A Self-Adaptive Placement Protocol for Mobile Directories in MANETs

Sergio Gonzalez-Valenzuela[1], Son T. Vuong[2], Victor C. M. Leung[1]
[1]Department of Electrical and Computer Engineering
[2]Department of Computer Science
The University of British Columbia
Vancouver B.C., Canada V6T1Z7
1+604+822-6932[1], 1+604+822-6366[2]
{sergiog,vleung}@ece.ubc.ca and vuong@cs.ubc.ca

ABSTRACT
We present our Service Directory Placement Protocol (SDPP), a multi-directory scheme for service discovery in Mobile Ad-hoc Networks (MANETs). SDPP promotes the use of both fixed and mobile directories co-existing in a hybrid setting comprised of devices with different memory availability. Our proposed system is based on a heuristic approach, whose performance is optimized by formulating the directory-placement problem as a Semi-Markov Decision Process solved by Q-Learning. Performance evaluations reveal typical performance gains ranging between 15% and 75% of SDPP compared with a default broadcast approach for MANETs comprised of hosts moving at pedestrian speeds. A two-step process for practical implementation based on "off-line" computer simulations is also described.

Categories and Subject Descriptors
C.2. [**Network Architecture and Design**]: Wireless communication, C.2.2 [**Network Protocols**]: Protocol architecture, I.2.6 [**Learning**]: Parameter Learning.

General Terms
Algorithms, Performance, Design, Experimentation.

Keywords
Service discovery, Mobile Computing, Machine Learning.

1. INTRODUCTION
The increasing popularity of low-cost radio interfaces continues to drive research geared towards ad-hoc communications of miniaturized electronic devices. However, significant design issues arise due to the existing plethora of device types and user mobility, as the traditional data-forwarding role of computer networks shifts into that of ubiquitous service-providers [1]. Unlike their fixed-topology counterparts, Mobile Ad-hoc Networks (MANETs) are comprised of devices whose place in the network varies in time, making it harder for peer hosts to locate each other. In addition, open-access MANETs need to possess self-configurable features, including those that pertain to service advertisement, discovery and provision. To this effect, open-access MANETs should be able to efficiently self-manage in a seamless fashion, so that certain hosts can discover and access the services or digital assets currently being shared by other hosts.

In this paper, we are interested in service discovery issues in MANETs, and in particular, finding an efficient technique that reduces consumed bandwidth and improves the service discovery success rate. An important motivation for this is that the possible applicability of existing protocols for service discovery to MANETs is uncertain. For instance, protocols such as Universal Plug and Play (UPnP) [2] and Jini [3] were designed for static networking environments. Consequently, alternative solutions have been recently proposed by academia. Among the most significant contributions is GSD, a high-level ontology-based approach that enables selective forwarding of service discovery queries in MANETs and reduces packet overhead [4]. GSD's effectiveness depends heavily on whether all MANET hosts support this higher level of information abstraction. Though conceptually similar, a field-theoretic approach was presented in [5], where service discovery queries are modelled as electrostatic particles that are routed to service providers. The efficiency of this system depends on the accuracy with which the Capacity of Service parameter of service providers is assessed among MANET hosts. Other approaches exist too, which rely on surrogate or volunteer hosts that process service discovery queries in a proxy-like fashion [6], [7].

One significant shortcoming seen in these and other approaches that promote the use of directories for service discovery is that they are unable to adapt to topology changes in MANETs. This has a clear impact on the system's effectiveness as the MANET's topology changes from one configuration into another (e.g., Fig. 1.1(a) into Fig. 1.1(b)), causing some hosts to become physically more distant from their service providers. In earlier work, we advanced the Service Directory Placement Algorithm (SDPA) to tackle this issue. SDPA promotes the directory's roaming through the network to get service information closer to the MANET hosts that might need it [8]. In this paper, we enhance SDPA and propose the Service Directory Placement Protocol (SDPP), a methodology whereby fixed service directories coexist with a variable number of mobile directories that roam the MANET in an intelligent manner. As exemplified in

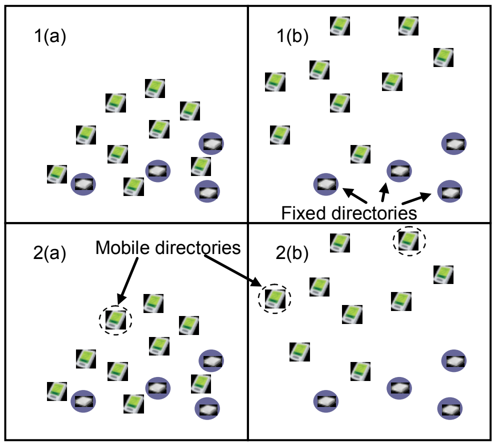

Figure 1. Static vs. mobile directories in a MANET setting

Fig. 1.2, a change in the MANET's topology from (a) to (b) triggers the replication of fixed directories to mobile devices, bringing service information closer to hosts in remote network regions. However, this approach gives rise to the issue of when and where to clone directories. To obtain an efficient decision-making policy, we model this sequential directory placement problem as a semi-Markov Decision Process (SMDP). The policy employed for these decisions is obtained by employing a learning system that is exposed to a variety of situations driven by service querying patterns, host mobility, and physical memory availability. This enables the number of duplicate directories and their placement to effectively adapt to the prevailing network conditions.

