MODEL-BASED SERVICE SELECTION FOR RELIABLE SERVICE ACCESS IN MANET

Omid Bushehrian

Dept. of Computer Eng. and IT, Shiraz University of Technology
Shiraz, Iran

Email: bushehrian@sutech.ac.ir

ABSTRACT

Due to the dynamic infrastructure in Mobile Ad-hoc Networks (MANET), supporting a reliable and timely service access is one of the main requirements of applications deployed over MANETs. In this paper, a reliability-aware service selection scheme is proposed in which a client node collects and ranks the available service providers based on their estimated response time and reliability. However, estimating the reliability of a service provider in MANET is a challenging issue because it is strongly dependent on the current conditions of the network (dynamic or stable). To estimate the network state based on the local observations, Hidden Markov Model (HMM) is applied prior to the service selection.. The experimental results show that by using our method, not only the unreliable service providers are filtered but also, comparing to the existing service selection methods, lower violations of the client response time requirements (defined in the client SLA) are observed.

Keywords: MANET, Reliable Service Access, Hidden Markov Model, Service selection

1. INTRODUCTION

A Mobile Ad-hoc Network(MANET) is a self-organizing collection of wireless mobile nodes where the communications of nodes are performed without any fixed infrastructure. The most accepted architecture for application development for MANET is Service-Oriented Architecture(SOA) i.e. the mobile nodes provide/consume services to/from other nodes [1][5][6]. However, satisfying the QoS requirements in service access is a challenging issue.

Service discovery and *service selection* are two main procedures which have to be executed by a client reactively or by the network proactively to enable the service access. Since a service might be provided by several nodes in the network with different qualities, the service discovery procedure is needed to collect the list of all available service providers. Once the service list is collected, a selection procedure is used to pick one service provider considering some QoS criteria.

One of the important challenges in service discovery and assignment over dynamic networks is the successive service disconnections which is very problematic in critical applications with firm reliability requirements [3]. Each disconnection entails executing the service discovery procedure again, hence resulting in delay and packet broadcast in the network. Furthermore, the response time of the service provider is another important factor for the clients during the service access. For instance, a robot in a mine field whose planner module is failed needs to use the planner service of other robots in the network. Obviously, this robot needs a timely and reliable planner service to accomplish its mission.

One of the main sources of unreliability in service access over MANET is the *network failure*. However the node mobility itself is not considered as the cause of network failure, rather, the changes in the network

topology is the main source of this kind of failure. The topology change happens when the neighbour nodes of a node change, hence the previously established network routes are not valid anymore. Therefore, it is possible that there is mobility in the network while the node neighboursremain the same over time (no topology change happens). For instance, consider a set of robots moving in a field such that their relative positions are not changing.Obviously in a changeable network, service providing is less reliable in comparison to a stable one. In many MANETs applications such as rescue or discovery missions where mobile nodes team up to accomplish a mission, the topology of the mobile nodes in the network is usually affected by the conditions of the environment in which the mobile nodes are movingand it is desirable that each node isable to estimate the current network conditions (stable or changeable) by observing some local parameters. By knowing the current network condition, each node will be able to select a service provider among available ones based on its QoS requirements more intelligently. For instance, in a changeable network condition, a node never selects a distant service provider if a reliable service access is required and prefers the service providers located in its neighbourhood.

In previous studies, many works have been done on the service discovery methods [9][11][12][13]. Service discovery can be achieved either in application layer or in the network layer. Designing the service discovery in the network layer, called *cross-layer design*, can reduce the packet broadcasts in the network significantly [11][13]. There are also a few studies on the service selection algorithms based on both reliability and performance [9][12], in which the reliability estimation methods are merely based on the service provider distance to the requester node (measured in terms of hop counts). However, we believe that the network conditions (dynamic or stable) has a direct impact on the reliability of service providers, i.e. in a relatively stable network (in which the nodes rarely go beyond each other's signal coverage), the reliability of a distant service provider is the same as the reliability of a close one assuming that other sources of service unavailability such as intermediate node failures or service failures are negligible.This assumption is valid and realistic in many applications. For example, in a network of moving robots with powerful (or solar rechargeable) batteries, working together in a field with normal conditions, the intermediate node failure is very low since the main sources of node failure in MANET are energy limitations or leaving the network voluntarily[15]

In this paper, a model-based method for reactive service selection over MANET's is presented which supports both reliability and performance of the service delivery. The reliability is measured in terms of the number of successive disconnections during the service access perceived at the client side and the performance is measured in terms of the service response time required by the client.

The objectives of our research has been twofold: (1) like previous reliable service selection methods, the unreliable service providers should be filtered during the service selection; and (2) the violations of client response time requirements during the service access should also become minimal. To achieve these goals, we have proposed an adaptive method for ranking the service providers based on the current state of the network. In this method, each node first estimates the current state of the network based on its local observations using Hidden Markov Model (HMM) [7], and then ranks the service providers according to their reliability and response time. For example, in our adaptive selection method, in a stable network conditions, a fast but distant (in term of hop counts) service provider is not penalized due to its distance to the client and hence can compete with the service providers which are closer to the client node during the selection procedure.

