Structuring Pervasive Services in Infrastructureless Networks[*]

Anandha Gopalan, Taieb Znati and Panos K. Chrysanthis
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260, U.S.A
Email: {axgopala, znati, panos}@cs.pitt.edu

Abstract

Realizing the potential of pervasive computing will be predicated upon the availability of a flexible, mobility-aware infrastructure and technologies to support seamless service management, provisioning and delivery. Despite advances in routing and media access control technologies, little progress has been made toward large-scale deployment of services and applications in pervasive and ubiquitous environments. The lack of a fixed infrastructure, coupled with the time-varying characteristics of the underlying network topology, makes service delivery challenging. This paper addresses the need for new service models and proposes a mobility-aware service architecture to support pervasive service provisioning and delivery in an infrastructureless environment.

The proposed methodology centers around the concepts of virtual homes and mobility signatures, and decouples service discovery from current location of the server. We discuss the main functionalities of the proposed architecture and describe the underlying service discovery mechanisms and protocols. We also present a novel, piece-meal, location-driven traffic forwarding algorithm, and analyze its performance using a simulation-based study. The results show that the algorithm exhibits good performance for a variety of network environments.

1. Introduction

Advances in wireless technology and portable computing along with demands for greater user mobility have provided a major impetus toward development of an emerging class of self-organizing, rapidly deployable network architectures referred to as ad-hoc networks. Mobile Ad-hoc networks, which have proven to be useful in small-scale military applications, are expected to play an important role in future commercial settings where access to a wired network is either ineffective or impossible. Despite the advantages provided by ad-hoc networks, however, the large-scale deployment of services and data-intensive applications over these networks has been lagging. This is mostly due to the lack of an efficient and secure architecture to support the basic functionalities necessary to enable a computation platform for pervasive computing and services.

[*]This work has been partially supported by an NSF-ITR award ANI-0325353 and by NSF award ANI-0073972

Mobile ad-hoc networks are unique, in the sense that, a participating ad-hoc node can function both as a host and as a router, thereby dynamically creating paths between clients and servers. Unlike a fixed wireless network, however, locating a node becomes difficult as nodes may exhibit high levels of mobility. Several challenges must, therefore, be addressed in order to develop an efficient, mobility-aware service architecture to support service delivery in a robust and scalable manner. These challenges directly impact the main capabilities necessary to support the basic operations of services and applications in a pervasive computing environment. These capabilities include:

- Service registration,
- Service discovery,
- Mobile server location, and
- Traffic routing and forwarding.

Service registration is the mechanism by which a server, managing a collection of related resources, exports its interface and binding information to the network. The interface typically describes the functionalities of the managed resources and the set of operations that clients can invoke to access these resources.

Service discovery is the mechanism by which clients discover the services that are available in the network. Typically, clients attempt to import binding information related to a server by looking up the name of the server of interest with the registration server. If a match occurs, the registration server returns the corresponding binding information.

Mobile server location is the mechanism by which network clients identify the current position of the mobile server. Since a mobile server can move frequently, an efficient mechanism must be in place whereby clients can locate, with high probability, the server in the network, including the registration server.

Traffic routing and forwarding is the mechanism by which application and control traffic reaches its intended destination. The strategy used to forward traffic efficiently must take into consideration the time-varying dynamics of the network, server mobility and power-consumption. The tradeoffs between these important design factors and network characteristics must be recognized and alternatives carefully evaluated.

Accommodating the mobility, both of clients and servers, clearly requires mechanisms and information which go beyond the binding information required for a typical client-server architecture in wired networks. In addition to an access interface, the server must also register its location information and its mobility profile in order to facilitate interaction with clients "anytime, anywhere". This information, however, changes dynamically, as the servers move from one location to another. Efficient mechanisms must, therefore, be in place to update this information as servers move. The goal of this paper is to address the fundamental design issues of service infrastructure for pervasive computing and provide a comprehensive solution which takes into consideration node mobility and resource constraints.

The main contribution of the paper is a novel service-architecture that allows the deployment of pervasive services and applications in ad-hoc networks. The proposed architecture is scalable and robust and does not impose any location restrictions on the servers and services. The basic tenet of the proposed architecture revolves around the concepts of *virtual home* and *mobility signature*. A node's virtual home is a physical area in the network where the node is most likely to be located. As such, the virtual home acts as an anchor for the node and can be viewed as a "rendezvous" point between a server and its client. When the server moves out of its virtual home, it leaves behind a *mobility signature*, which provides "hints" about the server's current location. This enables clients to locate and initiate interaction with the server outside of its virtual home. The mobility signature is maintained by a selected set of *proxy* nodes, through renewed recruiting as proxies move out of the virtual home. The *main* advantage of this approach is that each node selects the mobility prediction model, which is deemed most appropriate to the current activity, rather than than using a network-wide mobility model which may not be applicable to different itineraries and situations.