The formulation of SDPP as an SMDP is based on the assumption that the rate of MANET topology changes is slow enough to allow the learning algorithm's convergence to an optimal decision policy. We believe that this is a reasonable assumption given that the existing technologies that facilitate MANET deployment (e.g., WiFi or Bluetooth) only support host movement at low speeds given their reduced transmission range. In other words, these technologies do not support rapid topology reconfiguration. To ensure that SDPP provides a practicable solution, we propose a two-step implementation approach. In it, the system is first put to learn "off-line" (i.e., as a computer simulation) to find in a relatively short period of time a suitable decision-making policy that directories can employ when roaming the MANET. In the second step, the numeric values that comprise these decision-making thresholds can be incorporated along with SDPP into mobile hosts (i.e., as an add-on module), for use in a working MANET. The advantages and disadvantages of this approach are discussed later.

We also introduce a Service Entry Ranking System (SERS) to grade the spatiotemporal popularity of all available services to prioritize their advertisement order in the directories. This simple approach leverages the learning system ability to maximize the time-average performance by reducing packet overhead attributed to the directory localization processes. The rest of this paper is organized as follows. Section 2 explains the operation principles of SDPP. The formulation of the learning system is given in Section 3, and Section 4 introduces SERS. The simulations' setup and performance evaluations are presented in Sections 5 and 6, respectively. Implementation aspects are discussed in Section 7, and Section 8 concludes this paper.

2. THE SERVICE DIRECTORY PLACEMENT PROTOCOL

In this section, we introduce the foundations that enable a scalable multi-directory architecture for service discovery in MANETs. SDPP is designed to leverage IETF's Service Location Protocol v2 (SLPv2) [9] for its possible use in a peer-to-peer fashion. The basic architecture of SLPv2 consists of User Agents (UAs) in individual hosts, Service Agents (SAs) associated with service providers, and Directory Agents (DAs) that maintain service directories. SAs register their service information with one of the existing DAs, which in turn periodically broadcasts DA_ADVERT packets to advertise services. Consequently, DAs act as proxies for UAs attempting to discover services. In this regard, UAs individually broadcast service request queries (SRV_RQST) to DAs, which reply with the corresponding service information contained in a SRV_RPLY packet. In addition, attribute reply packets (ATTR_RPLY) follow attribute request queries (ATTR_RQST).

For the incorporation of SDPP into SLPv2 as an add-on module, we further categorize DAs as Fixed Directories (FDs), or Mobile Directories (MDs) that are deployed to mobile hosts as needed. To accomplish this, UAs append to SRV_RQST queries special meta-data of localized network conditions. Upon receiving the corresponding query, a DA passes this meta-data to the local host's SDPP module, whereas the service query is routinely processed by the SLPv2 module. FD duplication follows depending on whether the decision policy determines that this action yields performance improvements in the MANET as a whole. To avoid possible conflicts, FDs wishing to duplicate their contents first send a request to the recipient host candidate, which chooses one of these FDs to initiate the copy. This approach becomes advantageous when a cluster of hosts that are physically distant from a FD enter a pattern in which several service queries are concurrently and repeatedly being issued.

We also contend that SLPv2's operational principles can be enhanced to support peer-to-peer interaction between DAs in a relatively straightforward manner. In this regard, the SDPP module at the local host can coordinate communications with remote DAs by controlling SLPv2 through the use of APIs. In other words, SDPP can instruct SLPv2 to issue the corresponding signals for service (de-)registration (SRV_REG/DEREG) as needed, in addition to those required to discover the presence of other DAs, as explained shortly. Since we propose SAs and DAs co-existing at the local service provider, remote DAs may process regular SLPv2 signals issued by the local SAs to comply with the IETF specification.

We propose a simple procedure for the maintenance of the peer-to-peer FD (DA) network. Here, FDs discover peers to form a "volatile" backbone, whose links may depend on mobile hosts that fracture the current topology when they move. Our objective is to ensure that any given FD remains logically linked to at least one more neighbour FD insofar as the MANET's current topology allows it. To achieve this, we introduce a LOCATE_PEERS packet type. SAs broadcast a LOCATE_PEERS packet in a 2^k incremental-hop fashion in order to control the search depth for FD neighbours at the k-th search attempt. FDs reply with a LOCATE_PEERS_RPLY packet accordingly, causing the FD peer discovery process to stop. FDs then proceed to exchange service-related information. The inter-directory update procedure itself is realized by employing SLPv2's existing service registration signals (SRV_REG) in a prioritized manner as

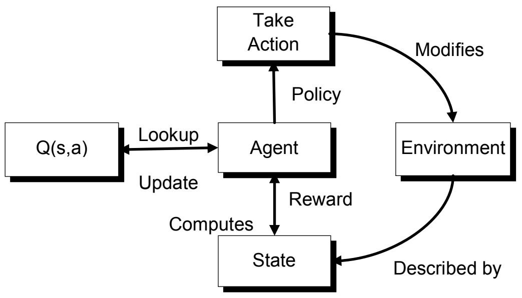

Figure 2. The Q-learning model

recently mentioned. The reason for this is that memory-constrained FDs might be unable to store all of the service descriptions and attributes. Finally, unanswered service queries previously sent onto the backbone network trigger a timeout, causing the local neighbour table's erasure, followed by another peer discovery process as previously described.

3. THE MULTI-DIRECTORY LEARNING SYSTEM

3.1. Foundations and Rationale for the Use of Markov Decision Processes

We assume a high degree of individuality in users' subjective behaviours, their mobility patterns, and their movement history. We also assume that the number of queries received at the current directory's location varies in accordance to users' independent preferences and needs. Therefore, if the long-term spatial distribution of hosts remains statistically homogeneous, we find no constraints for any particular topology or service query condition from recurring sometime into the future [10], [11]. Furthermore, the time until a mobile service directory revisits any given condition (state) will be finite if the circumstances surrounding a MANET environment warrant a limited number of observable conditions. These assumptions are pertinent for hosts moving at low speeds in small to moderately sized networks [12], and are justified in situations where hosts cooperate to facilitate MANET packet forwarding.