The remainder parts of this paper are organized as follows: In Section 2, the related works are explained. The proposed model-based method is presented in Section 3. The experimental results are presented in Section 4 and finally, Section 5 concludes the paper and gives the future research directions.

2. RELATED WORKS

Previously, Hobeke et al., [2] have studied two service discovery schemes. In Directory-less and resource discovery mechanism, nodes reactively request services when needed and/or nodes proactively announce their services to others, which is an attractive approach for infrastructure less networks. The alternative scheme is the directory-based one and involves directory agents where services are registered and service requests are handled.

Cross-layer design of service discovery protocols has been studied in many papers. For instance,

Halonen&Ojala[1]have integrated the SOA service discovery into the ad-hoc routing protocol. A prototype implementation based on the optimized link state routing (OLSR) protocol is also presented. There is no central element in this approach and the service discovery is fully distributed on a mesh network. Since there is no central registry to collect the available service descriptions, there is no need to actually both publish and discover the services. These operations can actually be considered in parallel with ad hoc routing protocols, which are either proactive or on-demand.

In Li's work [4], a distributed and adaptive algorithm based on a new QoS parameter, called service efficiency, is presented. The aim is to guarantee the service availability with the lowest resource cost in MANET by dividing nodes into groups and dynamically creating and terminating service instances in each of the groups. However, dynamic service creation is not possible for all kinds of deployed services.

Riva et al., [10] proposed a novel model of service provisioning in ad hoc networks based on the concept of context aware migratory services. Unlike a regular service that executes always on the same node, a migratory service can move to different nodes in the network in order to accomplish its task. The migration is triggered by changes of the operating context, and it occurs transparently to the client application. The main shortcoming of this method is that not all services can be easily migrated to other nodes due to lack of required resources on the target node.

The problem of seamless hand-off to a new service provider when the original service provider becomes unavailable is studied thoroughly in [8]. In the presented model, the entire network is divided into domains and in each domain K nodes with higher capability are elected as directory nodes and the best of them are appointed as the domain elector. All the service provider nodes then find the nearest directory in their domain and register their services with the directory. Service handoff is triggered if a disconnection takes place between the service provider and the user. The new service provider which is least loaded is selected among all the registered providers in the directory. However, in this study the reliability of the providers are not considered in the service selection procedure. Furthermore, using the service directories and elector nodes, results in complex hand-off protocols with too many message types.

In [3], an adaptive service discovery protocol that enhances the performance of service discovery is proposed. Their main focus is to use an adaptive core node election mechanism that changes whenever the load increases and is also robust against network failures. The selection algorithm here is based on both distance and service capability of the provider.

In [13], a light-weight reliable service discovery based on the notion of Service Magnetic Field (SMF) is presented. Adopting a cross-layer design, a requester node always selects a provider with the strongest felt SMF among all the available providers. The closer (in terms of hop count) a service provider is to the requester node, the stronger SMF is perceived by the requester.

In [9], a cross-layer designed reliable service selection protocol which guarantees both timelines and energy efficiency is proposed. The main idea is to incorporate the intermediate nodes capacity in reliability estimation of a multi-hop link to a service provider. In addition to the link reliability, load of the service provider is also another important factor in service selection. However, the effect of node movement pattern is not considered in this work.

In [12] link expiration time and energy are two factors used to select the most reliable service provider. The link expiration time between A and B is computed using a location based method in which the amount of time for A and B to go beyond each other's radio range is computed based on their relative velocity. However, the method is based on the assumptions that all the node clocks are synchronized and the nodes velocity are not variable over the time.

We believe that in addition to the hop-count and intermediate node energy which were considered in the previous works as the main factors affecting the reliability of a service provider, the general network conditions (dynamic or stable topology, stable or faulty service providing) are also very important. When the network topology is stable i.e. the nodes rarely go out of each other's signal coverage and service failure rate is also low, the hop count is not an important factor for service selection due to the fact that in such conditions a distant service provider is almost as reliable as the close one and hence other factors such as provider load (or response time) have to be used for service selection. Likewise, in different

network conditions, different factors have most impact on the service selection. Hence, the idea is to estimate the future network state first and then to apply an appropriate service selection criterion based on that.

In this paper, we assume that there are two main sources of service unavailability: 1) *Network failure* due to the dynamic topology in which the nodes go out of each other's signal coverage frequently, and *2) Service failure* which happens due to either lack of energy or other faults during the service executions. For simplicity in this paper we ignored *Node failure* probabilities in which a node goes off due to lack of energy. Furthermore, it is assumed that each client requires a service in long term and sends requests to the service provider consecutively. The response time requirement of each client is specified in its SLA (Service Level Agreement) as well.

3. The MODEL-BASED METHOD

In our proposed method, each node may host a *client process*, a *service process* and a management agent process called *Service Manager(SM)* which is responsible for executing the service discovery and service selection protocol (see Fig. 1). The intermediate nodes which are neither a service consumer nor a service provider must have the SM executing on them too in order to contribute in service discovery and selection protocol.