The second contribution of this paper is an algorithm for information dissemination. This algorithm, called PMLD forwards traffic in a piece-meal, location-directed manner. To limit flooding in the network, PMLD uses the knowledge about the location of the source and the direction of the destination to forward traffic in a cone-shaped manner towards the destination. The intermediary node to forward traffic is chosen by using a priority-based scheme that imposes a priority on the neighboring nodes in a way, such that nodes which are more in line with the direction of the destination have higher priority to forward the message. This reduces the delay that traffic suffers on its way towards the destination. As traffic progresses towards the destination, the highest priority node responsible for forwarding traffic calculates a new cone and re-iterates the process.

The rest of the paper is organized as follows: Section 2 talks about the related work in this area, Section 3 provides details of the system topology, Section 4 details the different components of the proposed service-architecture. Section 5 details the simulations and the results and Section 6 concludes the paper and identifies areas of future work.

2. Related Work

This section details the work related to this paper.

Service discovery provides an interface by which clients and servers discover the services that are available in the network. There have been some protocols for service location and discovery that have been developed for LANs, namely: Service Location Protocol (*SLP*) [3] and Simple Service Discovery Protocol (*SSDP*) [19]. *SLP* relies on agents to search for and locate services in the network: a *user agent* is used on behalf of users to search for services, while a *service agent* advertises services on behalf of a server and finally, a *directory agent* collects the advertisements sent out by the *service agent*. *SSDP* uses HTTP UDP on the reserved local multicast address *239.255.255.250* and the $SSDPport$ while searching for services. The ideas of SLP and SSDP cannot be directly applied to ad-hoc networks due to their reliance on an existing network structure.

The Grid location service (GLS) [8] provides distributed location information service in mobile ad-hoc networks. GLS combined with geographic forwarding can be used to achieve routing in the network. A node X "recruits" nodes that are "closest" to its own ID in the ID space to act as its location server. Similar to GLS, our scheme also uses a distributed scheme to store location information, but instead of using *one* node as a location server, a group of nodes are selected from within an area to act as the location server. The information is stored in a manner such that only k out of N fragments are necessary to *re-construct* it.

[7, 6, 18] use the concept of home agents (or) home regions. Each node in the network is mapped to an area (using a hash function) in the network that is designated as its home agent (or) home region. The home region holds the location information about the mobile nodes which map to this location. A mobile node updates its location information by sending updates to its home region. In our scheme, a mobile node does not keep updating its *virtual home*, rather it leaves a trail behind that can be used by other nodes to locate it. Also, the *virtual home* is used as a "rendezvous point", where the node is most likely to be located.

Due to an infrastructureless architecture, routing in an ad-hoc network is achieved by relying on the nodes in the network. The routing protocols can be classified into three categories: pro-active [14], re-active [9, 13] and hybrid [1, 5]. There is a new class of routing protocols for ad-hoc networks that rely on the position of a node in space rather than on the topology of the network. These protocols rely on the fact that the nodes in the network know their location (using a service similar to GPS [2]). This is used to optimize the routing protocol by sending the routing information in a direction that is *closer* to the destination rather than broadcasting it. Examples of location-based routing protocols are: LAR [10] and DREAM [16]. We differ from DREAM and LAR by not flooding the network with location updates; rather, messages are forwarded by intermediary nodes on a piece-meal basis (where the position of the destination is re-calculated and hence the direction of forwarding is changed to suit the direction of the destination). This leads to our message forwarding algorithm being scalable when compared to DREAM and LAR.

Mobile IP [4] was developed to facilitate mobile computing. The main idea was to have the ability for users to be able to take their computing environment along with them without having to change their configurations. The only requirement was that the user should have a connection to the internet in the new location. The mobility of the users is handled by allowing the mobile node to have two IP addresses: a *home-address* and a *care-of address*. The home-address of the node is its fixed IP address, while the care-of address is the address that the node acquires at its new location. Traffic intended for the mobile node (that was held up) is forwarded by its *home agent*, upon receipt of the node's care-of address. While mobile IP solves the problem arising due to the mobility of the nodes; services that are provided are not continuous, owing to the fact that the node has to register its care-of address with its home-agent. This paper addresses this short-coming to provide a *service-continuous* architecture that handles mobility by using a piece-meal, location-driven traffic forwarding algorithm.

3. System Topology and Infrastructure

3.1. Virtual Home

Consider an ad-hoc network covering a specific geographical area, denoted as A. We perceive this area to be divided into zones as shown in figure 1, ($A = \bigcup_i Z_i$, where Z_i is a zone). A zone for example, can represent neighborhoods or an administrative domain where geographical proximity facilitates communication between nodes of the zone. Each zone Z_i, contains one or more nodes whose *virtual homes* map to a neighborhood in Z_i. The virtual home of a node is defined as the physical area within a zone where the node is most likely to be located. A virtual home may contain one or more nodes and is centered around an *anchor* node, that is used by the nodes in that virtual home to determine their current location.

In each virtual home, a set of dynamically selected *proxy* nodes act as a *management information base (MIB)* that store and maintain mobility information about nodes in the virtual home. This mobility information is in the form of a vector (called the *mobility signature*) and consists of the expected direction and speed of travel of a mobile node.

