DARPA Contract # W15P7T-05-C-P413
Deliverable #3(c)

Mobility Management in EDIFY

Evaluation via Analysis and Simulation

Lehigh University

Oct 31, 2005

Abstract—**Standard ad hoc routing protocols do not work in intermittently connected networks since end-to-end paths may not exist in such networks. A store-and-forward approach [7] has been proposed for such networks. The nodes in such networks move around. Thus, the architecture proposed in [7] needs to be enhanced with a mobility management scheme to ensure that nodes that wish to correspond with mobile hosts have a way of determining their whereabouts. The mobile hosts may move a short distance and hence remain within the vicinity of a DTN name registrar (one communication link away) or they may move far away (multiple communication links away). In this report, we present the mobility management scheme and partition detection mechanisms we propose for DTN environments. In addition, we provide simple analytical formula to evaluate the latency required for performing location updates, and the useful utilization that each node can use for data transfer assuming that the communication links between nodes are periodically available for a short period of time. We also present some simulation results on the location update latency and useful utilization in multihop wireless networks to which the analytical model caters. Our simple analytical model and simulations allow us to draw insights into the impact of near/far movements on the useful utilization. Furthermore, we also study scenarios where a DTN mobile node visits various DTN regions supported by wired DTN nodes. More detailed simulations permit comparisons of different registration schemes for location updates in such scenarios. Our preliminary evaluations indicate that the home DNR update scheme achieves the highest query success rate with the lowest location update overhead but it has the largest average query response time.**

Keywords-disruption tolerant networks; mobility management; partition detection; ad hoc networks

1. INTRODUCTION

There are many real mobile networks in which wireless devices are intermittently connected. Most of the time, a complete path from source to destination does not exist or such a path is highly unstable and may change after it has been discovered. Examples of such networks are wildlife tracking sensor networks [1], military networks [2], and inter-planetary networks [3].

Since in such intermittently connected networks an end-to-end path between a source and a destination may not exist, conventional mobile ad hoc network routing protocols such as DSR [4], AODV [5], etc., will not work. Reactive routing schemes will fail to discover a complete path and proactive routing schemes will fail to converge resulting in spurious topology update messages. However, this does not mean that messages cannot be delivered from the source to the destination. It just means that a message needs to be sent over an existing link, get buffered at the next hop until the next link is up, etc., until it reaches its destination.

Researchers working on Delay Tolerant Networking [7] proposed a bundle delivery protocol to allow nodes in intermittently connected networks to communicate via the store-and-forward approach. We have proposed enhancements to the Delay Tolerant Network architecture work which we refer to as EDIFY [6] to deal with mobility. We refer to intermittently connected networks that have been enhanced with the bundle delivery protocol and other EDIFY features as disruption tolerant networks (DTNs).

In this report, we first discuss why the existing mobility management schemes do not work in a DTN environment. Then, we provide more details on how the mobility management scheme works in EDIFY. We also discuss how to recognize movement (as changes to network topology) within the problem of partition detection. The direct connectivity between two nodes in an intermittently connected network may toggle between on/off states all the time. Thus, it is important to understand what fraction of the on-times a mobile host can use for data transfer after it spends some time performing location updates when it roams around. We refer to this metric as the useful utilization. In this report, we provide a simple analytical model that allows us to draw insights about the impact of near/far movement on the useful utilization a mobile node can have for data transfer after performing location updates. Then, we also evaluate a scenario where mobile nodes move according to a Zebranet trace [1] and see how frequently they perform registrations with a visiting DNR and how such registration events trigger location update messages using different location update schemes. We also evaluate the average query success rate and query response time at different querying rates when different location update schemes are used.

The rest of the report is organized as follows: In Section 2, we discuss related work in the area of mobility management and explain why they do not apply to the DTN environment. Then, in Section 3, we briefly describe the EDIFY architecture. In Section 4, we discuss how mobility management is done in EDIFY. In Section 5, we discuss the importance of partition detection and approaches to address it. In Section 6, we describe an analytical model that allows us to compute the location update latency and useful utilization assuming the links between nodes are available periodically for a certain percentage of the time. Then, via simulation, we show that the results obtained via analysis match closely with the simulation results. In Section 7, we describe in some detail the design of our more detailed simulator. This simulator allows us to evaluate a scenario where mobile nodes move according to the Zebranet trace and evaluate the transmission overhead caused by the registrations of these mobile nodes with visiting DNRs. We also use this simulator to evaluate the average query success rate and query response time with different query rates and transmission ranges. We present and discuss the simulation results we obtained using this simulator. We conclude with some remarks in section 8.

2. RELATED WORK

Several network layer mobility solutions, e.g., Mobile IP [11] and SIP mobility [12], have been proposed to provide mobility transparency. However, such protocols cannot work in partitioned networks since the protocols assume that an end-to-end connection exists.

Transport layer mobility solutions also exist. In TCP Migrate [9], the mobile host periodically updates a unique fully-qualified domain name for itself in the Domain Name System (DNS). The corresponding host uses this information to find the current IP address of the mobile. The protocol stacks of both the mobile host and the corresponding host are also modified to migrate the TCP sessions across prolonged disconnections and IP address changes. However, end-to-end connections are still required with this approach. In addition, frequent DNS updates may result in high control overhead.

Some researchers working on Delay Tolerant Networks [7] proposed a notion of bundle transfer, where a message is wrapped into bundles and these bundles are delivered from one hop to another, then stored at that next hop before another opportunistic link appears to further the bundles towards their final destinations. Custody transfer occurs as the bundles are transmitted from one hop to another and the responsibility of reliable delivery of each bundle lies in each DTN router that is involved in the delivery path. Note that there is a delivery path but no simultaneous end-to-end path. However, their existing solution does not support mobility. We have proposed extensions [6] to their approach to deal with mobility. In [10], the authors also propose an architecture to deal with disconnected networks. They propose a cellular-like solution that consists of Home Location Register and Visiting Location Register. The VLR contains information on the custodian DTN router for a visiting mobile.

3. EDIFY ARCHITECTURE

In the EDIFY architecture [6], there are several types of nodes, namely, regular DTN nodes, DTN Name Registrars (DNRs), and DTN gateways. We briefly describe the functionalities provided by each type of node below:

- Regular DTN nodes

In EDIFY, all nodes participating in the DTN have the ability to send and receive bundles to other nearby nodes using the underlying networking infrastructure. When a node joins a group, the node is informed of a default node (a gateway) to which bundles may be sent. The node also implicitly knows of the group registrar, from which each node acquires its name within the group. Every node in the DTN can have one or more names but only one name will be selected to be their "home name", e.g., bob@cse.lehigh.edu

- DTN gateways

A DTN gateway is a DTN node that offers forwarding services to one or more destination groups. Nodes that perform forwarding services form the backbone of the DTN. DTN gateways may also advertise the availability of routing services to non-local groups to other gateways inside or outside of the group to which a gateway belongs.