Additionally, the lack of memory of previous events indicates that the limiting probabilities of finding the system in any given state will converge to a certain value once it has been operating for a long time. Therefore, our performance improvement objective can be maximized on a time average basis if a suitable policy is employed to dynamically relocate MDs in accordance to the users' combined mobility and service querying patterns. Hence, we can approach this sequential decision-making process by modeling SDPP as a *Markov Decision Process* (MDP).

MDPs consist of several elements: a learning agent (not to be confused with the agents in SLPv2), states, actions, transition probabilities and a reward [13]. In essence, the *agent* is a decision-maker exposed to the system that we wish to control, which can be described as being in one of a potentially large number of *states*. The agent takes *actions* in an attempt to steer the system into a more desirable state, each of which is assigned a subjective value of its worthiness. Because the behaviour of the system is considered stochastic, there is a certain probability of the system's falling into a given state after a certain action is taken. The objective of the agent is thus to maximize the accrued sum of *rewards*, also known as the return R, received over the long term, at which point the steady-state transition probabilities are obtained. The transition time elapsed between the agent's decisions instants (τ_i- τ_{i-1}) is known as the *decision epoch*, which in our case is defined as being real-valued, effectively turning our system into a *semi*-Markov Decision Process (SMDP).

We solve the SMDP by means of a Reinforcement Learning (RL) technique known as Q-learning [14], whose objective is to gradually adjust the initially arbitrary values of a Value Function (VF) to its optimal ones. In our case, this result can be achieved by allowing the agent to learn which directory-copy actions produce the highest reward in the long run. Here, a slow-decreasing discounting factor γ with an initial value in the range (0,1) is used to reduce the weight of future rewards r accrued into R at time t. An optimal policy π for the state-value function $V(s)$ is found when the final value of any state s equals the expected sum of rewards $E\{R\}$ for the process starting at an initial state s, after choosing an action a from the set $A(s)$ that yields the largest reward [13]:

$$V^{\pi}(s) = E\{R_t|s_t = s\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}|s_t = s\right\} \qquad (1)$$

In Q-learning, state-action pairs (s,a) are mapped to their corresponding maximum reward value according to the Bellman Optimality Equations [14]:

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s'} P(s'|s,a)\, max_{a'\in A(s)} Q^*(s',a') \qquad (2)$$

Here, $P(s'|s,a)$ denotes the probability of going from state s to state s' when an action a is taken. For the case of SMDPs solved by means of Q-learning, the discount factor γ must be re-defined, since the states' sojourn time is a continuous random variable. Thus, integrals replace summations to compute the expected return [15]:

$$\begin{aligned} &Q^*(s,a) \\ &= \sum_{s'\in S} p(s'|s,a) \int_0^{\infty}\int_0^{t} e^{-\beta\sigma}\rho(s,a)d\sigma dF_{ss'}(t|a) \\ &+ \sum_{s'\in S} p(s'|s,a) \int_0^{t} e^{-\beta t} max_{a'\in A(s)} Q^*(s',a')\, dF_{ss'}(t|a) \end{aligned} \qquad (3)$$

This expression yields the following result to account for the real-valued sojourn time between state transitions:

$$\begin{aligned} &Q_{t+1}(s,a) \\ &= Q_t(s,a) + \alpha_t \left[\tfrac{1-e^{-\beta t}}{\beta} r(s,s',a) + e^{-\beta t} max_a Q_t(s',a') - Q_t(s,a)\right] \end{aligned} \qquad (4)$$

Here, a learning rate α is introduced to allow the algorithm to converge on one of possibly several existing policies, and β controls the rate of exponential decay. The model of Q-Learning is shown in Fig. 2. An appealing feature of Q-learning is that it does not require formulating a *model* of the underlying system's environment. Therefore, the need to compute the states' transition probabilities is eliminated. This is particularly appealing for the case of SDPP, *since distinct mobility models can be employed*.

3.2 Formulation of SDPP as an SMDP

In SDPP, the delivery of SRV_RQST queries at a FD marks the end of one decision epoch and the beginning of a new one, at which point, the learning agent takes an action. The elements of the RL process for SDPP are described next.

<u>States</u>: The state of the network at time epoch τ_i is an abstract representation that incorporates the following factors as measured for the duration τ_i -τ_{i-1}:

- The difference between the number *n* of neighbouring hosts at both the location of the querying node *U* and in the current location of the directory *V* at time *t*.
- The hop distance between the querying host *U* and the directory *V* at time *t*.
- The average hop distance traversed by queries *q* generated at *U* hosts received by the directory *V* in its current host location at time *t*.
- The difference between the number of queries *q* generated in the neighbourhood of the querying host *U* and those processed by the directory *V* in its current location at time *t*.
- The total directory memory *g* available at the host *U* where duplication is being pondered.

<u>Actions</u>: Only two possible actions are considered for this system, and are described as *A*(*s*)={*copy directory, do not copy directory*}.

