calculate network proximity (|cs| = sum |cs2|)
END FOR
FOR each task DO
Select the service in the p shortest path that minimize |cs|
END FOR
END
```

Fig. 2: Service discovery algorithm

We assume the number of tasks and number of services for each task is known a priori.

Construction of paths: The initial step is to construct potential path containing k closest for all residing pairs. First, consider pairing between two consecutive tasks t_1 and t_2. The algorithm retrieves all available services for tasks t_1 and t_2 including landmark vector V for each service. Node pair between two consecutive tasks. For each pair of services s_{ij} and s_{i+1j} for tasks t_1 and t_2, network proximity is estimated. If there is sufficient number of landmarks/milestones that two services s_{ij} and s_{i+1j} measure against, it is very likely one of these landmarks is located on the shortest path between services s_{ij} and s_{i+1j}. Suppose this landmark is l. The summation of distance (s_{ij}, l) and distance (s_{i+1j}, l) should be minimal if the triangle inequality holds. It is worth noting that violations of triangular inequality reported by Wang *et al.* (2007) are less likely to happen as the intermediate routers are not included as points of the triangle (Sharma *et al.*, 2006). The equation is given as the followings:

$$\text{min}-\text{plus}\left(S_{ij}, S_{i+1j}\right) = \min_{\forall l \in L\left(s_{ij}, s_{i+1j}\right)} \left(\text{dist}\left(S_{ij}, l\right) + \text{dist}\left(S_{i+1j}, l\right)\right) \quad (1)$$

where, dist (S_{ij}, l) and dist (S_{i+1j}, l) are the network measurement from s_{ij} to l and s_{i+1j} to l, respectively. For example, in Fig. 3, the network proximity between s_{21} to s_{32} can estimated using landmark l_x.

Then, for a given service s_{ij} and candidate pair service set s_{i+1j}, $j \in J$, services s_{i+1j} can be ranked in the increasing order of the min-plus (). We can set the algorithm to return only k closest services s_{i+1j} from the ranking. For example, in Fig. 3, the network proximity between s_{21} to s_{31} and s_{21} to s_{32} can be estimated using min-plus (s_{21}, s_{31}) and min-plus (s_{21}, s_{32}), respectively. Then, we can rank distance of these two pairs in ascending order. The equation is given as the followings:

$$\text{Ranked min}-\text{plus}\left(S_{ij}, S_{i+1j}\right) = \min_{\forall j \in J, l \in L\left(S_{ij}, S_{i+1j}\right)} \left(\text{dist}\left(S_{ij}, l\right) + \text{dist}\left(S_{i+1j}, l\right)\right) \quad (2)$$

The above step is repeated for the next consecutive pairs of composite services for tasks i+1 and i+2. This will eventually give all possible paths containing k closest pairs for all tasks. For example, consider a content adaptation request containing three adaptation tasks. Each task can potentially be performed by five services: {t_1: s_{11}, s_{12}, s_{13}, s_{14}, s_{15}}, {t_2: s_{21}, s_{22}, s_{23}, s_{24}, s_{25}} and {t_3: s_{31}, s_{32}, s_{33}, s_{34}, s_{35}}. When estimation is performed for k equal to two, the closest two services returned for each service are given as Fig. 4b. On contrary, if brute force method is implemented, the number of possible paths is $5^3 = 125$, as in Fig. 4a. Specifically, each node in a task is potentially be connected to each node for the consecutive task(s).

As depicted in Fig. 4b, two closest services are returned for each service potentially serving tasks t_1 and t_2. If a service has no link (i.e., it does not have any predecessor service closer to it), it can be discarded. For instance, s_{25} is not a closer candidate from any service for task t_1. The number of potential paths generated using the proposed algorithm is 20. An example of a path is $s_{11} \rightarrow s_{22} \rightarrow s_{31}$. Figure 4a depicted the potentials paths using brute force method. As suggested by Fudzee and Abawajy (2011a) the total number of paths generated by brute force method is, where n is a number of service providers for each task and m is a number of tasks for a particular n. By comparing Fig. 4a and b, we can see the reduction ratio of paths achieved by the proposed method is 125-20 = 105. Then, for each path, we sum up the estimated network proximity for every residing pair. The summation equation is given as below:

$$\sum_{i=1, j=1}^{IJ} \text{min}-\text{plus}\left(S_{ij}, S_{i+1j}\right) \quad (3)$$

These paths can further be ranked based on the path estimated network proximity. We can further refine the algorithm to return p shortest paths. Now, only a small number of services for each task contained in p paths are returned. Figure 5 illustrates the ranked paths with the given estimated network proximity for every k = 2 closest pairs.