- DTN Name Registrar (DNR)

Every group in the DTN world has a registrar (DNR) associated with it. We assume that the DNRs are given standardized DTN names e.g. dnr@lehigh.edu where lehigh.edu is the group's name. The DNRs form an overlay network above the DTN. The function of a registrar is provided by one or more (for robustness) DTN nodes. The registrar is responsible for communication with the parent group(s). It offers the mandatory service of registering the members and visitors of its group. It is responsible for ensuring the authenticity and eligibility of the nodes requesting to be registered, either as a visiting node, or as a full member node. The registrar ensures that the node identifier assigned to a requesting node is locally unique. In addition, the registrar receives registration updates from group members that are visiting elsewhere. Such updates provide information on how to reach these nodes so a DNR may update appropriate gateways with the latest node reachability information. An example of an EDIFY DTN is shown in Figure 1.

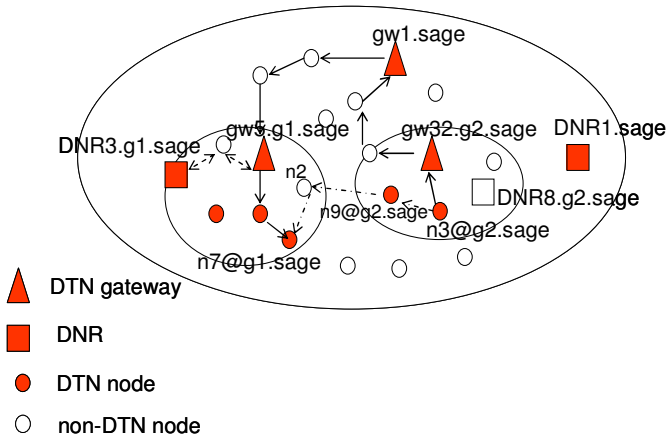


Figure 1: EDIFY DTN

In Figure 1, we show one administrative group of DTN nodes. This group is further subdivided into two subgroups. There is a DNR at the highest level of the hierarchy with a DTN name, DNR1.sage. There is also one DNR each in each subgroup with DTN names DNR8.g2.sage and DNR3.g1.sage. When node n3@g2.sage wants to send a bundle to n7@g1.sage, it checks its default forwarding policy which indicates that it should send any bundles destined to other subgroups of sage to gw32.g2.sage. When the bundles arrive at gw32.g2.sage which participates in an intragroup forwarding protocol, gw32.g2.sage has a forwarding entry that indicates that any bundles destined to g1.sage should be sent to gw1.sage. At gw1.sage, there is a forwarding entry that says that bundles destined to g1.sage should be forwarded to gw5.g1.sage. Once the bundles reach gw5.g1.sage, gw5.g1.sage can seek the help of DNR3.g1.sage to resolve the DTN name n7@g1.sage to a routable address and forward the bundles to the destination node.

The above description discusses how intragroup bundle forwarding takes place. The intergroup bundle forwarding protocol is out of the scope of this report and will be covered elsewhere.

4. MOBILITY MANAGEMENT

Since nodes move around in DTNs, we need to design a mobility management scheme to ensure that either the gateway or the DNRs know how to forward the bundles to any DTN node. There are three major scenarios that we need to consider:

- Individual Visiting Node Scenario

When a DTN node visits a new area, it broadcasts a DNR discovery message. Any DTN node that hears this message should respond with a unicast DNR announcement message to inform the node of the nearest DNR about which it knows. The visiting node can then register with that local DNR and possibly obtain a visiting identifier. The visiting node can also activate a forwarding feature to request that the local DNR forwards a "location update" message to its home DNR. Since we are dealing with intermittently connected networks, this location update message may not reliably arrive at the home DNR. Thus, intermediate DNRs that receive location update information will cache such information in the case that the policy of caching location update information is enabled.

- Group Visiting Scenario

When a group of nodes visits another place, instead of doing individual registrations with the local DNR, they can elect a representative, and have that representative node register with the local DNR on behalf of all of them to reduce the amount of control overhead required for location updates. This representative node will act as an impromptu gateway/DNR for the visiting group. Any messages it receives from the home group via the local gateway will be delivered to all the nodes in the visiting group.

In the two mobility scenarios described above, after receiving the registration message, the local DNR should include the local default gateway information in its registration response message to the visiting node or the representative of the visiting group. In addition, it is assumed that the local DNR will periodically send a list of visiting nodes to the gateways within its own group and to the top-level DNR to which it belongs. This is done to ensure that if the home DNR of a visiting node issues a query about the node's latest location, the visiting top-level DNR will be able to answer that query. In addition, the visiting top-level DNR will be the proxy that sends the location update information to nearby DNRs with the intention of passing this information eventually to the home DNR if there is such an opportunity. The local DNR can set its own policy on the maximum number of DNR hops that such information will be propagated to minimize the control overhead for such location update messages.

- Mobile Network Scenario

In some scenarios, a whole network, e.g., the network hosted inside an airplane [8], may move around at a fast speed. A mobility management scheme needs to be designed to handle such mobile network scenarios as well as scenarios where a network can be partitioned into multiple groups due to geographical obstructions or enemy attacks. One approach is to have the mobile network register itself with a visiting DNR and individual nodes within this mobile network will perform registration with the DNR of the mobile network. For example, let say bob@cse.lehigh.edu is on Plane101.SIA. Currently, Plane101.SIA is at San Francisco Airport. So, Plane101.SIA registers with the DNR of San Francisco Airport network. In addition, Plane101.SIA has pre-registered at the DNR.SIA with its flying schedule so that DNR.SIA knows which airport network to probe for the presence of Plane101.SIA at any particular time. Bob registers with Plane101.SIA and asks the DNR of Plane101.SIA to inform its home DNR (dnr@lehigh.edu) if possible of his whereabouts. Plane101.SIA can send this location update information to the DNR at San Francisco Airport network. If there is network connectivity between San Francisco Airport network and Lehigh University, then Bob's home DNR can certainly get this information. Otherwise, some intermediate DNRs like the DNR at San Francisco Airport Network will have cached information of this location update, and will be able to answer future queries about the whereabouts of Bob.

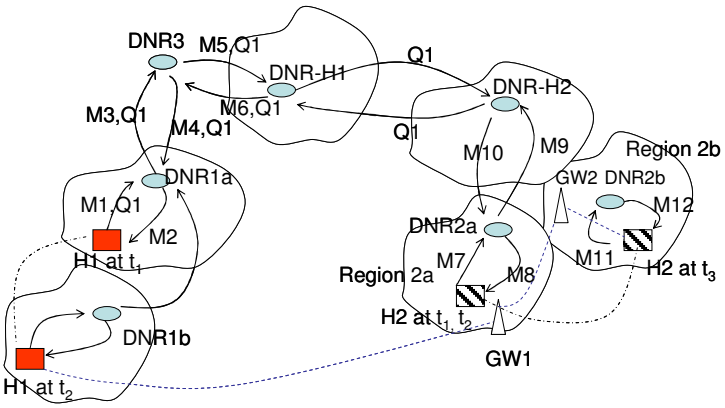


Figure 2: A Node Mobility Scenario

In Figure 2, we illustrate an individual node mobility scenario. Assume that a mobile host H1 is visiting an area served by DNR1a at time t_1. H1 first broadcasts a DNR discovery message. After hearing a response from either a local node or DNR1a itself, H1 is ready to perform a registration with DNR1a. H1 turns on its home registration feature so DNR1a will be asked to act as a proxy to forward the location update information to DNR-H1, the home DNR of H1. This location update information, M3, will be referred to as home location update (HLU) message. The HLU message, M3, is delivered to DNR-H1 via DNR3. Our protocol requires any intermediate DNR to acknowledge the HUL message it receives and cache the information before attempting to forward it to the final destination. So, DNR3 will respond with a HLU acknowledge message (denoted as M4) before it tries to deliver the HLU message further to DNR-H1 (denoted by message M5). DNR-H1 will send an acknowledgment for the HLU message M5 if it receives it. If the communication link between DNR3 and DNR-H1 is not available and hence M5 cannot be delivered, then H1's location information will still be cached in DNR3 and DNR1a.