The size of a virtual home (in terms of the number of nodes) plays an important part while designing this framework for a service-architecture for ad-hoc networks. A large virtual home results in a large number of nodes that are a part of this virtual home. This leads to a higher overhead in terms of managing the virtual home. Having a small virtual home has the advantage of a reduced overhead in managing the virtual home, but the lack of nodes in the virtual home may have an impact on the *MIB* by not having enough nodes to maintain the information about *mobility signatures*. In our scheme, the size of a virtual home is not constant and is dependent on the *MIB*. If the size of the *MIB* falls below a certain threshold[1], the mobile node tries to recruit nodes from its neighboring virtual homes to become a part of the *MIB* in order for the *mobility signature* to be maintained.

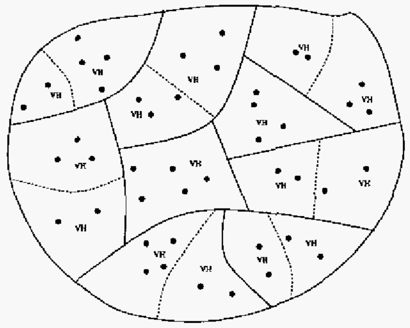

Figure 1. Partitioning the network into zones and virtual homes

4. System Architecture and Services

This section details the components of the system architecture. The proposed architecture tackles the important

[1]Optimal value for the threshold can be found using experiments.

problem of structuring pervasive services in an infrastructureless networks. There are several challenges that must be addressed to develop a service architecture to support pervasive computing. These challenges are related to the development of several capabilities necessary to support client-server interaction in this architecture. These capabilities include: service registration, service discovery, mobility management and data dissemination.

The interaction between the capabilities is shown in figure 2. Sections 4.1 - 4.5 explain these in more detail. The proposed architecture is flexible, scalable and robust and is able to adapt to the constantly changing network topology.

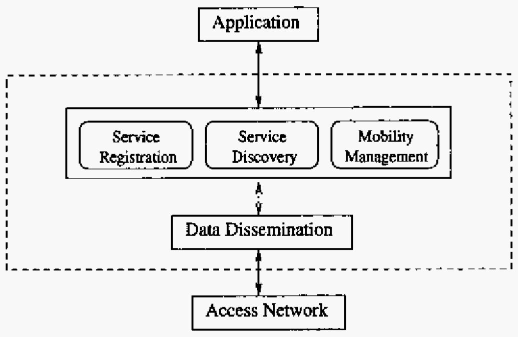

Figure 2. Components of the Architecture

4.1. Service Registration

Nodes must register their services with a *virtual home information server* (*VHIS*). These *VHIS*s form a DHT-based structure of mobile servers (along the lines of peer-to-peer networks [12, 15]). Every node that has a service, is associated with a *virtual home* which is determined by the hash value (based on a pre-defined hash function, known to the nodes in the network) produced by hashing the *service id*[2]. The *VHIS* within a virtual home is provided by a set of collaborative nodes that exist in that virtual home. Each *VHIS* is responsible for storing a portion of the hashed namespace. The *VHIS*s provide name service in a distributed and scalable manner. The server registers its node id, virtual home, the list of services that it offers and the list of its service interfaces that it exports. The server uses $PMLD$ (described in Section 4.5) to contact the *VHIS* it must register with.

4.2. Service Discovery

A client requiring a service from the network queries the *VHIS*, determined by hashing the corresponding service id into the virtual home of the *VHIS*. The selected *VHIS* resolves the requested service into the virtual home of the server(s) that offer(s) this service. The client can choose to select a particular server and this decision may be made based upon factors like past history, the distance to the virtual home of the server and the stability of the server. Once the client decides on a particular server, the client uses the knowledge about the virtual home of the server and $PMLD$ (described in Section 4.5) to interact with the desired server.

[2]We use this as a generic term, this could include the service description, service name or other attributes of the service.

4.3. Client-Server Interaction

After a client discovers the services that are available in the network (or has successfully queried for a service), it resolves the server (containing the required service) into its virtual home. A mechanism is required by which client-server interaction can be achieved. As mentioned before, nodes in the network can be mobile and the algorithm must take this into account. Algorithm 1 and figure 3 detail the procedure by which a client node C obtains a service from a server node S that is mobile.

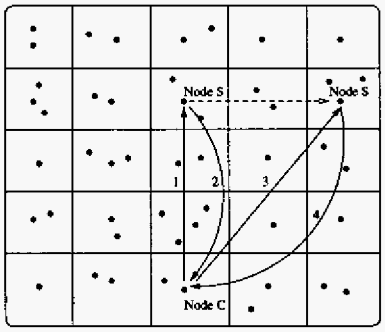

Figure 3. A mobile server

The components of the *mobility signature* $[t_0, V(t_0), D(t_0), P_V(t), P_D(t)]$ in algorithm 1 are:

- t_0: starting time
- $V(t_0)$: expected average starting speed.
- $D(t_0)$: expected initial direction.
- $P_V(t)$: Predictor for speed after t units of time since departure.
- $P_D(t)$: Predictor for direction after t units of time since departure.