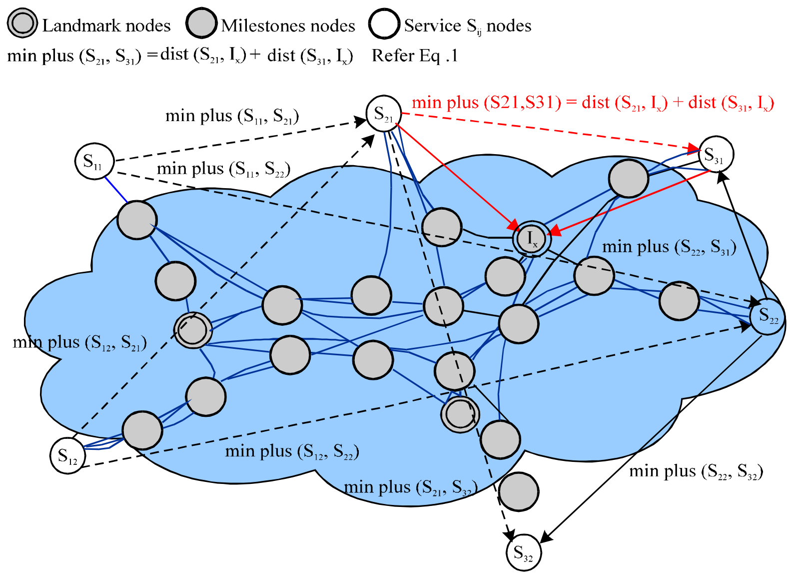

Fig. 3: Illustration of estimating network proximity

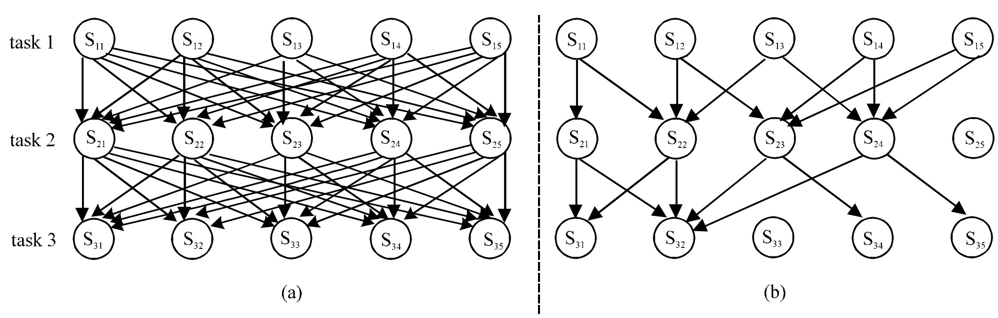

Fig. 4(a-b): (a) Potential paths using brute force methods and (b) Paths after k = 2 is returned

Ranked composite providers (top p paths)
Σprox. s_{15}, s_{24}, s_{32} = 18+20 = 38 ms
Σprox. s_{12}, s_{22}, s_{32} =18+21 = 39 ms
Σprox. s_{15}, s_{23}, s_{34} = 19+20 = 39 ms
Σprox. s_{15}, s_{24}, s_{35} = 18 21 = 39 ms

Fig. 5: Potential paths after k = 2 is returned

Finally, for each task, the broker selects s responsive and alive services from the top p paths. The reason behind selecting s services is because the top p paths are formed using estimation method. The broker can randomly select one of the s services. Alternatively, the broker can use QoS criteria to select the best possible service in a manner similar to Fudzee and Jemal (2010).

The proposed algorithm has several strengths. It improves the information about local services and select closer sets of composite service providers. It leverages partial mapping instead of mapping all services to a Cartesian space. Also, to check aliveness or reachability, it requires RTT (Round Trip Time) measurement to only a small number of services.