<u>Reward</u>: The reward function that we employ incorporates the following elements:

- *J* ATTR_RQST *queries* (*QH*) received from up to *I* mobile hosts.
- *L* ATTR_RQST *queries* (*QD*) received from up to *K* MDs.
- One SRV_RQST query received from any given host that marks the end of the decision epoch.
- *M* ATTR_RQST *packets* (*PH*) received from up to *I* mobile hosts, and their corresponding ATTR_RPLY packets.
- *N* ATTR_RQST *packets* (*PD*) received from up to *K* MDs and their corresponding ATTR_RPLY packets.
- *H packets* transferred when a service directory is copied to a host in the current decision epoch.

$$r(s,s',a)=\frac{\left[\sum_{i=1}^{I}\sum_{j=1}^{J}QH_j^i-\left(\sum_{k=1}^{K}\sum_{l=1}^{L}QD_l^k+1\right)\right]}{\left[2\sum_{i=1}^{I}\sum_{m=1}^{M}PH_m^i(s,s',a)+2\sum_{k=1}^{K}\sum_{n=1}^{N}PD_n^k(s,s',a)+\sum_h^H(s,s',a)\right]} \tag{5}$$

The reward signal *r*(*s*,*s'*,*a*) (5) is thus obtained by dividing the total number of service discovery queries for the current time epoch over the number of packets generated by these same queries resulting for the current epochs, the latter of which ends when the corresponding SRV-RQST query is received at the local FD. This indicates that the LA's reward increases when it sees a larger number of queries with respect to the number of packets, and can be achieved when an FD is copied to a host that ends up servicing more queries from neighbouring hosts. However, if the new MD possesses limited memory to accommodate a sizable number of service descriptions and corresponding attributes, then the FD that duplicated the directory will inevitably observe rely ATTR_RQST queries originating from the new MD that yield unwanted overhead traffic, for which the LA is accordingly penalized. However, the LA will eventually infer the threshold memory value at which the amount of saved bandwidth surpasses the overhead attributed to relied ATTR_RQST onto the backbone.

4. THE SERVICE ENTRY RANKING SYSTEM

SERS is built on the premise that it is counter-intuitive to copy service description/attributes that are seldom being queried in the recent time frame. Doing otherwise ultimately leads the MD's into relaying ATTR_RQST queries onto the backbone network. Therefore, we propose SERS to rank the popularity of individual directory entries based on the frequencies with which they have been recently queried (contrary using to the number of queries that each entry receives). This makes the system fairer to those services that have been available only for a short portion of the time. Otherwise, services that have been available for longer periods would see their ranking unfairly favoured even if their current popularity is low. Individual service entry ranking (SER) values are locally kept and updated. The more often an entry is queried, the higher its value grows, whereas a lack of queries lowers this value according to expression (6):

$$SER_i=\frac{1-e^{-1/\lambda_i}}{\sum_{j=0}^{n}\left(1-e^{-1/\lambda_j}\right)} \tag{6}$$

Here, SER entry *i* will see its value adjusted in proportion to the rate λ with which it is being queried, and it's normalized to the sum of each weighted value of up to n entries at the local directory. Therefore, the query interarrival time for each respective entry is recorded at the local directory (either FD, or MD), since this value is referenced for the recurrent SER calculation. Consequently, a directory entry for a particular service will see its SER value increased or decreased to reflect its current spatiotemporal popularity in the MANET.

As mentioned before, SLPv2's existing service registration (SRV_REG) messages can be readily employed to copy entries from the sender FD to the target device (the new MD). However, memory limitations in hardware-constrained devices might impede performing a full directory copy. Thus, priority is given to entries that possess the highest SER values during the copy process. If the attributes' memory footprint of the current entry is too big to fit into the target memory space, then only the service description is copied, and the process continues to copy the following entries with the highest SER. This process is repeated until either all of the local entries have been copied should the new MD have an equal or greater amount of memory, or if the available memory at the target host is depleted. Both the interarrival query time and the SER value for the directory's entries become part of their attributes. Later, entries with lower SER values are replaced at the local FD/MD by entries with higher SER values fetched from other FDs upon receiving an ATTR_RPLY packet in response to a previously relayed ATTR_RQST query onto the backbone network.

5. IMPLEMENTATION AND SIMULATION SETUP

Our simulations employed the Mobility Framework of the OMNeT++ Network Simulator [16]. We probed the behaviour of the proposed system with networks of 100 and 200 mobile hosts, plus a fixed number of stationary service providers whose physical position in the network is uniformly distributed. Mobile hosts are dispersed in areas of 700 m^2 and 1000 m^2 respectively, and remain within the area's boundaries during the simulation (not a toroid), as we would expect to see real users behave in a real setting. Hosts move according to the well-known random waypoint mobility model, with travel speeds of 1 or 2 m/s, and pause times of 900 seconds. Although this mobility model is known to yield a heavier concentration of elements in the deployment area's center [11], it provides a better approximation to real user mobility than simpler random walk-based models. Hosts' transmission range is limited to 100 m, which yields a