In the case of node S not being in its *predicted* position, the nodes in the vicinity of this position hold on to the message until they hear from node S (in which case the message is transmitted) or there is a timeout (sufficiently large to incorporate in-accuracies in the *mobility signature*), after which the message is discarded. This ensures that S receives the message.

Algorithm 1 Handling server mobility

```
Input: Void
Output: Result
SERVER-MOBILITY()
(1)   C receives VH(S) from the VHIS
(2)   Using directional routing, C sends messages towards VH(S) (msg 1 in figure 3)
(3)   case S is in VH(S)
(4)       Client-server interaction is established between C and S
(5)   case S is currently not in VH(S)
(6)       The nodes in the VH(S) reply with the mobility signature of S, a metric: $[t_0, V(t_0), D(t_0), P_V(t), P_D(t)]$ (msg 2 in figure 3)
(7)   C uses the mobility signature to determine with high probability the current position of $S$ and sends messages in this direction (msg 3 in figure 3)
(8)   S upon receiving the messages by C acknowledges it (msg 4 in figure 3) and initiates interaction with C
(9)   return
```

4.4. Mobility Management

This section details the mobility management aspect of this service architecture. The *mobility signature* is used by a mobile node to leave behind information about its mobility pattern; using this information, other nodes in the network can try and *predict* the current location of the mobile node.

Consider the situation when a node A becomes mobile and leaves its virtual home. Node A must leave some information behind using which, other nodes in the network can try to locate A. Earliest methods to handle mobility were using mobile IP which used the concept of a *home-address* and a *care-of* address. The problem with this scheme was that traffic intended for node A is held at the *home-agent* of A, until node A sends its home-agent its care-of address. This leads to a lot of overhead in the network (and also increases the latency). Our traffic-forwarding algorithm does not require a node to keep updating its position to its virtual home. Node A has some knowledge about its intended destination and hence its direction and speed of travel. Node A leaves behind this information in the form of a *mobility signature* (as mentioned in algorithm 1) with select *proxy* nodes that act as the *MIB*. With a reasonable probability, nodes which wish to contact node A can predict the new location of node A based on its *mobility signature* and the `elapsed` time since this information was provided.

The nodes in the MIB form a *quorum* and are responsible for holding the *mobile signature* of A. In the event that there are no nodes (or too few nodes) in its virtual home, A queries its neighboring virtual homes in an effort to recruit more nodes to be a part of the *MIB*. Nodes that are a part of the *MIB* may also become mobile, in which case they follow the same procedure as A to leave behind their *mobility signatures*. Algorithm 2 details the steps used by a mobile node to recruit *proxy* nodes to form the quorum-based *MIB*.

Algorithm 2 Forming a quorum-based MIB

```
Input: Void
Output: Result
FORM-QUORUM($k$)
(1)   Broadcast request
(2)   While (((n = $recv - reply()$) $\leq$ $k$) & ! timeout)
(3)       Expand broadcast radius
(4)       Broadcast request
(5)   $Select - Best(k, n)$
(6)   return
```

The quorum-based *MIB* is selected by the $Select - Best(k, n)$ function which selects k nodes out of the n nodes that replied to the request. This selection is made based on the mobility of the nodes, the security and the power-level at each node[3].

Two important cases arise during quorum selection:

- Case 1: The node cannot find any node in its *virtual home* and its neighbors to hold its *mobility signature*. In this case, the mobile node reaches its *new* virtual home and tries to periodically update this information with the MIB in its *old* virtual home, until a timeout.

- Case 2: Number of nodes available is less than k. In this case, the *mobility signature* is stored with the available nodes that replied to the broadcasted request, even if the number is less than k. This is to ensure that the *mobility signature* is available, in case a node wants to contact the mobile node. If more nodes join that virtual home over time, these nodes are recruited by the existing *MIB* to hold the *mobility signature*.

Consider the scenario when, node A deviates from the mobility pattern given by its *mobility signature*. Node A now sends a *correction* for this *mobility signature* back to

[3]The criteria for quorum selection will be studied as part of future research.

its virtual home. Any node wishing to reach A now uses this *corrected* mobility signature to locate A.

4.5. Piece-meal, location-driven traffic forwarding Algorithm (PMLD)

This section talks in detail about the traffic forwarding algorithm that takes advantage of the *mobility signature* left behind by a node when it leaves its virtual home. This *mobility signature* consists of the metric $[t_0, V(t_0), D(t_0), P_V(t), P_D(t)]$ (as shown in algorithm 1) and is stored with the *MIB* in the virtual home of the node.