Similarly, another mobile host H2 which is visiting an area served by DNR2a will perform a registration after discovering DNR2a. Assume that at time t_2, H1 moves to an area served by DNR1b. H1 will perform similar registration with DNR1b after discovering DNR1b. Now, H1 may not want DNR1b to send home location update message to DNR-H1 but request that DNR1b sends a location update message to its previously registered visiting DNR, DNR1a.

Now, assume that H1 wants to send a bundle to H2. H1 sends its bundle to the default gateway that DNR1b assigns when H1 performs registration. That default gateway will then query DNR1b if it does not know how to forward the bundle to H2. Alternatively, that default gateway may have information on how to send the bundle to a home gateway within the area covered by DNR-H2. When the bundle arrives at that home gateway, the home gateway queries DNR-H2 and finds out that H2 is now visiting at an area served by DNR2a and hence delivers the bundle to a local gateway in that area which eventually delivers the bundle to H2. H2 may add additional header in the bundle acknowledgement (if bundle acknowledgement feature is turned on) or a special "location update" message to H1 so that H1 knows its visiting identifier, and hence H1 can send bundles directly to H2 (using H2's visiting identifier) rather than via its home network.

Let us assume that H2 moves to an area (region) served by DNR2b while the bundles from H1 are being sent from its home gateway towards a local gateway in Region 2a served by DNR2a. Until H2 performs location update with DNR2b, and DNR2b shares this information with DNR2a, the local gateway in Region 2a has to store all the bundles. This local gateway also needs to query DNR2a periodically to obtain new information where to forward those stored bundles destined to H2. Thus, we see that there are two key differences between our approach and Mobile-IP: (a) visiting DNRs cache location update information so that local gateways can query nearby DNRs for new forwarding information rather than having to rely on the Home Agent to supply such information, (b) local gateways need to store bundles and query local registrars for updated information to forward the stored bundles.

5. PARTITION DETECTION

The intermittent connectivity that necessitates disruption-tolerant networking affects not only the availability of an instantaneous route between communicating nodes, but also the ability to route within the DTN framework. Links connecting nodes or networks may disappear entirely as a node moves or is destroyed. At one or more levels, a DTN needs to recognize link failures and take appropriate action based on that change in network topology.

In EDIFY, the discovery of neighboring networks is provided through periodic neighbor discovery broadcasts, and the resulting heartbeat messages generated between neighbors generates a signal whose absence signifies the state change of the DTN link, and eventually leads to a change in routing. (This process may be somewhat slower when the link has a pattern of intermittent connectivity.) When that change in routing is the result of a partitioning of two networks that used to be connected, the DTN store and forward mechanism will typically retain messages destined for the disconnected network in the hope that a future connection can be made, until those messages expire, generating an error report to the sender.

Partitioning is not solely a problem between networks. In mobile ad hoc networks, member nodes can move out of range, be turned off, or fail. In addition, subsets of the network can deliberately (or accidentally) move in en masse. Thus, it is necessary to recognize and manage node failures and disconnections, as well as group partitionings. We manage individual node failures and movement through the need to periodically re-register with the local DNR (e.g., soft state), and via location updates to the home DNR when remote.

Given the dynamic nature of our ad hoc network topology, an efficient partition discovery (and perhaps prediction) mechanism is needed. Two approaches to partition discovery could be applied here. The first is a distributed technique, proposed by Ritter et al. [14], which selects a set of probing nodes (typically those with low degree) that exchange heartbeats. Active nodes additionally have adjacent buddies that take over upon node failure. As a result, this scheme is able to distinguish between node failure and partitioning. The second, proposed by Jorgic et al. [15] uses knowledge of the local network topology to recognize weakly connected subgraphs and predict impending partitionings. Jorgic et al. showed that local knowledge in the form of k-hop topological information can find critical nodes and links effectively.

Note that an ad hoc network may be partitioned into different subgroups and these different subgroups may operate independently of the others without much impact. However, for the purposes of DTN identity and thus routing, separate groups may need to have separate identities and infrastructure. Therefore, in EDIFY we assume that one of these two mechanisms is utilized to detect and respond to partitionings. In either case, when a node believes that a partitioning has occurred, it starts the process to self-organize by forming a new subgroup and electing a subgroup DNR. This process happens autonomously and recursively over time, if needed. We also assume that when dynamically created subgroups are rejoined for a sufficient amount of time, they retire their subgroup status and operate as members of the supergroup.

6. PERFORMANCE ANALYSIS OF MOBILITY MANAGEMENT SCHEME IN WIRELESS DTNS

In this section, we perform back-of-the-envelope analysis on a simple mobility scenario to determine what fraction of the link available time a node can use for data transfer after performing location update in an environment where only opportunistic links are available. The performance metrics we consider are (a) useful utilization which is defined as the fraction of connectivity intervals that can be used for data transfer after performing location updates, (b) the latency (delay) it takes to perform location update procedure, and (c) the number of overhead messages that are generated by the mobility management scheme that we design.

The parameters in our analysis are as follows:

- R = Residence time of a mobile in an area assumed to be exponentially distributed with mean $\tilde{R}$
- lT = available time of a particular communication link every T seconds. We assume that the communication link is opportunistic and is available for lT seconds every T seconds.

- d = completion time for visiting DNR registration. When the link is available, it takes d seconds to transfer the message across one DTN hop.
- H = number of hops before reaching a DNR. The variable, H, follows either (a) a truncated geometric distribution with a mean of E[H] (we refer to this as Model 1). E[H] is a function of two parameters, p and M where M is the maximum number of hops in the network, or (b) an adjusted truncated geometric distribution where one can adjust the probability of the single hop scenario (this probability is referred to as m) to reflect a topology that favors a single hop scenario. We refer to this as Model 2. The probabilities for other hop counts are set to the truncated geometric distribution value but adjusted such that the sum of these values (except hop=1) sums up to (1-m).