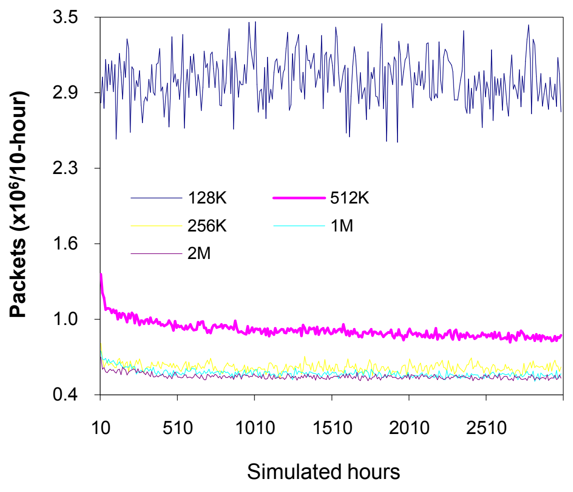

Figure 3. Compound System behaviour with parameters 100H, 64 KB, 10 P, 1 m/s

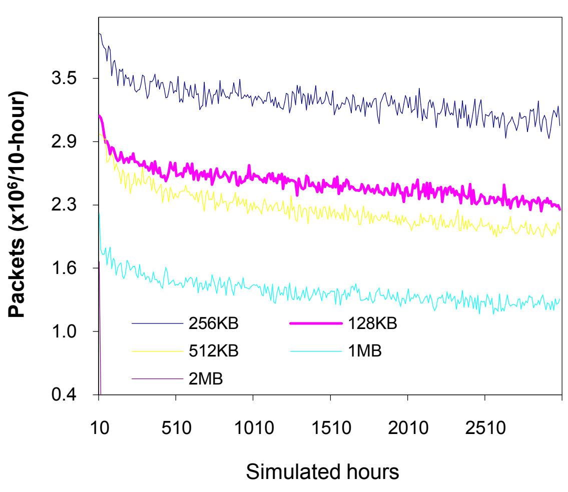

Figure 4. Compound System behaviour with parameters 100H, 64 KB, 10 P, 1 m/s

multi-hop network. The packet length at the network layer is arbitrarily set at 1500 bytes, and the query interarrival time is exponentially distributed with a mean of 2 minutes. We employ a generic MAC layer, and we assume that collisions in the wireless medium are rare. Experiments run for a total of 3000 hours of simulated time, which includes a 600-second warm-up period wherein service providers are sequentially enabled to allow for a gradual discovery of their peers. The learning system remains disabled during this "warm-up" period, after which user hosts begin issuing service discovery queries and both the learning and SER systems at the DAs are enabled.

We also measure the performance penalties paid by the service discovery system when having a limited RAM memory available at the hosts for use by MDs. This provision has been entirely ignored in existing research. We assume that this amount of simulated memory exists in banks of 2^n kilobytes, where, n assumes a value within the range [7…11] for MANETs of 100 hosts, and within the range [9…13] for MANETs of 200 hosts. Each service entry is comprised of a single description and an arbitrary number of attributes that occupies a geometrically-distributed number of bytes. Therefore, the total byte-count per service entry results in an Erlang-distributed value, which is a special case of the Gamma distribution that we use to represent integer-type memory footprints. Given the lack of actual information on service entries' memory footprints, we arbitrarily employ values of 64 and 128 KB. Also, the probability of each service entry being queried is uniformly distributed.

FDs and learning agents coexist as symbiotic pairs in the service providers. This allows reward values to be individually attributed to each of the agent's actions. A small value of 0.1 is initially set for both α and β in equation (4) at each of the FDs' respective learning systems, which in turn perform their work in a non-cooperative fashion (individually from other learning agents). Therefore, the obtained polices are not shared with one another. In addition, each MD computes the number of service queries that it is able to successfully process. If an MD observes more packets relayed onto the backbone network than those that are locally processed (resulting in a negative balance), then the MD surrenders its role. In this case, all directory-related information is flushed, and the host assumes a regular status.

We tested SDPA with a slightly modified version of SLPv2 so that, upon receiving a SRV_RPLY packet, a regular host broadcasts a DA_ADVERT to its 1-hop neighbours to inform them of the DA's location just discovered. Although, this task is originally performed by the DAs, in SDPP neither FDs nor MDs issue any DA_ADVERT signal. Other than that, service/attribute requests/replies are routinely processed according to the corresponding IETF specification. In addition, our simulation implements a nearly-generic loop-free routing scheme wherein paths are discovered on-demand. This allows us to isolate the effect that any particular routing algorithm might have on our system, and therefore measure the effectiveness of our approach in a reasonably unbiased fashion.

6. SIMULATION RESULTS AND DISCUSSIONS

We evaluated the packet overhead incurred by SDPP, as well as the directory location/discovery success rate. Evidently, the fewer packets attributed to service discovery queries while maintaining a high success rate for directory localization, the better. We gauge these performance metrics against a baseline broadcast flooding (where no MDs are employed). We also evaluate SDPP against existing multi-directory approaches. However, accurate comparisons are not straightforward given the variety in methodologies, evaluation platforms, simulation parameters and performance metrics employed. As a result, the existing literature on SDPs for MANETs also reports comparisons against a default broadcast-flooding approach [4]-[7].

Service discovery packets are counted in a moving-window fashion by sampling their number every 10 simulated hours, at which point the count is reset to 0. Figs. 3 and 4 illustrate the typical behaviour of a sample set of measured data in our system once SDPP is put into action in a 100-host MANET under the specified simulation parameters. Our first observation is that the learning system is only incapable of finding an efficient copy decision policy that yields better system performance when the average memory available for use by an MD is relatively small. In this case, the constant swapping in and out of the low-SER entries at the directories causes oscillation in the learning system when only 128KB of RAM are available for MD use at the hosts.

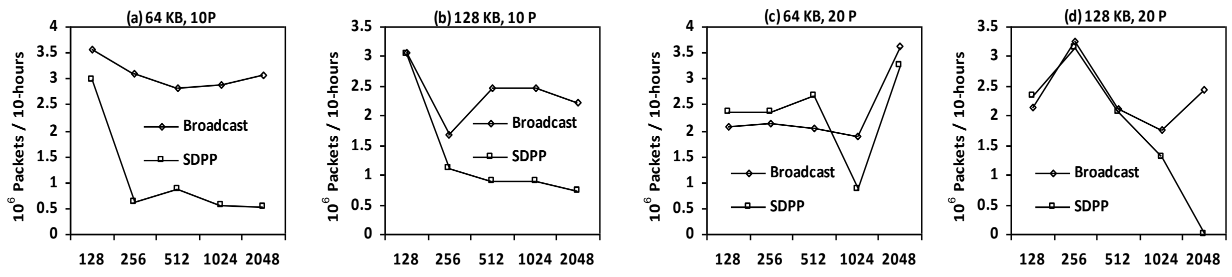

Figure 5. Summarized system behaviour for 100 hosts moving at 1 m/s.