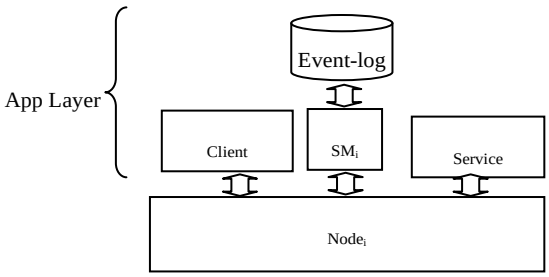

Fig. 1.The architecture of a sample node in our method

A client process triggers the service selection protocol by sending a *ServQuery* message to its local SM asking for a service provider. The SM executes the protocol as shown in Fig.2 and returns the selected service provider identifier back to the local client process using a *ServSuggestion* message. The node which initiates the protocol is called the *root* node.

Each *ServReply* message received by the SM process of node *j*, contains a collection of triples *(id,rt,hpc).* Each tripleassociates a service provider node identifier (*id*), with its estimated response time *rt* and its distance to the current node *j*, in terms of hop-counts, denoted by *hpc*.The estimated response time of a service process, s_i, executed on a node, n_j, is computed by the SM process of the same node using the following formula:

$$rt_{ij} = L_{ij} \times D_{ij} \qquad (1)$$

Where $L_{i,j}$ is the queue length behind s_i and $D_{i,j}$ is the estimated execution time of s_i on n_j(service demand of s_i).

The *EstimateState()* functionin Fig.2 is used to estimate the current state of the MANET using Hidden Markov Model. State estimation is performed based on the recent network events recorded in the *Event-log* by SM. The *Add2EventLog()*function is invoked by the SM to record two kinds of events: 1) When the local client process requests for a service provider by sending the *ServQuery* message to the SM; and 2) When any change is observed in the routing table entries of the local node.

The *SelectProvider()* function is used to rank the service providers collected at the root node, based on their response time, hop-count and last TTF(Time To Failure) considering the current state of the network.

3.1. **State estimation using HMM**

In our proposed method, the service selection is performed depending on the current state of the network. In some previous works, a two state network, "normal" and "congested", is defined [14]. However, in this work, the network state is defined as a combination of the topology conditions and the service availability conditions (in application layer). Therefore, the network state is defined as the pair: *(topology, serviceAvailability)*. The *topology* is either 'stable' or 'dynamic' and the *serviceAvailability* is either 'stable' or 'faulty'. Faulty service availability means that most of the service processes in the network frequently fail or stop due to lack of energy or limited resources. Therefore, at any given time, the network may reside in any of the following four possible states: *(Stable, Stable), (Stable, Faulty), (Dynamic, Stable), (Dynamic, Faulty)*. To estimate the current state of the system, each node uses HMM to compute the most probable system state (out of the four states) based on its local observations. Here the *'Client Disconnection Rate'(CDR)* andthe *'Route Change Rate' (RCR)* are observed periodically by the SM. Therefore, four independent observations are possible as listed in Table 1.

High *CDR* value is an indication of general service unavailability in the network. However, to distinguish between unavailability due to dynamic topology and unavailability due to faulty service providing, the second observation (RCR) is used. Obviously high RCR value is an indication of dynamic network topology.

```
Service Manager Algorithm
while (true){

currentState=EstimateState();
    p=NextPacket();
switch (p.ProtocolName){
case "ServQuery":
      Create a ServReQmessage and broadcast it.
Add2EventLog(time_stamp, p.serviceName,"ServQuery");
case "ServReq"   :
      Discard if repetitive message, store the sender ID as its parent
and broadcast the message, wait for
theServReply message.
case"ServReply":
If the local node is the root node:
selected=SelectProvider(p,currentState);
create a ServSuggestion message and send back to the client
process
else
       After a timeout, mergeall the receivedServReply messages into
onemessage. If the local node also provides the requested
service with the estimated response time rt, append
pair (id,rt,hpc) of the local node to the newly created
ServReply message. Send the message back to its parent.
  }//switch
If a change is observed in the local routing table
Add2EventLog(time_stamp, "RouteFault");
}
```