Based on our assumptions and the parameters we use, it is easy to derive that

- E[D|k], the expected delay required to perform location update to a particular DNR given the fact that it takes k hops for the location update information to traverse, = (d+T(1-l))*k
- Using model 1, the expected number of hops, E[H], can be computed as

$$E(H) = \frac{1-(1-p)^M - pM(1-p)^M}{p(1-(1-p)^M)}$$

Using model 2, the expected number of hops, E[H], can be computed as

$$E(H) = m - \left(\frac{(1-m)}{(1-(1-p)^M - p)}\right)p + \left(\frac{(1-m)}{(1-(1-p)^M - p)}\right)\left(\frac{1-(1-p)^M - pM(1-p)^M}{p}\right)$$

The expected latency of performing location update, E[D] can be derived as

$$E(D \mid H = k) = k\left(d + \frac{T}{2}(1-l)\right)$$

After some manipulation, one can show that for Model 1 & Model 2,

$$E(D) = \left(d + \frac{T}{2}(1-l)\right)E(H)$$

Assuming that the mobile host remains in its visiting area for R seconds, then the useful data transfer time, $U = \left\lfloor \frac{(R-D)^+}{T} \right\rfloor * lT$. The available data transfer time is $A = \lfloor R/T \rfloor * lT$ So, as in [10], we define the useful utilization, u, as U/A. We resort to simulation to evaluate this metric for the two models we used.

We built a simple simulator to see if the simulation results we obtain will match closely with the results obtained using the above analytical model. Using an example shown in Figure 3, we illustrate what our simple simulator does. In our simulator, we assume many registration events happen. For each registration event, we randomly pick the number of hops the registration message needs to traverse before reaching the DNR. Let us assume that the registration needs to traverse the links n1-n2, n2-n3, and n3-n4 before it reaches the DNR at n4. The simulator assumes that the on/off patterns of the links are not synchronized. If the message arrives at node 2 but the link n2-n3 is down, then the message will be queued and delivered only when the link is available. With this simulator, we can evaluate the location update latency. The simulator also generates a resident time. Once the location update is successful, then the remaining resident time will be useful data transfer time and hence we can compute the average useful utilization over many registration events. With this simulator, we can evaluate how good our

analytical model is. Our analytical and simulation results for the average location update latency are plotted in Figures 4 & 5.

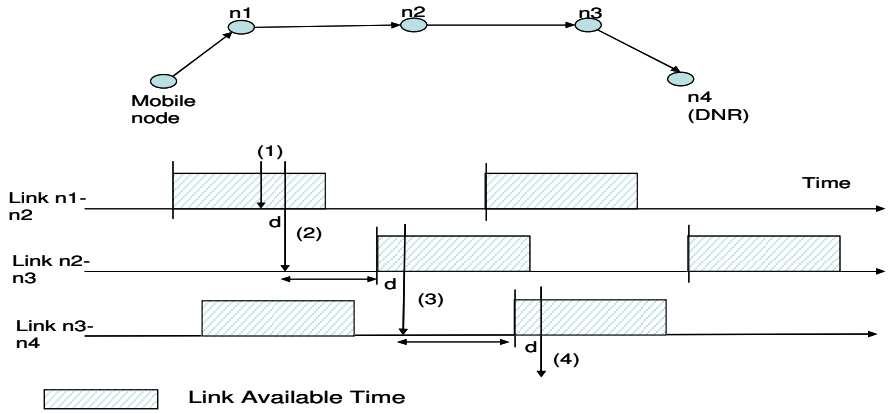


Figure 3: Asynchronous Link On/Off Pattern in Simulator

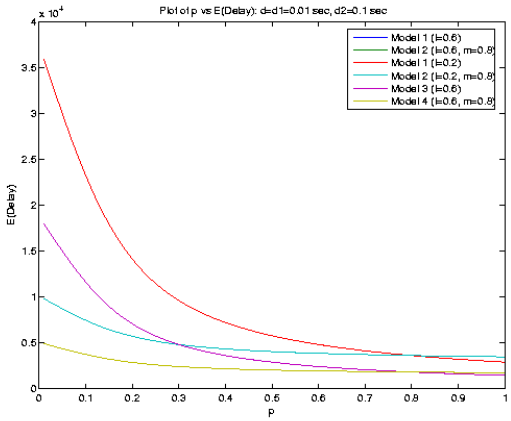


Figure 4: Expected Delay vs. p (using analysis)

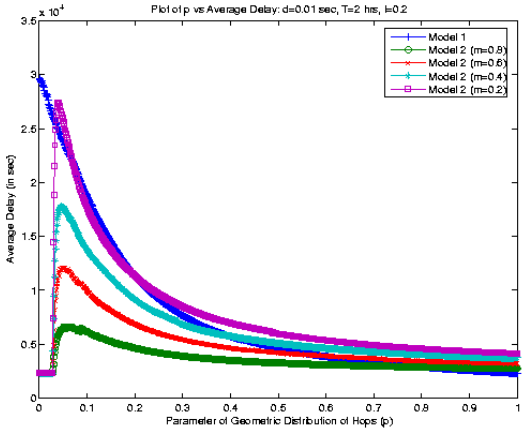


Figure 5: Expected Delay vs. p with l=0.2 (via simulations)

Figs. 4 & 5 plot the expected delay versus the parameter p using approximate analysis and via simulations. M is set to 25 in these plots. As one can observe in Figs. 4 & 5, the expected delay in performing the location update procedure is smaller for model 2 since most of the scenarios in model 2 will be the single-hop scenario. Fig. 5 indicates that the approximate analysis we have for expected delay matches closely with the simulation results. The useful utilization for both models when l=0.2 and l=0.6 are plotted in Figs 6 & 7 respectively. For these plots, we set T=7200 seconds (2 hours), and mean residence time=3.47 hours. For Model 1, we set p=0.2 (equivalent to E[H]=4.9 hops) so one can observe that when m=0.2, the curve for Model 2 will be almost the same as the curve for Model 1. When m is larger than 0.2 (equivalent to single hop scenario being more probable than what is predicted using the truncated geometric distribution), the useful utilization improves since it takes shorter time to complete the location update procedure and hence more time can be used for the data transfer.

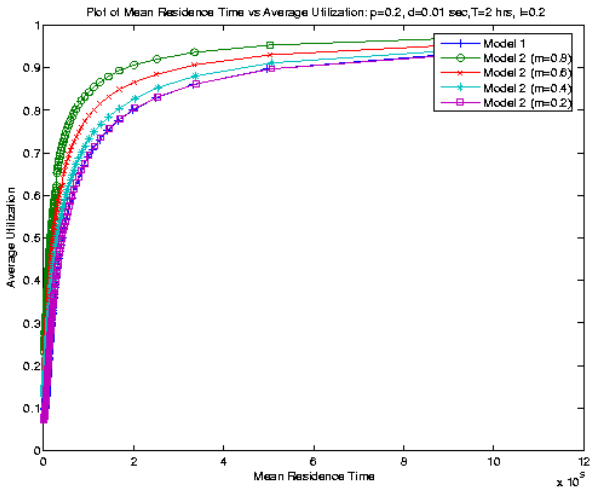


Figure 6: Average Useful Utilization versus Mean Residence Time (l=0.2)

We also have other models where the completion time for location update may be d1 second for the single hop scenario and d2 second for the scenarios with more than 1 hop. We referred to such a model as Model 3 if we use truncated geometric distribution for the variable H, and Model 4 if we use the adjusted distribution as in Model 2 for the variable H. The derived E[H] for models 3 & 4 are as follows:

For Model 3,

$$E(D) = (d2 + \frac{T}{2}(1-l))E(H) + \frac{p}{1-(1-p)^M} \times (d1 - d2)$$

For Model 4,

$$E(D) = (d2 + \frac{T}{2}(1-l))E(H) + m(d1 - d2)$$

Our further investigation indicates that the impact of different location update completion time is small since the major component of the expected latency time is the periodicity of the opportunistic link, T.