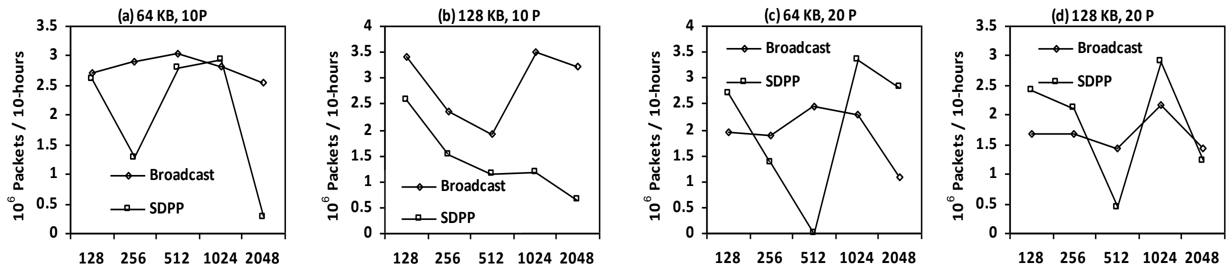

Figure 6. Summarized system behaviour for 100 hosts moving at 2 m/s.

However, for the remaining of test cases, the plots show a gradual packet overhead reduction attributed to the learning system, which takes place approximately one-third into the simulation period.

Figs. 5-8 summarize the performance of SDPP against the broadcast approach by comparing the average number of packets observed in 10-hour periods during the last third of their corresponding simulations (which in the broadcast case remains constant). For instance, the average number of packets observed in the last third of each curve shown in Fig. 3 yields one dot in the plot for SDPP that is shown in Fig. 5(a). These curves depict how the packet overhead behaves as the DA memory window is gradually increased. We can observe a dramatic decrease in packet overhead when the maximum allotted memory space is increased from 128KB to 256 KB (from one simulation outcome to the next), service entries' memory footprint average 64 KB, hosts move at 1 m/s, and there are 10 service providers in the network. For 100-host MANETs, SDPP's performance shows packet reductions between 18% and 75% over the broadcast flooding scheme as shown in Figs. 5 and 6. Overall, SDPP outperformed the baseline approach roughly 75% of the time for this simulation set. Nonetheless, our best results were obtained in the 200-host simulations, with 90% of the results favouring SDPP's use over the baseline approach. As in the 100-host case, performance improvements are highly variable too. Improvements range from being fairly conservative to others spanning nearly a whole order of magnitude, as depicted in the 512 KB case of Fig. 7(a).

We ascribe these variations in performance to two main factors. The first one deals with the distinct parameters that we employed in our simulations. Whereas some of these parameters were arbitrarily chosen (due to the lack of realistic information in the literature), others attempt to mimic the possible conditions observed in a real MANET setting. In particular, the RAM memory window allocated for MD use at the hosts played an important role in the simulations' results, which yielded the best results after being increased in the 200-host cases. The reason for this is that a larger number of MANET hosts translates into service entries being queried more often at the FDs, thus causing more frequent swapping of service entries and increased bandwidth consumption. This results in the individual learning systems abstaining from delegating MD duties to regular hosts. In the extreme case, no MDs are employed, and the pure-FD approach is favoured. Nonetheless, by increasing the DAs' memory window, we allow for more entries that can be kept for longer periods of time, thus reducing traffic. In fact, it can be seen that there are instances in which the system seems to reach a stabilization point when the upper-limit of the memory window allocated to the DA reaches 1 MB. In general, MANETs in our experiments see improved performance as mobile hosts allocate memory on the higher end of the memory window, which can be confirmed in Figs. 5(d), 6(c) and 7(a), wherein the limited number of service entries can be effectively handled by a surplus of simulated RAM at the hosts. We observe that, in certain cases, hosts have enough memory to store all of the available services' information in the MANET. This leads to the unrestricted replication of the directories to all mobile hosts until they all become MDs, and SRV_RQST queries are no longer issued. Therefore, each mobile host has a local directory copy that it can reference to enquire about any of the available MANET services. This suggests that performance variations are not necessarily attributable to SDPP, but to the operating parameters of the MANETs hosts. Hence, SDPP system learns to work in a best-effort fashion, instantiating as many MDs with larger directory memory allocations as there are available improve performance.

In addition to the above, our results suggest that performance variations between the 100- and 200-host MANETs arise as a result of the actual physical distribution of FDs in the MANET. Given that FDs are randomly distributed, a portion of them may sometimes suffice to cover the MANET deployment area, rendering the use of MDs as unnecessary. Conversely, FDs in a different simulation might end up more or less clustered within a particular deployment area. In such cases, the MANET benefits from the use of MDs servicing the voids left by the irregular (random) distribution of FDs in the deployment area. At present, the learning system is unable to deal with this issue because the

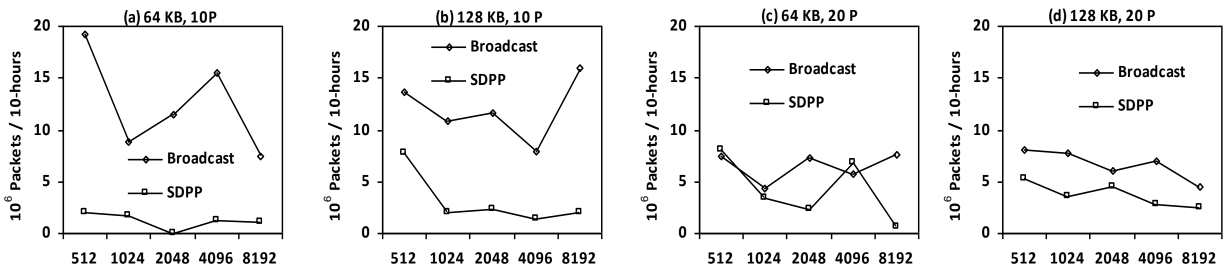

Figure 7. Summarized system behaviour for 200 hosts moving at 1 m/s.