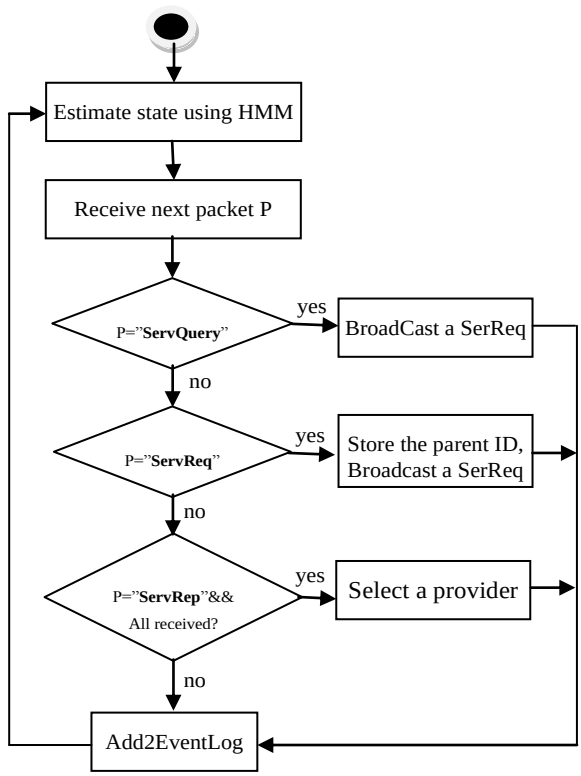

Fig. 2.Service Manager Algorithm(right) and its pseudo-code (left)

The *EstimateState()* function in Fig. 2, estimates the most probable state of the network using the *forward-backward* algorithm [7] assuming that the transition matrix T (where T_{ij} is the probability of transiting

from state *i* to state *j*) and the event matrix *B* (where B_{ij} is the probability of observing event *j* given a particular state *i*) are known.

In our experiments (discussed in the next section), we have chosen the values of B_{ij} based on the semantic relationship between state *i* and event *j* in the network. The probabilities of state changes in the transition matrix *T* in each experiment have also been defined based on the type of network used for that experiment in our MANET emulator.

Table1. List of possible observations

Observation	*CDR*	*RCR*
1	Low	Low
2	Low	High
3	High	Low
4	High	High

3.2. Service selection policy

Service providers are normally ranked according to their response time.However, two penalty factors are defined here to penalize a service provider due to its unreliability: 1) The distance to the client, measured in terms of hop-counts; and 2) the last TTF of the service provider computed by the SM using its local Event-log. The former is used to filter service providers with potential unavailability due to the *network failure* and the latter is used to filter the service providers with potential unavailability due to the *service failure.* However, depending on the current network state, the former, the latter or both penalty factors might be applied. The general ranking function is as follows:

$$rank(s) = rt_s + \alpha \times \frac{hc_s}{N} + \beta \times \frac{1}{ttf_s \times M} \quad (2)$$

Table2. Term Definitions

Term	***Definition***
P	set of all service providers discovered by SM
N	Number of network nodes
rt_s	Response time of service provider s ϵ P
hc_s	Hop-count of service provider s ϵ P
ttf_s	the last Time To Failure(TTF) value of s recorded in the local Event-log
M	Max(ttf_s) for all s ϵ P
α,β	penalty factor selectors α,β ϵ {0,1}

For example, when the network topology is stable, all the service providers regardless of their hop-counts are equally reliable (due to the fact that the probability of intermediate node failures are assumed negligible),hence, the first penalty factor is excluded from formula (2) by choosing α=0. The same justification is applied for choosing (or not choosing) the second penalty factor.

The definition of terms in formula (2) is presented in Table 2. Note that, if there is no previous record for service provider *s* in the Event-log, the value of ttf_s is set to *M*. The value of penalty factor selector α and β are determined based on the current network state as listed in Table 3.

Table3. Penalty factor selectors

Network State	
(Stable, Stable)	α=0, β=0
(Stable, Faulty)	α=0, β=1
(Dynamic, Stable)	α=1, β=0
(Dynamic, Faulty))	α=1, β=1

4. IMPLEMENTATION AND EXPERIMENTAL RESULTS

To evaluate different service discovery and selection protocols,we implemented a MANET emulator in Java [17]. This emulator enables designers to define different aspects of service providing in MANET such as: service to nodes assignments, service execution times on each node (in terms of logical clock ticks), power consumption scheme, radio coverage radius of nodes and *change policy* of the network. The latter is used to define different node movement and service failure patterns for the network along with the probabilities of switching from one pattern to another during the simulation. Each node in this emulator is implemented as a separate Java thread and communicates with *SM thread*, *service process* thread and *client process* thread using *virtual channels*. The emulator uses a soft clock to control the execution speed of the emulation. The clock period is set before starting the emulation using the emulator UI. In our experiments, the clock period was chosen as 500ms. The speed of packet delivery, client nodes, service provider nodes are all dependent on the clock period value in the emulator. The client process is modelled as a state machine with four states as shown in Fig.3.

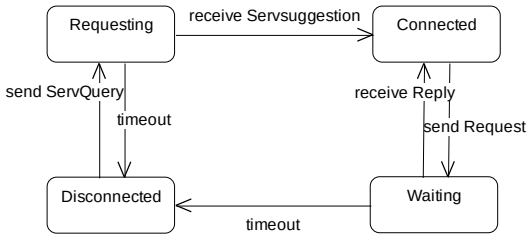

Fig.3.The client process states

Initially the client resides in the "Disconnected" state, once a service is required, a "servQuery" message is sent and the client transits to the "Requesting" state. Upon receiving a "servSuggestion" message which contains the selected service provider, the client transits to the connected state and subsequently starts sending and receiving application level messages to/from the service provider. If the reply is not received from the service provider (timeout), the client goes to the disconnected state.