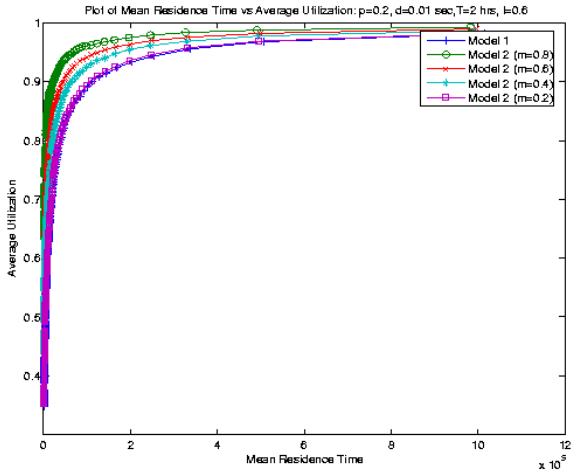


Figure 7: Average Useful Utilization vs. Mean Residence Time (l=0.6)

7. PERFORMANCE EVALUATION FOR INTEGRATED WIRED/WIRELESS DTNS

The performance evaluation in the earlier section concentrates on scenarios where the links between nodes follow on/off patterns. However, in some real life scenarios, some DTN nodes may be formed over the wired Internets so we would like to build a more complex simulator where we can simulate dif-

ferent scenarios including the integrated wired/wireless DTNs. We describe below the more complex NS-2 based simulator that we have built.

7.1 Descriptions of NS-based Mobility Management Simulator

7.1.1 Pre-configuration

Each DTN node maintains the following databases:

1. Adjacency (list of adjacencies/neighbors with link status information)
2. Visited List (list of groups the node has visited and registered with)

In addition to the above-mentioned databases, a DNR maintains the following databases:

1. Members (details of members and visitors of the group)
2. DNR Neighbors (details of the 1-hop DNR peers)
3. Location Update Cache (location information of nodes)

Each DTN node in the system is pre-configured with its own DTN name, the DTN name of its home DNR, its node type (regular / DNR) and whether it is a wired node or a wireless node. Wired DNRs are pre-configured with information about other wired DNRs that form the DNR overlay. We assume that the DNRs run a DSDV-style routing protocol so each DNR has a topological view of the DNR overlay which is another database (DNR peers database) that every DNR maintains.

7.1.2 Network Bootstrapping

On startup, each node periodically sends out a “Hello” bundle to its 1-hop (underlying network hop) neighbors with TTL value set to 1. DNRs may choose to set the TTL field higher (currently set to 2). When a DTN node hears a “Hello” from any other DTN node, it adds / updates its 'Adjacency' database with this node information. It also registers the link to the particular node to be active. Periodically each node looks up its 'Adjacency' database and purges out inactive links (using time-stamp information).

7.1.3 Periodic DNR beacons

We assume that each DNR broadcasts a beacon periodically. All DTN nodes that can hear such a beacon can store information about the announcing DNR in its cache but they will not register with that announcing DNR unless they have lost contacts with their individual home DNR.

7.1.4 DNR Discovery Process

When a DTN node stops hearing from any of the DNRs with which it has registered, it looks up its 'Adjacency' database for an entry of a DNR. If there is such an entry, the DTN node will send a registration bundle to that DNR. If there are multiple DNR entries in its adjacency database, the DTN node will send a registration bundle to the DNR which sends the latest beacon announcement message. If there is no such entry, it sends out a “DNR Discovery” bundle with TTL value set to 1 (underlying network hops). This process repeats itself with increasing TTL until an entry is found, at which time the mobile node sends a “Registration Request” bundle to the DNR.

If the DTN node receiving a “DNR Discovery Request” is a DNR, then it sends out a “DNR Discovery Response” bundle to the requesting node populated with its own information. Once the node that issues the DNR discovery message receives a “DNR Discovery Response” bundle, it caches that information.

7.1.5 Registration Process

When a DNR receives a “Registration Request”, it creates a new locally unique visiting identifier for the requesting node. It adds the requesting node's information along with the new local identifier into its 'Members' database. It then sends a “Registration Response” bundle to the requesting node granting the node visiting rights and an identity in the local group. When the requesting node receives the “Registration Response”, it updates its 'Vis-

ited List' database with its new identity information. It now advertises its new local identifier in subsequent "Hello" bundles to its neighbors.

7.1.6 Location Update Process

A "Location Update" bundle of a mobile node is sent out by a DNR when a mobile node registers with the DNR. There are two schemes which one can use to disseminate the new location information of this mobile node, namely the Home-DNR scheme and the Neighbor scheme [16] which we describe below:

1. Home-DNR Scheme
2. Neighbor Scheme (with a given k)

In the Home-DNR scheme, the visiting DNR sends a "Location Update" of the mobile node only to the Home DNR of the mobile node. In the Neighbor scheme, the visiting DNR sends out the "Location Update" of the mobile node with a TTL value of the bundle set to k. It sends this bundle to its 1-hop DNR neighbors. When a DNR receives a "Location Update" bundle, it caches this bundle in its 'Location Update Cache', decrements the value of TTL by 1, and if TTL > 0, forwards this bundle to its 1-hop DNR neighbors. The DNRs use these "Location Update" bundles to update their databases with the latest location information of their members/visitors.

7.1.7 Query Resolution Process

A DNR determines the current location of a mobile node by sending out a "Node Query" bundle. When a DNR receives a "Node Query" for a mobile node, it sends back a "Node Query Response" to the requesting DNR.

Types of "Node Query Response":

- Success: the receiving DNR looks up its visitor list and see that the mobile node is currently registered.
- Hints: the receiving DNR returns the identity of the DNR (referred to as the hinted DNR) that most recently sent out a location update of this node. The receiving DNR finds this information from its location update cache. Currently, we assume that the location update cache timeout is a large value. The querying DNR will then issue a unicast node query to this hinted DNR to find out if the mobile is still visiting at its group.
- Failure: the DNR has no information about this mobile node.

Corresponding to the Location Update schemes, we have the Home-DNR scheme and the Neighbor Scheme (with a given k) of query resolution. In the Home-DNR Scheme, the querying DNR sends a "Node Query" of the mobile node to the Home DNR of the mobile node. This is natural as we expect the mobile node's latest location information to be found at the Home DNR of the node (corresponding to the Home-DNR Scheme of Location Updates).