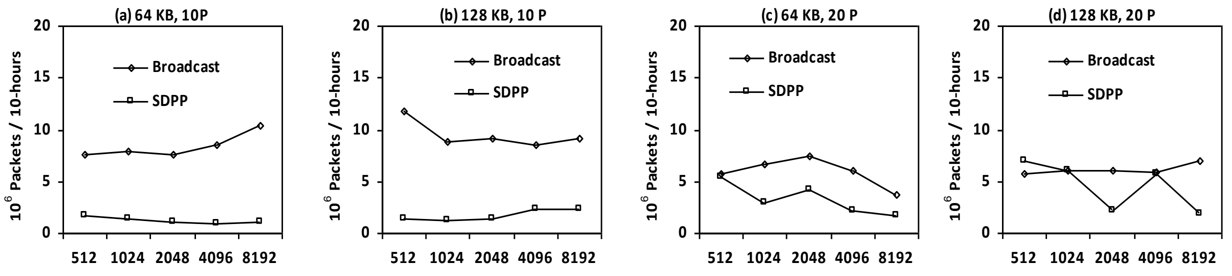

Figure 8. Summarized system behaviour for 200 hosts moving at 2 m/s.

current system state representation conveys no information about the FDs' actual position that can employ in making MD-cloning decisions. Although this issue is straightforward to solve in a simulation setting, it is not the case in a in a real MANET setting. Therefore, we decided to simulate the real MANET setting to examine the behaviour of SDPP in such circumstances.

Our results also reveal that SDPP is more scalable than its SDPA predecessor. SDPP also compares well with existing approaches under somewhat similar simulation parameters. For instance, Fig. 13(c)/(d) of the GSD approach yields a network load that amounts to roughly 100,000 messages in a 100-host MANET after 75 simulated minutes [4]. SDPP's performance parallels this result for the same simulated time when the hosts allocate 512KB on average for directory copies. However, GSD's messages are not network layer packets, but application layer messages. It is unclear from the available information how many TCP packets their messages would yield. Similarly, SDPP produces half as many packets than those reported in [6], even when the number of MANET hosts considered in our SDPP evaluation is twice as large. Table 2 shows the directory localization success rates often approaching the 100% efficiency mark in a 200-host MANET, meeting or exceeding the results in Table 1, and those reported in the referenced approaches. For instance, a success rate of 96% is attained when hosts that move at 1 m/s allocate an average of 128 KB of directory memory space, there are 10 services are available in the MANET, and the average memory footprint of a service description plus its attributes averages 64 KB. Results for the 100-host MANET experiments (not shown) are similar.

7. PRACTICAL CONSIDERATIONS

As mentioned in Section 1, it is evident that the learning system cannot be implemented "as-is" in real MANETs, given the long time it would take for the Q-function to approach optimality. Therefore, we propose a simple two-step process for the implementation of SDPP in a real MANET scenario. In the first step, the system is put to learn "off-line" (i.e., as a simulation) in order to populate the Q-function with the values that yield average maximized performance for a predefined host mobility model as deemed suitable. In this regard, the host mobility model should be carefully designed in order to reflect the expected conditions as realistically as possible. In the second step, the Q-function obtained from these simulations can be loaded into real service providers for use by SDPP. In this case, not all of the state-action pairs need to be stored. Instead, this reference table would be comprised only by the subset of states with the smallest cardinality (e.g., either the relocate, or do not relocate subset). We believe this to be a practicable approach, since our Q-function tables yield less than 20,000 states, which would require only a few tens of kilobytes of ROM space. On the other hand, DA information can be stored in RAM space as available in the mobile devices' corresponding hardware. The information needed to assemble the system's state abstraction can be readily obtained through cross-layer communications with the Medium Access Control (MAC) layer, and with the routing layer.

We also observe that a number of service discovery schemes based on the so called *web-services* framework have been recently proposed. Network services are accordingly described and categorized in a highly-structured and information-rich fashion in an attempt to facilitate service matching. However, this leads to increased memory footprints for service descriptions, becoming a problem in MANETs comprised by thin devices with memory limitations. This is a consideration of particular importance for media-rich service information that may include audio, images or even video. In this regard, the contribution introduced by SDPP becomes apparent.

8. SUMMARY AND CONCLUSIONS

We have shown the effectiveness and limitations of employing both fixed and mobile directories for service discovery in mobile computing environments. The formulation of SDPP's as an SMDP and its solution through the Q-learning technique proved helpful in defining a directory-copy policy that can save bandwidth. We also showed that allowing directories to become mobile leverages the performance of the service discovery system,

Table 1. Broadcast directory localization rate

Success rate for 200 hosts moving @ 1 m/s			
Avg. entry footprint 64 KB		*Avg. entry footprint 128 KB*	
Directory size	Providers (10/ 20)	Directory size	Providers (10/20)
512KB	0.88 / 0.95	512KB	0.92 / 0.93
1024KB	0.90 / 0.95	1024KB	0.85 / 0.95
2048KB	0.96 / 0.96	2048KB	0.92 / 0.96
4096KB	0.88 / 0.96	4096KB	0.90 / 0.95
8192KB	0.94 / 0.96	8192KB	0.84 / 0.96
Success rate for 200 hosts moving @ 2 m/s			
Avg. entry footprint 64 KB		*Avg. entry footprint 128 KB*	
Directory size	Providers (10/ 20)	Directory size	Providers (10/20)
512KB	0.90 / 0.91	512KB	0.84 / 0.95
1024KB	0.92 / 0.92	1024KB	0.89 / 0.95
2048KB	0.92 / 0.94	2048KB	0.88 / 0.96
4096KB	0.94 / 0.96	4096KB	0.89 / 0.94
8192KB	0.79 / 0.93	8192KB	0.91 / 0.93