The proposed service selection protocol was implemented and plugged into the emulator as a Java class which implements a standard interface defined for selection algorithms in the emulator.

4.1. Simulation Parameters

The HMM event matrix, *B*, is shown in Table 4. Events recorded in the local Even-log during the past *w* clock ticks (the observation window) are observed by the SM. When the number of recorded '*ServQuery'* events in the local Event-log, within the observation window, is higher than *cdr_threshold*, the CDR value is assumed *High* otherwise it is *Low*. Likewise, when the number of recorded '*RouteFault*' events in the local Event-log, within the observation window, is higher than *rcr_threshold*, the *RCR* value is assumed *High* otherwise it is *Low*.According to HMM principles, at each time instant t, the system determines the observation O_t by measuring the values of CDR and RCR at that time regardless of the values of metrics at previous times.

Table 4. The HMM event matrix(B)

	Observations: (CDR,RCR)			
Network States	(Low,Low)	(Low,High)	(High,Low)	(High,High)
(Stable, Stable)	99%	1%	0%	0%
(Stable, Faulty)	1%	0%	99%	0%
(Dynamic, Stable)	0%	1%	0%	99%
(Dynamic, Faulty)	0%	0%	1%	99%

Another simulation parameter is the *Network Service Density (NSD)* defined as the number of service providers in the network, $|P|$, divided by the number of network nodes, N:

$$NSD = \frac{|P|}{N} \quad (3)$$

Table 5. The HMM transition matrix(T) for "stable" network type

Network States	(Stable, Stable)	(Stable, Faulty)	(Dynamic, Stable)	(Dynamic, Faulty)
(Stable, Stable)	90%	3.33%	3.33%	3.33%
(Stable, Faulty)	90%	3.33%	3.33%	3.33%
(Dynamic, Stable)	90%	3.33%	3.33%	3.33%
(Dynamic, Faulty)	90%	3.33%	3.33%	3.33%

Table 6. The HMM transition matrix(T) for "dynamic" network type

Network States	(Stable, Stable)	(Stable, Faulty)	(Dynamic, Stable)	(Dynamic, Faulty)
(Stable, Stable)	5%	5%	40%	50%
(Stable, Faulty)	5%	5%	40%	50%
(Dynamic, Stable)	5%	5%	50%	40%
(Dynamic, Faulty)	5%	5%	40%	50%

Three network types with different state change policies were defined as test beds to perform the experiments: *Stable*,*Dynamic* and *Changeable*. It is possible to configure the emulator to emulate the behaviour of each network type. In a *Stable* network type, the most probable network state (out of the four previously defined states) is (Stable, Stable). In a *Dynamic* network type, the most probable states are (Dynamic, Stable) and (Dynamic, Faulty). However, in a *Changeable* network type, all the network states are visited during the simulation with the same chance. Once the emulator changes the current state of the network to a new state, the movement pattern of the network nodes and the service failure rates are changed as well. For example, if a transition happens from (Stable, Stable) state to the (Dynamic, Faulty) state by the emulator, the movement pattern of the nodes will also be changed to *"random walk"* and the failure rate of the service providers will be increased significantly. The transition matrix T of the HMM is defined according to the above mentioned network types. For example, for a *Stable* network type, the transition matrix is defined as presented in Table 5. However, if we want to use our method in real environments, the HMM has to be trained using a training sequence of observations which is not necessary in our experiments since the transition probabilities are already defined within emulator by the researcher.

Table 7. The HMM transition matrix(T) for "changeable" network type

Network States	(Stable, Stable)	(Stable, Faulty)	(Dynamic, Stable)	(Dynamic, Faulty)
(Stable, Stable)	25%	25%	25%	25%
(Stable, Faulty)	25%	25%	25%	25%
(Dynamic, Stable)	25%	25%	25%	25%
(Dynamic, Faulty)	25%	25%	25%	25%

4.2. Experiments

4.2.1. State estimation accuracy

The objective of the first experiment was to evaluate the accuracy of the state estimation procedure executed by the SM using the proposed HMM. The simulation was performed two times: firstly, with a *Changeable* network and low service density (NSD=0.2); and secondly, with a *Dynamic* network and high service density (NSD≈1). In both scenarios, the service distribution was random. The simulation parameters are as follows: N=30, simulation length=10000 clock ticks, w=100, cdr_threshold=5 and rcr_threshold=3.

The values of *cdr_threshold* and *rcr_threshold* were chosen as the lower bounds of the number of disconnections and the number of *"RouteFaults"* (in previous w ticks) respectively, recorded for clients when the network state is changed to "dynamic" by the emulator. In the real world implementations, the choice of these values is completely dependent on the interpretation of qualitative terms "High" and "Low" in HMM event matrix (see Table 4) from the client nodes point of view in the environment. In the applications with strict QoS requirements the threshold values should be chosen lower.