Analysis of algorithm:
Proposition 1: The maximum number of paths P (m) containing k services variation is bounded by Eq. 4. k is a number of closest service providers estimated for a particular task t_1, t_2, t_3,Y, t_{n-1} (excluding the last task t_n), m is the number of tasks having a particular k (excluding the last task t_n) and is the number of services of task t_1:

$$P(m)=\left(1^{m_i}\times\ldots\times(k-1)^{m_{k-i}}\times k^{m_k}\right)\times S^{t_i} \qquad (4)$$

Proof: First, using mathematical induction approach, we proof that the number of paths created at each service j of task t_1 i.e., $Pm(S_{ij})=\left(1^{m_i}\times\ldots\times(k-1)^{m_{k-i}}\times k^{m_k}\right)$ is true:

Basis: Product rule states that if a procedure is done by two tasks (let us say, k_1 and k_2 ways to do task 1 and 2, respectively), there are $k_1 \times k_2$ ways to do the procedure. Similarly by substitution, if a path initiated at is constructed by two links (let us say k_1 and k_2 ways to do link 1 and 2, respectively), then the number of paths initiated at is $k_1 \times k_2$ as well.

Initial step: Using the standard notation i.e., task: For any positive integer m, let P (m) be the product rule for m tasks. For the basis case, take m = 2 (this refer to product rule for two tasks). Now assume that P (m) is true. Consequently, P(0) = 0 is true.

Inductive step: Consider (m+1) tasks. t_1, t_2,Y., t_m, t_{m+1}, which can be done in k_1, k_2,Y, k_m, k_{m+1} ways, respectively. By the product rule of two tasks, the number of ways to do this is the product (multiplicity) of the number of ways to do m tasks, including k_{m+1}. By the inductive hypothesis this is $k_1 \times k_2 \times Y \times k_m \times k_{m+1}$, as desired.

Associate basis 1: If $k_1 = k_2$, $k_1 \times k_2 = k^2$ (in this way, we can group the same number of k together). Similarly, if $k_1 = k_2$, $k_1 \times k_2 \times k_m \times k_{m+1} = k^2 \times k_m \times k_{m+1}$ is true.

Associate basis 2: Next, if the number of paths initiated at each service of task t_l Pm (S_{lj}) is true, then the total number of paths initiated by services $S_{lj}, \forall j \in J$ is the result of multiplying P (S_{lj}) with J services available for task t_1.

Therefore:

$$P(m) = \left(1^{m_1} \times \ldots \times (k-1)^{m_{k-1}} \times k^{m_1}\right) \times S^{t_1}$$

holds.

Example: Suppose we have the following scenario:

Tasks	=	$\{t_1, t_2, t_3\}$
Services	=	$\{s_{11}, s_{12}, s_{13}, s_{14}, s_{21}, s_{22}, s_{231}, s_{24}, s_{31}, s_{32}, s_{33}, s_{34}, s_{35}, s_{36}\}$
Mapping	=	$\{t_1$: $s_{11}, s_{12}, s_{13}, s_{14}$; t_2: $s_{21}, s_{22}, s_{23}, s_{24}$; t_3: $s_{31}, s_{32}, s_{33}, s_{34}, s_{35}\}$
k for each pair	=	$\{t_1$ to t_2: 4; t_2 to t_3: 2$\}$

Here, we have 1 pair looking for k = 2 and 1 pair for k = 4. As per proposition 1, we have: $(1 \times 2^1 \times 3^0 \times 4^1) \times 4 = 32$ maximum paths.

Proposition 2: S^t (LT+kT-k) is the number of measurements required by the proposed algorithm.

Proof: Let S^{t_n} be the total number services available to perform a task $t_n \in T$ and $\{S^{t_1}, S^{t_2}, S^{t_3}, \ldots, S^{t_n}\} \in S$. Each service $s \in S^{t_n}$ conducts traceroute measurements to L landmarks with the total measurement of $S^{t_n} \times L$. If $\forall S^{t_n} \in S$ there is a constant number of s, the total traceroute measurements is $S^{t_n} \times L \times T$. If s for each S^{t_n} differs, the total traceroute measurements will be:

$$\sum_{n=1}^{N} S^{t_n} \times L$$

Then, each service for task t_n to t_{n-1} then performs k pingmeasurements with the total ping measurements of $S^{t_n} \times k$. If $\forall t_n \in T$ there is a constant number of k, the total ping measurements is $S^{t_n} \times k \times (T-1)$. If k for each t_n to t_{n-1} differs, the total ping measurements will be:

$$\sum_{n=1}^{n-1} S^{t_n} \times k^{t_n}$$

For general representation, assumes that there is a constant number of s and k. Thus:

$$(S^{t_n} \times L \times T) + (S^{t_n} \times k(T-1))$$

is the total number of measurements required. This equation can be simplified as:

$$S^{t_n}(LT + kT - k)$$

Example: Suppose we have the following scenario:

Tasks	=	$\{t_1, t_2, t_3\}$
Services	=	$\{s_{11}, s_{12}, s_{13}, s_{14}, s_{21}, s_{22}, s_{231}, s_{24}, s_{31}, s_{32}, s_{33}, s_{34}, s_{35}, s_{36}\}$
Mapping	=	$\{t_1$: $s_{11}, s_{12}, s_{13}, s_{14}$; t_2: $s_{21}, s_{22}, s_{23}, s_{24}$; t_3: $s_{31}, s_{32}, s_{33}, s_{34}, s_{35}\}$
k for each task	=	$\{t_1$: 4; t_2: 2; t_3: 2$\}$
Landmark	=	5

Here, we have 2 tasks looking for k = 2 closest services and 1 task for k = 4. As per proposition 2, we need: $(4 \times 5)+(4 \times 5)+(5 \times 5) = 65$ traceroute measurements and $(4 \times 4)+(4 \times 2) = 24$ ping measurements. The total of measurements needed is 65+24 = 89.