In the Neighbor scheme, the querying DNR checks its location update cache to see if it can find any DNRs that recently announced the location of this mobile node. If such an entry exists, it will send a Node Query bundle to that DNR. Otherwise, the querying DNR broadcasts the "Node Query" to all its 1-hop DNR neighbors. The idea here is that instead of sending the query to the Home DNR of the node, we check to see if any of the querying DNR's neighbors have the location information of the mobile node.

If the "Node Query Response" is a Success (or Failure), then the node query resolution succeeds (or fails). If the response includes the name of a DNR that last sent out a Location Update of the bundle, then the querying DNR sends the "Node Query" to this particular DNR. This process continues till either the node resolution succeeds or all the DNR 'hints' have been tried (resolution failed).

7.2 Experimental Simulation

7.2.1 Configuration

We used the BRITE Internet topology generator [13] to create a wired network of 100 nodes. We then randomly designate 50 of these nodes to be DTN nodes (40 regular and 10 DNRs).

Number of non DTN wired nodes	50

Number of regular DTN nodes	40
Number of DNRs in overlay	10
Average degree of the overlay nodes	2.5
Diameter of the overlay	5 hops

Table 1: Simulation Parameters

We then form 10 DTN groups each having 1 DNR and 4 regular DTN nodes. To form the DNR overlay, we set a threshold on the number of underlying hops. If a DNR node is within this threshold number of hops from another DNR, then these DNRs are neighbors in the DNR overlay.

We create 20 mobile DTN nodes pre-registered with one of the 10 groups (2 mobile nodes per group). We make use of the Zebranet trace to create a semi-realistic mobility model to drive the movement pattern of these 20 mobile nodes. The simulation area is confined to a 3000m x 3000m area. The simulation runs for a period of 1500 minutes. To study the Query Resolution process, we make each of the 10 DNRs randomly pick one of the 20 mobile nodes to query.

Metrics used:

- Number of Location Updates – this is a count of the number of registration events that happen during the simulation duration.
- Number of Location Update messages triggered – this is the number of location update messages that is triggered by each registration event.
- Cost of Location Updates – this is measured as the total number of hops traversed by all location update messages that are triggered by each registration event.
- Average Query Success Ratio – this is the fraction of queries that result in successful resolution.
- Average Delay per Success - this is a measure of the number of hops that a successful query needs to traverse before getting the response back to the querying DNR.

We compare the results obtained for three cases: Home-DNR Scheme, Neighbor Scheme (k=1) and Neighbor Scheme (k=3). We vary the querying rate (0.2, 0.4, and 1.0) and the transmission range of the nodes (250m, 300m, and 350m).

7.2.2 Experiments

In order to study the cost of the different schemes of Location Updates, we simulated all three scenarios: (a) Home-DNR Scheme, (b) Neighbor Scheme (k=1), and (c) Neighbor Scheme (k=3).

For each setting, we determine the

- Number of Location Update events
- Number of messages these Location Update events trigger
- Cost of all Location Update events.

For the Home-DNR Scheme, the number of messages triggered will equal the number of Location Update events, while for the Neighbor scheme, this number will depend on the value of ‘k’ used and the out-degree of the DNR sending out the Location Update.

The cost of Location Updates is determined by summing over the total number of underlying network hops that these Location Update messages traverse.

We repeated this experiment for three different transmission range settings of the nodes (250m, 300m and 350m). We anticipate that with larger transmission range, a mobile node will hear more DNRs and will stay with a visiting DNR longer.

Transmission Range: 250m

	Home-DNR Scheme	Neighbor Scheme (k=1)	Neighbor Scheme (k=3)
# of Location Updates	233	233	233
# of Location Update Messages triggered	233	523	4166
Cost of Location Update Messages	982	1365	10373

Transmission Range: 300m

	Home-DNR Scheme	Neighbor Scheme (k=1)	Neighbor Scheme (k=3)
# of Location Updates	287	287	287
# of Location Update Messages triggered	287	636	5008
Cost of Location Update Messages	1215	1673	12478

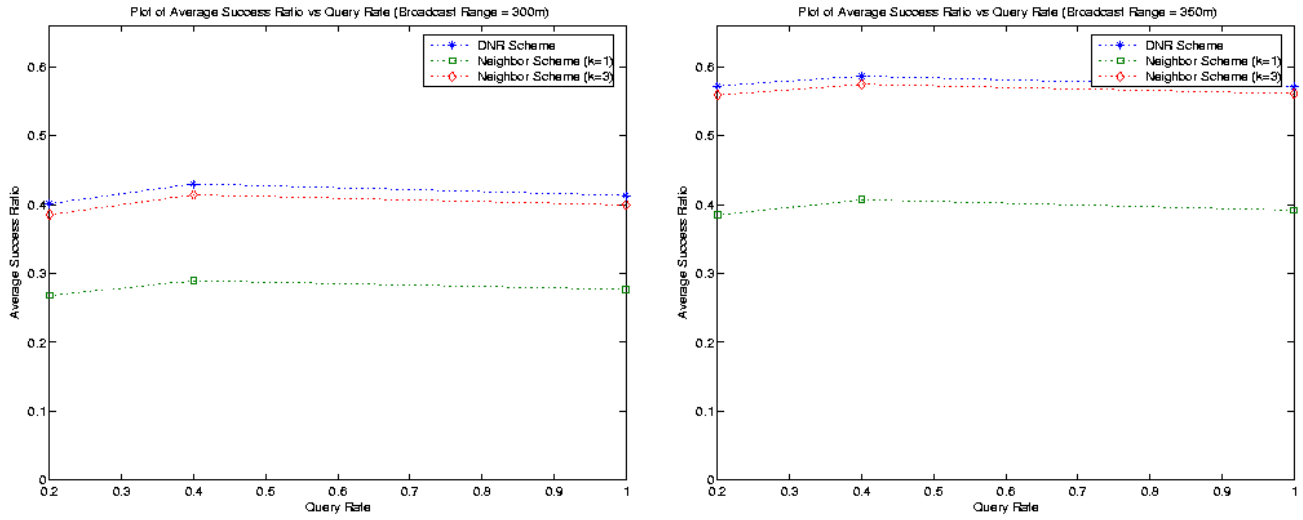
Transmission Range: 350m

	Home-DNR Scheme	Neighbor Scheme (k=1)	Neighbor Scheme (k=3)
# of Location Updates	356	356	356
# of Location Update Messages triggered	356	779	6144
Cost of Location Update Messages	1506	2043	15305

Table 2: Overhead and Cost of Location Updates with different Transmission Ranges

We notice that, in all of the above cases, the Home-DNR scheme results in the least number of Location Update messages and the lowest cost. But one observes that the Cost per Location Update Message in the Home-DNR Scheme (~4.2) is the highest among the three schemes. Hence, though the Neighbor Scheme involves many more Location Update messages (increases further with increasing k), there is not a proportional increase in the cost when compared to that of the Home-DNR Scheme. This is obvious as in the Neighbor scheme, the visiting DNR sends / forwards the Location Update messages to all its 1-hop DNR neighbors, which by definition, are closer to it than other DNRs in the overlay.

Next, we study the Query Resolution performance of the Home-DNR Scheme and the Neighbor Scheme.