As shown in Fig. 4, in a low service density network, the current network state is detected by the SM with a short delay after a global state change happens. This delay is due to the fact that the client needs enough observations before it can detect the new state, hence, enough disconnection events have to be recorded in the client Event-log before the state detection. The later the service disconnection happens, the longer the state detection delays will be. The delay in the state change at the client side is very useful since it prevents the fast (and wrong) state changes due to the transient congestions in the network. The simulation was repeated in a high service density network and the longer state detection delays were observed (see Fig. 5). Here, due to the higher number of available service providers, it is very likely that the SM always finds a fast service provider in its neighbourhood (or a few hops away). Obviously, the connection to an adjacent service provider is more stable despite the fact that overall network condition is not stable. As a result, the prevalent local observation in the client side is observation #1 (listed in Table 1) which is most likely observation which happens in state *(stable, stable)*. Therefore, in a network with a high service density the most likely state estimated by clients is the *(stable, stable)* and it takes more time (in comparison with low service density) that the disconnection rate (recorded in the Event-log) exceeds the observation thresholds and hence the state detection happens with longer delay. Choosing the value of the window size (w) too large or too small may lower the accuracy of the state estimation by the client. Large window size values may result in longer delays in state change while small ones may result in more sensitivity of the algorithm to the transient conditions in the network. We repeated the above experiments with different window sizes (not reported in this paper), however, the best results were obtained when*w* equal to 1% of the simulation length (*w=100*).

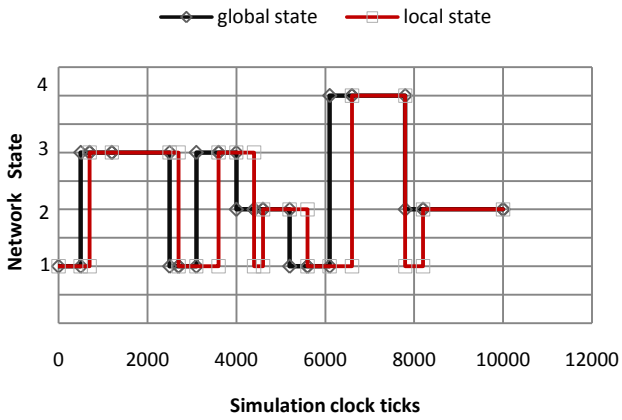

Fig.4.State estimation in a *Changeable* network with low service density. Network states are numbered from 1 to 4: 1:(stable,stable), 2:(stable,faulty), 3:(dynamic,stable), 4:(dynamic, faulty)

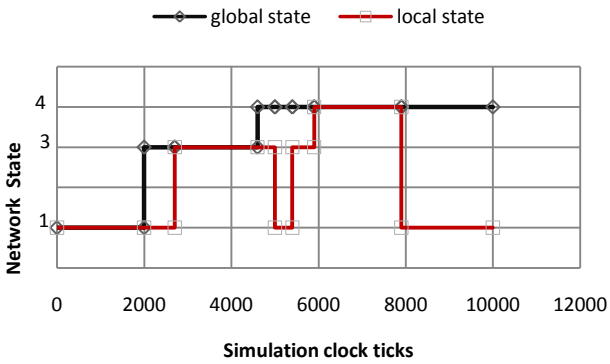

Fig.5.State estimation in a *Dynamic* network with high service density. Network states are numbered from 1 to 4: 1:(stable,stable), 2:(stable,faulty), 3:(dynamic,stable), 4:(dynamic, faulty)

4.2.2. SLA Violation and Disconnection Rates

The second experiment objective was to compare the SLA violation and disconnections number of the proposed service selection method with the existing ones. The SLA violation happens when the response time of the selected service exceeds the requirement of the client and is measured as the percentage of the service access time during which no SLA violation is observed. For example, in our service selection method, SLA violation may happen when the fast service providers are filtered due to their unreliability in dynamic network conditions.

The simulation parameter values were chosen the same as the previous simulation in section 4.2.1. As explained in section 2, only two service selection algorithms presented in [12] and [13], have particularly focused on the *network failure* issue, which happens due to the dynamic topology of the network. In the first paper, by introducing the SMF concept, the nearest (in terms of hop-counts) service provider to the client node is always selected (*HC* method). In the second paper, a *path expiration time*, based on the node movement patterns, is calculated for each service provider as a measure of its reliability (*PET* method). The comparison results of our proposed method, called *Adaptive*, with *HC* and *PET* methods are shown in Fig.6. The simulation was performed using three network types and two service density levels (with random service distribution) and one requester node in the network. Each experiment was repeated five

times. As shown in Fig.6(a), the disconnections number of the HC method is always lower than the Adaptive method. This is due to the fact that the HC method always selects the nearest service provider (regardless of its response time) while in the *Adaptive* method a distant service provider may also be selected based on the network conditions and its response time. However, as shown in Fig.6 (b), the *Adaptive* method resulted in very lower SLA violations, during the service access, than the HC method.