PERFORMANCE EVALUATION AND RESULTS

Simulation settings: We use simulation to study the efficiency of the service discovery execution against other methods, in term of getting closer service providers. Two performance metrics used are accuracy of getting closer providers and overhead in term of number of measurements required. Having closer providers reduce the total time to provide clients with adapted content versions. Also, it relatively reduces the amount of time required by clients with mobile devices to use wireless network. Having lower overhead improves the implementation practicality. We adopted the simulation and verification methodology described by Javadi *et al.* (2008).

To perform the simulation, we used data set provided by Gummadi *et al.* (2002). These data sets are used as input for the accuracy study for all approaches. We randomly assigned different roles for each node in the dataset. The random roles are landmark, milestone, services with varying adaptation functions and service broker. The number of nodes assigned to each role followed the specified setting in each simulation. To estimate the network proximity between composite services in a given path, the proposed algorithm utilizing min-plus equation is used. Then, we create the partial topology using the tool available at: http://topology.eecs.umich.edu/inet/.

Five different simulations were conducted to study the path accuracy metric towards (1) Number of tasks, (2) Number of services, (3) Number of nodes, (4) Number of k and (5) Number of p paths. These variations are chosen to evaluate the stability and scalability. At each run, we generated the number of adaptation tasks (T) to be between 2 and 5. We set the number of available service providers (S) for a particular task in the range of 5-20. The total number of nodes is varied from 100-1000. We set the number of k and p to be between 2-9 and 3-10, respectively. The number of landmarks is set to be constant at 5, as suggested by Sharma *et al.* (2006). The values we used for each parameter are in line with the current literature (Sharma *et al.*, 2006) and also reflect the actual environment.

We used two others algorithms (max diff and inner product) as the baseline to estimate the closest k pairs that construct p shortest paths. We chose these algorithms as it is widely accepted and is the closest to our algorithm. Similar to min-plus, max diff algorithm assumes that there is a large likelihood that there exists a landmark l such that s_{ij} is on the shortest path from s_{i+1j} to l or s_{i+1j} is on the shortest path between s_{ij} to l. The equation for max diff algorithm is given as follow:

$$\text{max diff}\left(S_{ij}, S_{i+1j}\right) = \max_{\forall j\in J, l\in L\left(S_{ij}, S_{i+1j}\right)} \text{ABS}\left(\text{dist}\left(S_{ij}, 1\right) - \text{dist}\left(S_{i+1j}, 1\right)\right) \tag{5}$$

Inner product is widely used in information retrieval for document ranking which similarly can be used for network proximity ranking. The equation of inner product is given as follow:

$$\text{Innerproduct}\left(S_{ij}, S_{i+1j}\right) = \text{Max}_{\forall j\in J, \sum l\in L\left(S_{ij}, S_{i+1j}\right)} \text{ABS}\left((r/(\text{dist}(S_{ij},1)^2)) \times (r/(\text{dist}(S_{ij},1)^2))\right) \tag{6}$$

We use r = 1 and this is in line with Sharma *et al.* (2006). Accuracy metric measures whether the actual shortest paths p' was returned in the proximity set p. For this purpose, the actual shortest paths p' were measured. The accuracy is given as the following:

$$a(i) = \begin{cases} 1 \text{ if } p' \in p \\ 0 \text{ if otherwise} \end{cases} \tag{7}$$

Then, mean accuracy is used to evaluate the accuracy for estimated shortest paths. It is given as the following:

$$acc_p = \frac{(\sum_{i=1}^{p} a(i))}{P} \tag{8}$$

For overhead metric, we compare the proposed algorithm with the brute force method. The brute force method is used to obtain 100% accuracy. We compute the overhead reduction in term of measurements required for the proposed algorithm against the brute force method using the following equation:

$$or = \frac{\text{min plus overhead}}{\text{bruteforce overhead}} \tag{9}$$

Then, we compute the average reduction ratio, defined as follows:

$$arr = \frac{\sum_1^c (or)}{\text{counter}, c} \tag{10}$$

where, c is the number of overhead reduction being considered.