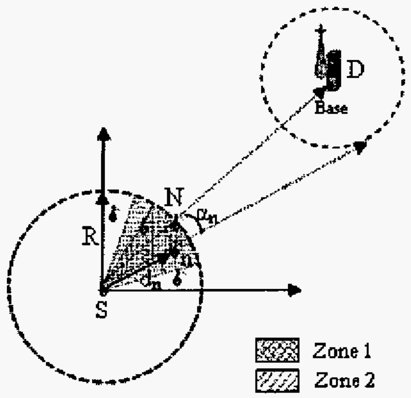

Figure 4. Directional Routing

Consider the scenario when a source S attempts to route traffic to a destination D and D is not present in its virtual home. The *mobility signature* of D is returned to S. Using this information S locates D and routes traffic to it. To limit flooding in the network, traffic is sent in a cone-shaped fashion towards D as shown in figure 4, (similar to [16], but here all the nodes need not know the position of every other node in the network). Nodes in zone 1 have the highest priority to forward the traffic, while the nodes in zone 2 have a lower priority. If no nodes are currently available in zone 1, the transmission area is expanded to include zone 2, after a timeout. This strategy imposes a priority on neighboring nodes in such a way that nodes which are more in line with the direction of the destination have higher priority to forward the message, thereby reducing the delay traffic suffers on its way towards the destination.

Consider the scenario when S sends a message to D using this approach. The nodes that receive the message sent by S calculate their priorities and based on this information, they either listen or forward the message. Upon hearing a message, an eligible node uses the above priority to decide if it should forward the message. Furthermore, upon hearing a transmission within the zone, the remaining eligible nodes drop the message. As the message progresses toward its destination, the highest priority node responsible for forwarding the message calculates a new cone and reiterates the process. This forwarding of messages happens on a *piece-meal* basis and hence the name for the algorithm. This algorithm is shown as a *pseudo code* in algorithm 3.

Algorithm 3 Forwarding Messages

```
Input: Void
Output: Result
FORWARD-MSG($\alpha_n$, $d_n$, $M(D)$)
(1)    Calculate $P_n$ using $\alpha_n$, $d_n$
(2)    While (!success)
(3)        Generate a random number in $[0, 1]$
(4)        With $P_n$
(5)            Calculate $V_L(D) = [V(t_0), D(t_0), P_V(T), P_D(T)]$,
               $V_L(D)$ gives the expected location of node $D$
(6)            Send limited-directed broadcast of $[V_L(D), M(D), L(N)]$
               (shown in figure 6), where $M(D)$ = message and $L(N)$ = N's location
(7)            success = $true$
(8)        With $1 - P_n$
(9)            Wait for the next time slot
(10)           if Msg-Sent by another node before timeout
(11)               drop request
(12)               return Success
(13)           else
(14)               continue
(15)   return Success
```

One important aspect of algorithm 3 is to calculate the priorities based upon which the nodes decide whether to forward the message or not. The priority function, P_n is the probability of forwarding (for each node n) and is dependent on α_n (the angle this node makes with the source) and d_n (the distance of this node from the source). Let the angle of the cone be α_c. Based on figure 4, it is clear that, if all nodes had equal energy reserves, node N is the best node in zone 1 to forward the message towards the destination and hence must have the highest priority. Our formula for calculating the priority must reflect this. We must choose a node within the cone that is the farthest away from the source and is also in the direction of the destination. To balance these two factors, weights are added to each factor in the formula[4]. Let R be the distance of N from S. The formula for calculating the priority is given by:

$$P_n = w_1 * \frac{d_n}{R} + w_2 * \frac{(\alpha_c - \alpha_n)}{\alpha_c} \quad (1)$$

$$P_n = 0, if d_n > R \text{ or } \alpha_n > \alpha_c \quad (2)$$

It can be clearly seen that $P_n \leq 1 \; iff \; w_1 + w_2 \leq 1$. The value of P_n is highest for node N, since α_n is 0 and $d_n = R$.

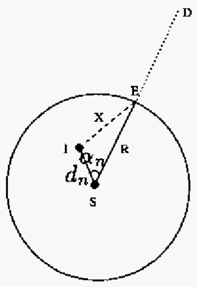

Figure 5. Calculation of α_n

To calculate P_n, an intermediary node needs to to calculate α_n. An intermediary node, I only needs to know its relative position with respect to the source and hence, does not require the use of GPS [2]. Consider figure 5; for node I to calculate the value of α_n, it needs to know d_n, R and x. Using the GPS-Free local co-ordinate system in [17] and the anchor node in its virtual home, node S calculates the co-ordinates of E (E is the point on the edge of the broadcast radius of S in the direction of D). This information can be embedded in the message to be forwarded. Node I can now calculate its position in the local co-ordinate system of S. Using this information, I calculates d_n, R and x. Now, using the cosine formula, we calculate α_n:

$$\alpha_n = acos\left(\frac{d_n^2 + R^2 - x^2}{2 * R * d_n}\right) \quad (3)$$

Another important aspect of the algorithm is the *directional routing* that is responsible for routing messages from a source to a destination. Each node along the path re-calculates the *cone* used to forward the message to the destination as shown in figure 6 (part A). Consider the timeline shown in figure 6 (part A). At time T_0, node D is at position D_0, at time T, the node is at position D and at time $T + \Delta$, node D is at position D_1. Now, consider the situation, when source S wants to send a message to D. It calculates with a certain probability (the method to calculate the probability is beyond the scope of this paper), a region where D can reside based on its *mobility signature*. Node S now calculates the angle α and hence derives the *cone* and sends the message towards the destination. Node $S1$ upon receipt of this

[4]The weights can be determined using simulation or analytical analysis.