By comparing to the PET method, it was observed that the Adaptive method particularly in dynamic and changeable networks resulted in a lower disconnections. This is because, the calculation of the path expiration time in PET method is based on the assumptions that all the node clocks are exactly synchronized and the nodes velocities are not variable over the time. However, these assumptions are not true in real world applications. Particularly, in a dynamic or changeable network, any prediction about the path expiration time becomes invalid soon. Moreover, in the *Adaptive* method the unreliability due to the *service failure* is also considered (which is not taken into account in PET method) when ranking the service providers (see formula 2). Both methods resulted in almost equal SLA violations during the service access time. To compare the overall performance of the Adaptive method with PET and HC methods using both metrics (disconnection rate and SLA violation rate), the concept of*Scalarizing* method in multi-objective optimization problems is used [16]. According to this method, when there are more than one metrics to compare solutions, it is possible to consolidate the metric values to a single value using an aggregator function. Hence, we normalized the number of disconnections to obtain the disconnection rates in the range of [0-100] (by dividing each DC number by the maximum DC number observed). The sum of DC rate and SLA violation rate for each method is shown in Fig. 6 (c).

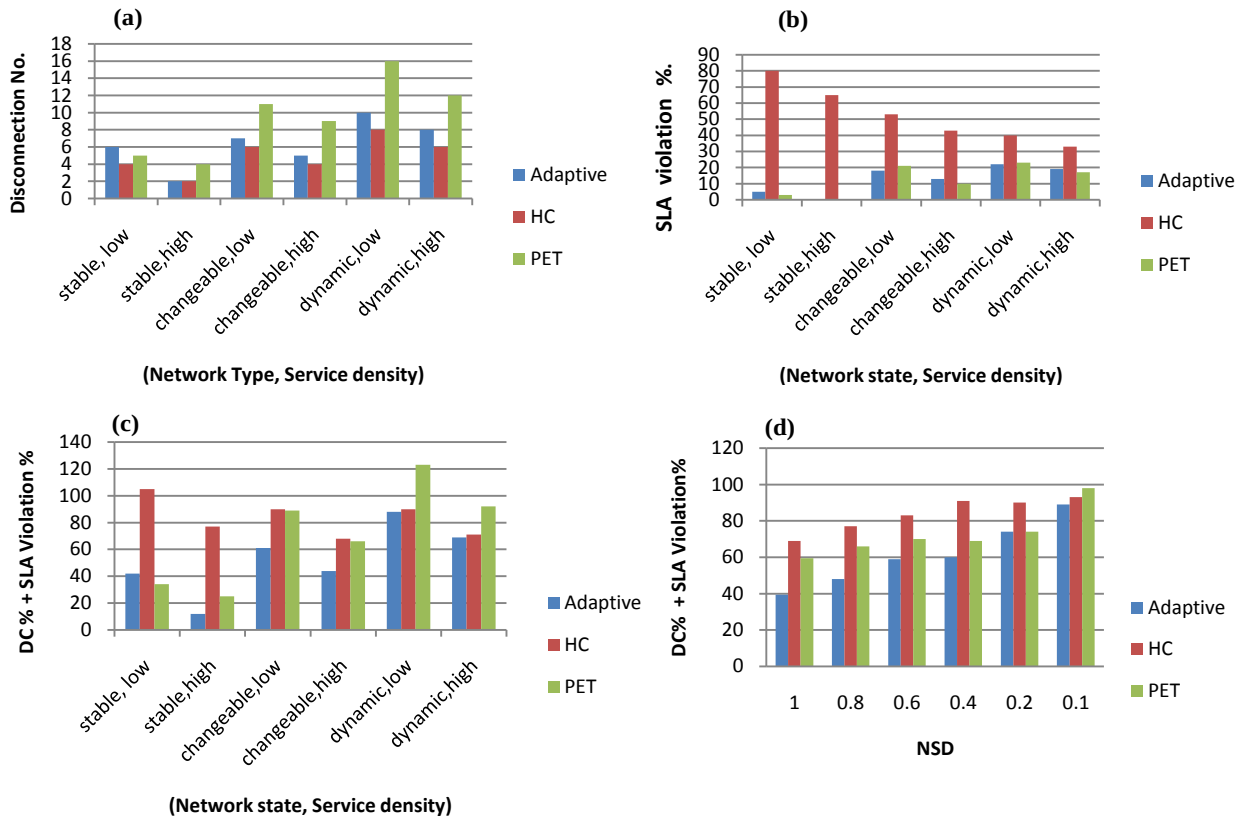

Fig.6.Comparing the disconnection rate (a) and the SLA violation percentage (b) of the proposed method (Adaptive), the Hop-Count method (HC) and the Path Expiration Time (PET) method in different network types and service density values. Chart (c) shows the overall comparison between methods.Chart (d) shows the comparision of methods in different NSD values.

In Fig. 6 (d), the effect of changing NSD values on the quality of service of three methods ispresented. The comparison has been made using the average values observed in different network types: “stable”, “changeable” and “dynamic”. For each network type and NSD value, the experiment was repeated 5 times. Obviously, decreasing the NSD value mayresult in lower quality of service in all methods due to lower availability of services in the network. Moreover, it was observed that the lower the NSD value, the closer the quality of service values in three methods. Consequently, the significance of the proposed method is more obvious in higher service density values.