(a) (b)

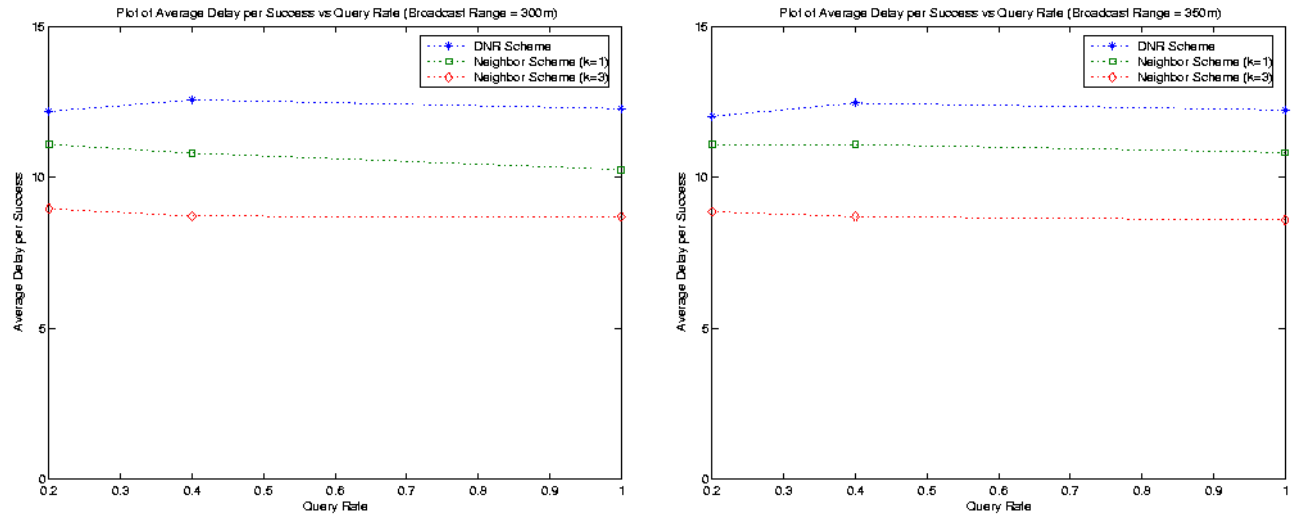
Figure 8: Average Query Success Rate vs. Query Rate with different Transmission Ranges.

The above figures show plots of the Average Success Ratio vs. the Query Rate. For each query rate, the Query Success Ratio is averaged over 5 trials (each trial corresponds to a random generation of node queries in the simulation).

We notice that the Home-DNR Scheme (denoted as DNR in the figures) performs the best, as it achieves the highest average success ratio.

The Neighbor Scheme (k=3) performs almost as well as the Home-DNR Scheme. This can be explained by the following: since the average out-degree of a DNR in the DNR overlay is 2.5, the Location Updates propagate to a significant fraction of DNRs.

The Neighbor Scheme (k=1) has the lowest performance. This is expected as the Location Updates following this scheme propagate only to the 1-hop DNR neighbors of the DNR sending the Location Update (a small portion of the overlay). Note that in both Neighbor Scheme (k=1) and Neighbor Scheme (k=3), the querying DNR queries only its 1-hop DNR neighbors. The value of 'k' is used only in the Location Update process.



(a) Transmission Range = 300 m (b) Transmission Range = 350 m

Figure 9: Average Delay/Success vs Query Rate

Figures 9(a) and (b) show plots of the Average Delay per Success vs. the Query Rate with two transmission ranges. For each query rate, the Delay per Success is averaged over 5 trials (each trial corresponds to a random generation of node queries in the simulation).

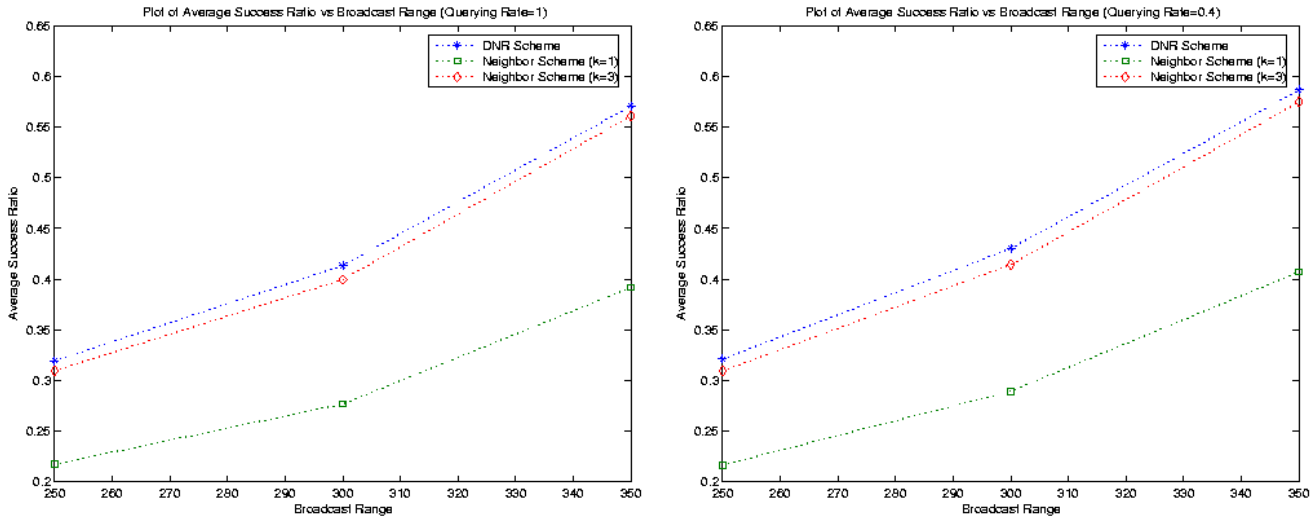
Here, we notice that the Home-DNR Scheme has the highest Average Delay per Success. This is because of the long resolution paths that a query takes (first to the Home DNR, and if required, then to the DNR that last sent a Location Update etc.)

The Neighbor Scheme (k=3) has the lowest Average Delay per Success. Neighbor Scheme (k=3) has a lower Average Delay per Success than the Neighbor Scheme (k=1). This is expected, since we assume that the cache time-out value is set quite high so the querying DNR sometimes has to follow multiple hints before finding the right visiting DNR when k=1.

From both figures, we observe that the Neighbor Scheme offers a lower Average Success Ratio but with the advantage of lower Average Delay per Success over the DNR Scheme.