RESULTS AND DISCUSSION

Table 2 presents the summary of performance metrics i.e., accuracy and overhead. The accuracy of brute force method is the highest as it relies on actual network proximity. This is similarly justified by Fonseca *et al.* (2005) and Comer (2006). Other approaches (proposed, max diff and inner product algorithms) are estimation based. The main drawbacks of brute force are the measurement overhead (i.e., number of PINGs required). If the number of services S for each task t_1, t_2, t_3,..., t_n is

Table 2: Performance summary

Performance metrics	Accuracy (average%)					Overhead (average No.) Measurement required				
Approaches	Tasks	Services	Nodes	k	p	Tasks	Services	Nodes	k	p
Proposed (Min-plus) algo.	85	81	85	83	78	250	263	210	254	210
Max diff algorithm	76	71	76	76	69	250	263	210	254	210
Inner product algorithm	63	60	63	64	58	250	263	210	254	210
Brute force	100	100	100	100	100	500	750	400	400	400

constant, the brute force method to obtain 100% accuracy and precision would be to conduct as discussed by Sharma *et al.* (2006). This renders brute force method impractical and expensive especially for Internet-scale deployments. For estimation based algorithms, the measurement overhead is identical.

Among simulated estimation based algorithms, the proposed min-plus algorithm outperforms the others in every variation, if and only if triange inequality holds (Wang *et al.*, 2007). In term of overhead, each estimation based algorithm requires equal number of network measurements as proposed by Gummadi *et al.* (2002). Also, some key findings were observed. First, the accuracy for all algorithms is relatively increased for task, k and p variations along x-axis. Second, there is a very slight accuracy decrement for all algorithms for service and node variations along x-axis. Third, the proposed min-plus algorithm outperforms others in every variation (tasks, services, nodes, k and p) of the simulations, with the exception of k variations when $k>7$. That is, the min-plus algorithm is more accurate for k below than 7 and this is a practical limit for k. Higher k value increases measurement overhead and potential paths. This result is in line with the study by Sharma *et al.* (2006) and we are able to confirm their finding.

Taken as a whole, the proposed algorithm reduces the overhead up to one magnitude if compared to brute force and provides high accuracy of getting closer service providers (i.e., more than 80% in average) if compared to other estimation based algorithms (i.e., less than 80% in average). This would not change much even if the network estimation method proposed by Dabek *et al.* (2004) is used as alternatively to the min-plus algorithm due to the same network estimation foundation. Also, we are interested to analyze the finding on accuracy if triangle inequality is violated, as our current comparison does not take this into account. This will enable us to verify Wang *et al.* (2007) finding.

CONCLUSION

In this study, we proposed a distance-sensitive service discovery mechanism for the service-oriented content adaptation platform. To the best of our knowledge, most (if not all) of the service discovery mechanisms for service-oriented content adaptation did not take network proximity into account. As such, the series of tasks may traverse distant composite providers to be served. This increases the accumulated time to achieve adapted content version. The proposed discovery mechanism is able to perform discovery of closer composite providers. It reduces the measurement overhead up to one magnitude if compared to the brute force method. In term of accuracy, the proposed algorithm outperforms the other simulated estimation algorithms. The proposed architecture is practical to be deployed for the Internet-scale. In future, we plan to study on how to incorporate QoS as the selection criteria. As the proposed algorithm generates top p paths base on estimation, the final best possible service can be determined using QoS criteria. In this study however, we refrain ourselves from the triangle inequality violation.

ACKNOWLEDGMENT

The authors would like to thank Universiti Tun Hussein Onn Malaysia (UTHM) for providing the research facilities and supporting research group activities in accomplishing this research study.

REFERENCES

Campo, J.M., J.L. Gazquez, S.B. Trickey and A. Vela, 2012. A new meta-GGA exchange functional based on an improved constraint-based GGA. Chem. Phys. Lett., 543: 179-183.