5. CONCLUSIONS AND FUTURE WORKS

In this paper, we have proposed a novel service discovery and selection algorithm based on both reliability and response time of service providers. Since the reliability of service providers in the network is affected by the general network conditions (dynamic or stable topology, stable or faulty service providing), we proposeda method to estimate the current state of the network using HMM. The experimental results showed that, in comparison to existing methods, our algorithm resulted in fewer disconnections in service access particularly in changeable and dynamic networks. Moreover, lower violations of the client SLA was also observed during the service access. In addition,it was observed that the proposed method is able to estimate the current network state more accurate in networks with lower service densities. The presented method can be applied in multi-agent environments where the mobile nodes (agents) team up to accomplish a rescue or discovery mission.

As the future work, the model-based service selection algorithm should be extended to incorporate intermediate node failure patterns in reliability estimation of a service provider. Another research direction is to address the problem of providing reliable composite services over MANETs.

REFERENCES

[1]. Halonen, T.&Ojala, T., (2006). Cross-layer design for providing service oriented architecture in a mobile Ad Hoc network, Proc. Of MUM '06, pp.193.

[2]. Hoebeke, J., Moerman, I., Dhoedt, B. &Demeester, P.,(2004). An Overview of Mobile Ad Hoc Networks: Applications and Challenges, Journal of The Communications Network, 3(3), 60-66.

[3]. Jayapal, C. &Vembu, S.(2011). Adaptive Service Discovery Protocol for Mobile Ad Hoc Networks, European Journal of Scientific Research, 49(1), 6-17.

[4]. Li, B., (2001).QoS-aware Adaptive Services in Mobile Ad-hoc Networks, Proc. IEEE IWQoS01, p. 251.

[5]. Lund, K., Eggen, A., Hadzic, D., Hafsoe, T. & Johnsen, F. T., (2007). Using Web Services to Realize Service Oriented Architecture in Military Communication Networks, IEEE Communications Magazine, 45(10), 47-53.

[6]. Neema, H., Kashyap, A., Kereskenyi, R., Yuan Xue&Karsai, G.(2010). SOAMANET: A tool for evaluating service-oriented architecture on mobile ad-hoc networks, 14th IEEE/ACM symposium on distributed simulation and real-time applications, p.179.

[7]. Yeow, W. L., Mahmud, R.,& Raj, R. G., “An application of case-based reasoning with machine learning for forensic autopsy”, Expert Systems with Applications, Vol 41, No. 7, 2014, pp. 3497-3505, ISSN 0957- 4174, http://dx.doi.org/10.1016/j.eswa.2013.10.054. (http://www.sciencedirect.com/science/article/pii/S0957 417413008713).

[8]. Raychoudhury, V.,Jiannong Cao, Weigang Wu &Canfeng Chen, (2011). Service Handoff for Reliable and Continuous Service Access in MANET, Proc. PDP’11, p. 172.

[9]. Rekik, J.D., Baccouche L. & Ben Ghezala, H., (2012). An energy efficiency and delay guarantee service selection protocol in MANET, Proc. IEEE MELECON'12 , p.498.

[10]. Riva, O., Nadeem, T., Borcea, C. &Iftode, L., (2007). Context-Aware Migratory Services in Ad Hoc Networks, IEEE TRANSACTIONS ON MOBILE COMPUTING, 6(12), 1313-1328.

[11]. Varshavsky, A., Reid, B., de Lara, E.,(2005). A cross-layer approach to service discovery and selection in MANETs, Proc. IEEE MAHSS'05, p.466.

[12]. Weiyu Chen, Weiwei Sun, Jiaqi Dong, Zhouyao Zhang &Yingxiao Xu, (2008). Discovering and Selecting Reliable Service in Mobile Ad Hoc Networks, Proc. IEEE APSCC08, p.181.

[13]. Xi Zhou, Yifan Ge, Xuxu Chen, Yinan Jing &Weiwei sun, (2011). SMF: A novel light-weight reliable service discovery approach in MANET, Proc. IEEE WiCOM'11, p.1.

[14]. Nickelsen, A., Olsen, R.L., Schwefel, H., (2011). Model-based decision framework for autonomous application migration , Proc. ASMTA11, LNCS 6751,55-69

[15]. Choudhary, A., Roy, O.P., Tuithung T.,(2014). Node failure effect on reliability of mobile ad-hoc networks, Proc. IEEE CSNT2014, p.207.

[16]. Hwang, C.L., Masud, A.S., (1979). Multiple objective decision making, methods and applications: a state-of-the-art survey. Springer-Verlag.

[17]. Bushehrian, O., Shahkolahi, M., (2012). An extensible emulator for investigating configuration algorithms in Mobile Ad-hoc Networks, Int. Journal of Computer Applications, 59(16), 19-23.