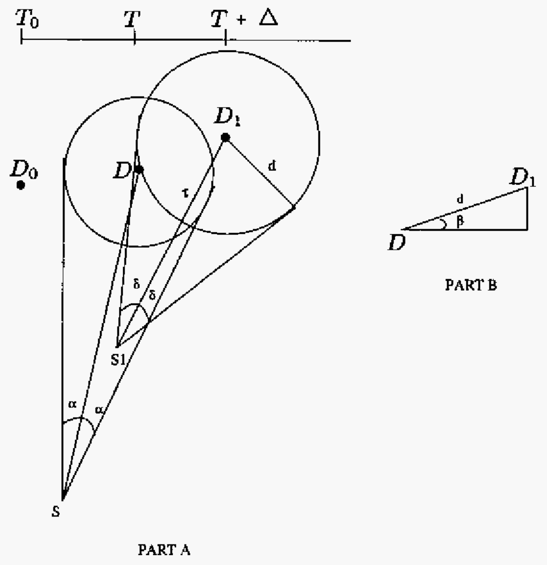

Figure 6. Directional Routing as destination moves

message, re-calculates the angle δ and hence re-calculates a new cone (based on new position of D) and forwards the message. The most important aspect in this protocol is the calculation of the cone, which is based on two things: the *expected speed* and the *expected direction* of D.

Consider part B of figure 6, this shows the movement of the destination from position D to D_1. Assume the expected direction of travel to be β with respect to the $x-axis$ and the expected speed of travel to be v. Let D be the point (x_1, y_1) and D_1 be the point (x_2, y_2) in the co-ordinate system. We know the position D and we need to find out the new location D_1 in terms of D. This is accomplished using the following equations (derived using simple laws of motion and trigonometry):

$$d = v * (T - (T + \Delta)) \Longrightarrow d = v * \Delta \quad (4)$$

$$x_2 = x_1 + d * cos\beta \quad (5)$$

$$y_2 = y_1 + d * sin\beta \quad (6)$$

Consider the scenario when node $S1$ receives a message from S that is intended for the destination, now $S1$ needs to calculate the angle of the cone. Now that we have the position D_1 and the value for d, we need to calculate δ based upon these values. Let $S1$ be the point (sx_1, sy_1). The following sets of equations help us derive δ:

Using the values of x_2 and y_2 from equations 5 and 6 respectively, we get:

$$\tau = \sqrt{(x_2 - sx_1)^2 + (y_2 - sy_1)^2} \quad (7)$$

Using the value of τ from equation 7 and the value of d from equation 4, we get:

$$\delta = arcsin(\frac{d}{r}) \tag{8}$$

This algorithm is used by all the nodes to build the cone to forward traffic towards the destination. The advantage with this scheme is that it adapts to the mobility of the node, the lesser a node moves, the smaller the cone, and the greater a node moves, the bigger the cone.

5. Simulation and Results

This section explains in detail the simulation environment used and the ensuing results.

5.1. Experimental Testbed

The protocol was implemented in the Glomosim network simulator [11] on Linux and was tested by providing different network scenarios. The Glomosim network simulator was developed at UCLA and is primarily used to simulate wireless networks. The first set of tests were conducted as part of the sensitivity analysis of the protocol. The second test compared the performance of our protocol to LAR [10]. We compared the performance of our protocol to LAR because LAR is also an on-demand location-based protocol and has been used as a benchmark for comparison for position-based protocols. The performance metric chosen for comparison was the throughput. Using the throughput of the protocols as a metric gives us a good indication about the performance of the protocol with respect to the delay and number of packets dropped in the network.

A basic sensitivity analysis of the protocol provides us with some ideas about the performance of the protocol under different network conditions. For this reason, we simulated three different types of networks differentiated by the channel characteristics and the mobility of the nodes. The channel characteristics used were those that were available in Glomosim: *TWO-RAY* (where the receiving antenna sees two signals, a direct path signal and a signal reflected off the ground) and *FREE-SPACE* (where radio wave propagation is in the absence of any reflections or multipath). The mobility models available in Glomosim that were used during the experiments were: *NO-MOBILITY*, where nodes are static and *RANDOM-WAYPOINT* mobility, where a node randomly chooses a destination and moves towards that destination. The speed of the node is chosen randomly between an upper limit (10 m/s) and a lower limit (0 m/s). Upon reaching the destination, the node pauses for a pause time (30s), before becoming mobile again. For the comparative analysis with LAR, we compare the throughput for a network of mobile nodes.