We also study the behavior of the schemes by varying the transmission range of the nodes. An increased transmission range has the following effects:

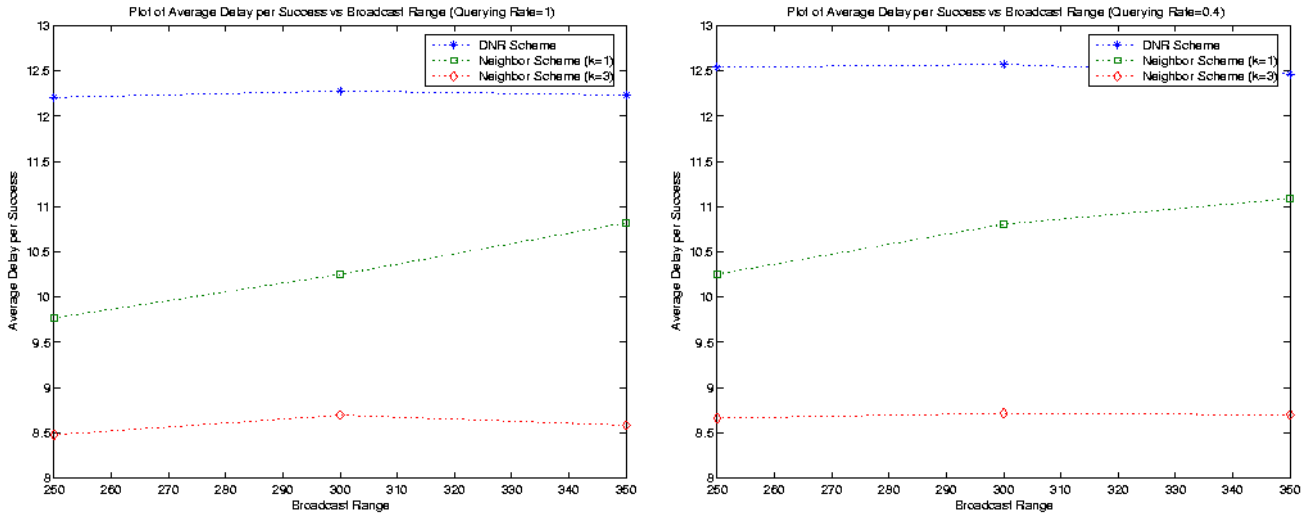
- Increase the number of DNR contacts seen by a mobile node
- Increase the contact duration of the mobile node and a given DNR
- Decrease the inter-contact duration of the mobile node and a given DNR



(a) Query Rate= 1 (b) Query Rate = 0.4

Figure 10: Query Success Ratio vs. Transmission Range at different query rates

Figures 10(a) & (b) show plots of the Average Success Ratio vs. Transmission Range (denoted as Broadcast Range in the figures). As expected, we observe that the Average Success Ratio of each scheme increases with increasing transmission range.



(a) Query Rate = 1 (b) Query Rate = 0.4

Figure 11: Average Delay/Success vs Transmission Range

Figures 11(a) & (b) show plots of the Average Delay per Success vs. Transmission Range. The Average Delay per Success remains fairly constant across different transmission ranges in the Home-DNR Scheme.

In the Neighbor Scheme (k=1), we see an increase in the Average Delay per Success with increasing transmission range. This is because with an increasing transmission range, a mobile node tends to stay with a visiting DNR for a longer period of time. In the case of a shorter contact duration between the mobile node and the DNR, it is more likely that the resolution of the query will lead to Failure (the mobile node has left the group). But with an increase in contact duration, this becomes less likely and there are more instances of successes, and notably, successes with long resolution paths, which in the former case would have resulted in failures.

In the Neighbor Scheme (k=3), we do not see a significant difference in the Average Delay per Success across different broadcast ranges. Again, with a value of k=3, we expect the latest Location Update of the mobile node to be available more frequently in the 1-hop DNR neighbors of the querying DNR.

8. CONCLUDING REMARKS

In this report, we have presented a mobility management scheme for DTNs. We have identified three major mobility scenarios that need to be addressed by the designed mobility management scheme. We then provide some back-of-the-envelope analysis to evaluate the latency involved in performing location updates in an environment where the communication links are only available for a duration of l*T seconds every T seconds and that the nearest DNR may be H hops away from the mobile host. By putting more weight on the probability of single hop scenario, we are mimicking the near movement scenario. Our preliminary study indicates that the location update latency and useful utilization results obtained via our simple analytical model match closely with the simulation results we obtain. Such a simple model allows us to obtain useful insights into different mobility scenarios in DTNs. In addition, we have built a more complex NS-based Mobility Management simulator to evaluate different location update schemes. Our simulation results for the integrated wired/wireless DTNs indicate that the home-DNR scheme produces the smallest transmission overhead and cost for the location updates as well as the highest query success rate but the home-DNR scheme suffers the largest query response time. The neighbor scheme with k set to slightly more than half the network diameter produces more transmission overhead and cost for the location updates and lower query success rate but it gives the smallest query response time. We believe that an adaptive scheme that switches between home-DNR scheme and neighbor scheme will be required to achieve reasonable transmission overhead, query success rate and query response time. We intend to explore more scenarios and make some recommendations in the coming months.

9. REFERENCES

[1] P. Juang, H. Oki, Y. Wang, M. Martonosi, L-S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet", Proceedings of ASPLOS-X, Oct, 2002.
[2] Disruption Tolerant Networking http://www.darpa.mil/ato/solicit/DTN.
[3] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss, "Delay-tolerant networking: an approach to interplanetary internet", IEEE Communication Magazine, 41:128-136, 2003
[4] D. Johnson, D. Maltz, "Dynamic source routing in ad hoc wireless networks", Proceedings of Sigcomm, Aug, 1996.
[5] C. Perkins, E. Royer, "Ad-hoc On Demand Distance Vector Routing", Proceedings of IEEE workshop on mobile computing systems and applications, pp 90-100, Feb, 1999.
[6] M. Chuah, L. Cheng, B. Davison, "Enhanced Disruption and Fault Tolerant Network Architecture for Bundle Delivery (EDIFY)", will appear in Proceedings of Globecom, 2005.
[7] K. Fall, "A delay-tolerant network architecture for challenged Internet", proceedings of ACM Sigcomm, 2003.
[8] Connexion By Boeing, http://www.connexionbyboeing.com
[9] A. Snoeren, "A Session-based Approach to Internet Mobility", PhD Thesis, MIT, http://www.cse.ucsd.edu/~snoeren, Dec 2002.
[10] A. Seth et al, "An architecture for Tetherless Computing"
[11] C. Perkins et al, "IP mobility support for IPv4", RFC3344, Aug, 2002
[12] A. Dutta, S. Madhani, W. Chen, O. Altintas, and H. Schulzrinne, "Fast-handoff schemes for application layer mobility management", Proceedings of PIMRC, 2004.
[13] A. Medina, A. Lakhina, I. Matta, and J. Byers. "BRITE: Universal Topology Generation from a User's Perspective", Technical Report BUCS-TR2001-003, Boston University, 2001. Available at http://www.cs.bu.edu/brite/publications.
[14] H. Ritter, R. Winter, and J. Schiller, "A partition detection system for mobile ad hoc networks", Proceedings of IEEE SECON, October 2004.
[15] M. Jorgic, I. Stojmenovic, M. Hauspie, and D. Simplot-Ryl, "Localized algorithms for detection of critical nodes and links for connectivity in ad hoc networks", Proceedings of 3rd IFIP MED-HOC-NET Workshop, 2004
[16] Spindle Project Team, BBN, "Draft SPINDLE Architecture", unpublished, May, 2005.