Table 2. SDPP directory localization rate

Success rate for 200 hosts moving @ 1 m/s			
Avg. entry footprint 64 KB		*Avg. entry footprint 128 KB*	
Directory size	Providers (10/ 20)	Directory size	Providers (10/20)
512KB	0.96 / 0.96	512KB	0.96 / 0.95
1024KB	0.96 / 0.96	1024KB	0.95 / 0.97
2048KB	0.99 / 0.98	2048KB	0.98 / 0.99
4096KB	0.97 / 0.97	4096KB	0.96 / 0.98
8192KB	0.98 / 0.99	8192KB	0.96 / 0.96
Success rate for 200 hosts moving @ 2 m/s			
Avg. entry footprint 64 KB		*Avg. entry footprint 128 KB*	
Directory size	Providers (10/ 20)	Directory size	Providers (10/20)
512KB	0.95 / 0.94	512KB	0.95 / 0.97
1024KB	0.96 / 0.97	1024KB	0.96 / 0.97
2048KB	0.97 / 0.98	2048KB	0.96 / 0.98
4096KB	0.97 / 0.99	4096KB	0.96 / 0.97
8192KB	0.96 / 0.97	8192KB	0.96 / 0.98

which depends on the hosts' speed and the memory size of the service directory. Still, SDPP effectively minimizes performance degradation in a best-effort manner by enabling the service discovery system to dynamically adjust to the current MANET's circumstances. We also note that the effectiveness of the Q-learning approach depends on whether the Markovian property of the underlying environment holds. Therefore, SDPP's degree of efficiency depends in part on the accuracy with which the host mobility model describes the movements of real people. However, we note that mobility models that do not adhere to the Markovian property may still yield sub-optimal results when solving the SMDP [13]. We believe that our proposed approach provides the necessary elements that motivate further explorations into the applicability of machine learning for service discovery in MANETs. SDPP also proved to be much more scalable than its predecessor, SDPA. We outlined the groundwork for SDPP to be employed in MANETs of slow-changing topology. However, SDPP can be foreseeable employed in any MANET in which the topology change rate is low enough to allow the learning system to respond accordingly before going into oscillation. Ultimately, the applicability of SDPP depends on the relative mobility between hosts in any MANET, provided that the hosts' speeds and transmission ranges are proportionately increased.

9. REFERENCES

[1] F. Zhu, M. W. Mutka, L. M. Ni, "Service Discovery in Pervasive Computing Environments," *IEEE Pervasive Computing*, Vol. 04, No. 4, pp. 81-90, Oct-Dec, 2005.

[2] The UPnP Forum. http://www.upnp.org

[3] Jini Network Technology. http://sun.com/jini

[4] D. Chakraborty, A. Joshi, Y. Yesha, T. Finin, "Toward Distributed Service Discovery in Pervasive Computing Environments," *IEEE Transactions on Mobile Computing*, Vol. 5, No. 2, pp. 97-112, February 2006.

[5] Lenders, V., May, M. and Plattner, B. "Service Discovery in Mobile Ad Hoc Networks: A Field Theoretic Approach," in *Journal on Pervasive and Mobile Computing*, Vol. 1, Issue 3, p. 343-370, Elsevier, September 2005.

[6] U. C. Kozat and L.Tassiulas, "Service Discovery in Mobile Ad-hoc Networks: An Overall Perspective on Architectural Choices and Network Later Support Issues," *Ad-Hoc Networks*, Elsevier, 2004.

[7] M.J. Kima, M. Kumara, B.A. Shirazib, "Service discovery using volunteer nodes in heterogeneous pervasive computing environments," *Pervasive and Mobile Computing*, Vol. 2, pp. 313–343, Elsevier, 2006.

[8] S. Gonzalez-Valenzuela, S.T. Vuong, and V.C.M. Leung, "A Mobile-Directory Approach to Service Discovery in Wireless Ad-hoc Networks," in press, *IEEE Transactions on Mobile Computing*.

[9] Service Location Protocol v2, IETF RFC2608, http://www.ietf.org/rfc/rfc2608.txt.

[10] G. Lin, G. Noubir and R. Rajaraman, "Mobility Models for Ad hoc Network Simulation," in *Proceedings of IEEE INFOCOM*, Hong Kong, China, March 2004.

[11] C. Bettstetter, G. Resta and P. Santi, "The Node Distribution of the Random Waypoint Mobility Model for Wireless Ad Hoc Networks," *IEEE Transactions On Mobile Computing*, Vol. 2, No. 3, pp. 257-269, July-September 2003.

[12] F. Bai, N. Sadagopan and A. Helmy, "IMPORTANT: A Framework to Systematically Analyze the Impact of Mobility on Performance of RouTing protocols for Adhoc NeTworks," in *IEEE INFOCOM*, San Francisco, USA, April 2003.

[13] R. S. Sutton and A. G.Barto, "Reinforcement Learning – An Introduction." *MIT Press*, Cambridge, USA.

[14] C. J. C. H. Watkins, "Learning from Delayed Rewards," *PhD thesis*, Cambridge University, England, 1989.

[15] S. J. Bradtke and M. O. Duff, "Reinforcement Learning Methods for Continuous-Time Markov Decision Problems," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. S. Touretzky and T. K. Leen, Editors, MIT Press, Cambridge, USA 1995.

[16] The OMNeT++ Discrete Event Simulator. http://www.omnetpp.org.