In all experiments, the throughput was measured with respect to the density of the network (number of nodes in the network). The number of nodes in the network was varied from 100 to 500 nodes and these nodes were randomly placed in a network grid of size 3000x3000m. The network simulated was thus varied from a sparsely populated network to a densely populated network. Traffic generated was CBR traffic using node 1 as the source and node 2 as the destination; node 1 sends packets, each of size 512 bytes to node 2 starting at time 0, with a packet sent every 10s. Traffic statistics are collected at the destination by measuring the total time taken (in $nano-seconds$) for the packets to reach the destination and also the total number of packets that reached the destination. The total time set for the simulation was 99s (this results in a total of 10 packets being sent by the source). Each data point represented in the graphs was the value averaged over 10 independent experimental runs. Table 1 shows a summary of the different design parameters used during the simulation and Table 2 shows the parameters used for the *RANDOM-WAYPOINT* mobility model.

Table 1. Summary of simulation parameters

Name	Value
No. of nodes	100 - 500
Node Distribution	Random
Channel Characteristic	TWO-RAY, FREE-SPACE
Grid Size	3000x3000m
Type of traffic	CBR
Size of packet	512 bytes
Number of packets	10
Mobility Patterns	NO-MOBILITY, RANDOM-WAYPOINT
Start of simulation	0s
Length of simulation	99s

Table 2. Parameters for the *RANDOM-WAYPOINT* mobility model

Name	Value
Node Speed (upper limit)	10m/s
Node Speed (lower limit)	0m/s
Node Pause Time	30s

5.2. Results

5.2.1. Sensitivity Analysis

In this section, we perform a series of experiments to measure the performance of our protocol as part of the sensitivity analysis. The experiments were conducted to measure the throughput of the protocol while varying the density of the network, the channel characteristics and the mobility of the nodes in the network.

The first set of experiments, results of which are depicted in figures 7 and 8 respectively, were performed by varying the channel characteristics. The reason for using different channel characteristics was to observe the effect that various transmission ranges have on the routing protocol. For the purpose of this experiment, the nodes in the network are assumed to be static. The second experiment, result of which is depicted in figure 9 was performed to measure the performance of the protocol for a network consisting of mobile nodes. For this experiment, the channel characteristic was not changed.

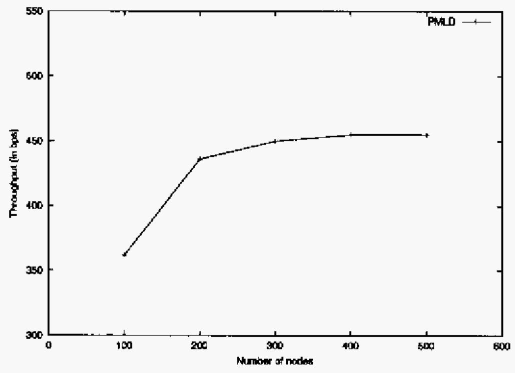

Figure 7. Channel Characteristic: Two-Ray (Using PMLD)

From figure 7, we conclude that the throughput increases as the density of the network grows. This can be attributed to the fact that as the network size increases, there is a higher probability of a node being available in the path towards the destination and hence this node can forward the packet towards the destination.

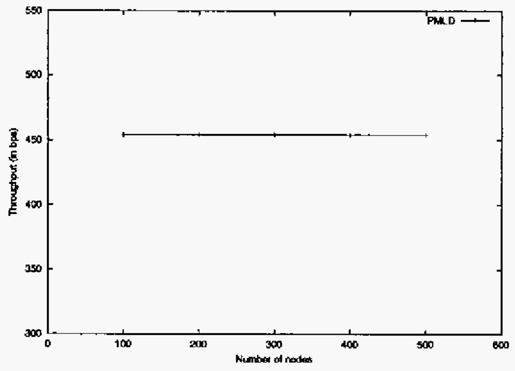

Figure 8. Channel Characteristic: Free-Space (Using PMLD)

From figure 8, we find out that the throughput is very high even for a network that is not very dense. The transmission range for the node using the channel, *FREE-SPACE* is usually much higher when compared to using the channel *TWO-RAY*. This increase in transmission radius in the nodes in the network leads to a lower hop-count for a packet that is transmitted from the source to the destination, thus leading to a substantially higher throughput.

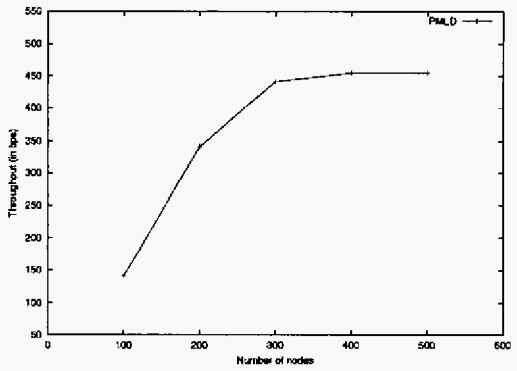

Figure 9. Mobility: Random-WAYPOINT (Using PMLD)

Figure 9, shows the case when mobility is introduced into the network. The channel characteristic used in this experiment was *TWO-RAY*. The reason for using the *TWO-RAY* channel characteristic was to make sure that the nodes moved out of range of each other. In the case of *FREE-SPACE*, the increased transmission range meant that sometimes nodes were still reachable even when they moved away from each other. The nodes follow the *RANDOM-WAYPOINT* mobility pattern and there is no assumption made as to which nodes are static and which nodes are mobile in the network. We observe that the throughput is slightly lower in this case when compared to the static case. This is to be expected due to the mobility in the network.