Comer, D., 2006. Internetworking with TCP/IP: Principles, Protocols and Architecture. Prentice Hall Press, New Jersey, ISBN: 978-0134685052.

Dabek, F., R. Cox, F. Kaashoek and R. Morris, 2004. Vivaldi: A decentralized network coordinate system. Comput. Commun. Rev., 34: 15-26.

Dastjerdi, A., S. Tabatabaei and R. Buyya, 2010. An effective architecture for automated appliance management system applying ontology-based cloud discovery. Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, May 17-20, 2010, Melbourne, Australia, pp: 104-112.

Fawaz, Y., G. Berhe, L. Brunie, V.M. Scuturici and D. Coquil, 2008. Efficient execution of service composition for content adaptation in pervasive computing. Int. J. Digital Multimedia Broadcast. 10.1155/2008/851628

Fonseca, R., P. Sharma, S. Banerjee, S. Lee and S. Basu, 2005. Distributed querying of Internet distance information. Proceedings of the IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies, March 13-17, 2005, New York, pp: 2876-2881.

Fudzee, M.F.M. and D.M.M. Jemal, 2010. Multi criteria content adaptation service selection broker. Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, May 17-20, 2010, Melbourne, Australia, pp: 721-726.

Fudzee, M.F. and J. Abawajy, 2011a. A protocol for discovering content adaptation services. Proceedings of the 11th International Conference on Algorithms and Architectures for Parallel Processing, October 24-26, 2011, Melbourne, Australia, pp: 235-244.

Fudzee, M.F.M. and J.H. Abawajy, 2011b. QoS-based adaptation service selection broker. Future Generat. Comput. Syst., 27: 256-264.

Gummadi, K., S. Saroiu and S. Gribble, 2002. King: Estimating latency between arbitrary Internet end hosts. Proceedings of the ACM SIGCOMM Workshop on Internet Measurement, November 6-8, 2002, New York, pp: 5-18.

Javadi, B., J.H. Abawajy and M.K. Akbari, 2008. An analytical model of interconnection networks for multi-cluster computing systems. Int. J. Concurrency Comput. Pract. Exp., 20: 75-97.

Kritikos, K. and D. Plexousakis, 2009a. Mixed-integer programming for QoS-based web service matchmaking. IEEE Trans. Serv. Comput., 2: 122-139.

Kritikos, K. and D. Plexousakis, 2009b. Requirements for QoS-based web service description and discovery. IEEE Tran. Services Comput., 2: 320-337.

Li, X., N. Santoro and I. Stojmenovic, 2009. Localized distance-sensitive service discovery in wireless sensor and actor network. IEEE Trans. Comput., 58: 1275-1288.

Meditskos, G. and N. Bassiliades, 2010. Structural and role-oriented web service discovery with taxonomies in OWL-S. IEEE Trans. Knowl. Data Eng., 22: 278-290.

Mohan, R., J.R. Smith and C.S. Li, 1999. Adapting multimedia Internet content for universal access. IEEE Trans. Multimedia, 1: 104-114.

Plebani, P. and B. Pernici, 2009. URBE: Web service retrieval based on similarity evaluation. IEEE Trans. Knowl. Data Eng., 21: 1629-1642.

Shahidi, M., A. Attou and H. Aghvami, 2008. Content adaptation: Requirements and architecture. Proceedings of the 10th International Conference on Information Integration and Web-based Applications and Services, November 24-26, 2008, New York, pp: 626-629.

Sharma, P., Z. Xu, S. Banerjee and S.J. Lee, 2006. Estimating network proximity and latency. ACM SIGCOMM Comput. Commun. Rev., 36: 39-50.

Song, X., W. Dou and J. Chen, 2011. A workflow framework for intelligent service composition. Future Gener. Comput. Syst., 27: 627-636.

Wang, G., B. Zhang and T. Eugene, 2007. Towards network triangle inequality violation aware distributed systems. Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, October 24-26, 2007, New York, pp: 175-187.

Zaman, N. and A. Abdullah, 2011. Different techniques towards enhancing Wireless Sensor Network (WSN) routing energy efficiency and Quality of Service (QoS). World Applied Sci. J., 13: 798-805.

Zhou, C., L.T. Chia and B.S. Lee, 2005. Semantics in service discovery and QoS measurement. IT Prof., 7: 29-34.