5.2.2. Comparative Analysis

In this section, we do a comparative study by comparing our protocol to LAR in terms of the throughput achieved for a network consisting of mobile nodes. The channel characteristic used for this experiment was *TWO-RAY* and the mobility pattern used was the *RANDOM-WAYPOINT* mobility provided in Glomosim. The statistics collected for LAR were available as part of the LAR implementation that is provided with Glomosim.

Figure 10 shows us that as we introduce mobility into the network, PMLD out performs LAR. There is no assumption made about the location of the destination once the simulation starts (the source knows the location at the beginning of the simulation). As nodes become mobile in the network,

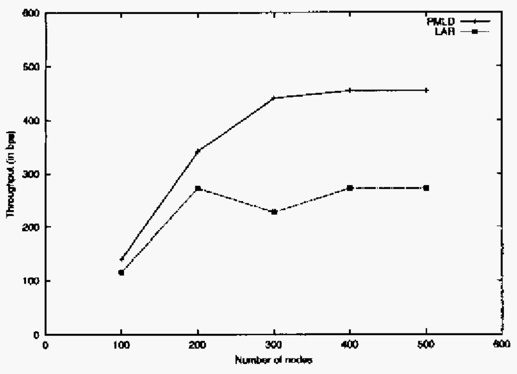

Figure 10. Mobility: Random-WAYPOINT (Using PMLD and LAR)

routes that were discovered by LAR at the beginning of the simulation may not be valid later and hence another route discovery must be performed. This overhead increases the latency to send packets from the source to the destination. PMLD is primarily a forwarding protocol and hence it does not incur the cost associated with forming and repairing routes. At 500 nodes (highly dense network), the value for PMLD achieves the maximum throughput (the same value achieved by PMLD in the static case with channel characteristics *TWO-RAY* and *FREE-SPACE*). The denser the network becomes, the better PMLD performs due to the availability of more nodes in the network that can forward the packet towards the destination.

6. Conclusion and future work

This paper makes two significant contributions in providing pervasive services in ad-hoc networks.

- The paper proposes a service-architecture that does not assume a fixed infrastructure and imposes no location restrictions on the servers and services in the network.
- The paper proposes and evaluates a new data dissemination and propagation algorithm that forwards traffic in a piece-meal, location-directed manner. A sensitivity analysis was performed on the protocol, after which it was compared to LAR. For a network of mobile nodes, it was found that the proposed protocol out performs LAR.

There is a lot of potential for future work in this area. The forwarding algorithm could be made power-aware by allowing the probability of forwarding at each node to grow according to the power at the node (high power nodes increase their probability faster than low power nodes, thus ensuring that nodes with higher power have a higher probability of forwarding).

References

[1] A. Gopalan, S. Dwivedi, T. Znati and A. B. McDonald. On the implementation of the (α, t)-Cluster Protocol on Linux. In *Proc. 37th Annual Simulation Symposium*, Apr. 2004.

[2] E. E. D. Kaplan. *Understanding GPS: principles and applications*. Artech House, Boston, MA, 1996.

[3] E. Guttmann and C. Perkins and J. Veizades and M. Day. Service Location Protocol, 1999.

[4] IP for Wireless/Mobile Hosts (mobileip) Charter. `http://www.ietf.org/html.charters/mobileip-charter.html`.

[5] Z. J. Haas and M. Pearlman. The Zone Routing Protocol (ZRP) for Ad Hoc Networks. *Internet Draft*, Aug. 1998.

[6] J. P. Hubaux, J. Y. Le Boudec, G. T., and V. M. Towards self-organizing mobile ad-hoc networks: the terminodes project. *IEEE Comm Mag*, 39(1):118 –124, Jan 2001.

[7] I. Stojmenovic. Home agent based location update and destination search schemes in ad hoc wireless networks. Technical Report TR-99-10, Computer Science, SITE, University of Ottawa, Sep. 1999.

[8] J. Li and J. Jannotti and D. De Couto and D. Karger and R. Morris. A Scalable Location Service for Geographic Ad-Hoc Routing. In *Proceedings of the 6th ACM International Conference on Mobile Computing and Networking (MobiCom '00)*, pages 120–130, Aug. 2000.

[9] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad hoc Wireless Networks. In *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[10] Y. B. Ko and N. H. Vaidya. Location-Aided Routing (LAR) in Mobile Ad-Hoc Networks. In *Proc. ACM/IEEE MOBICOM*, Oct. 1998.

[11] L. Bajaj, M. Takai, R. Ahuja, R. Bagrodia and M. Gerla. Glomosim: A scalable network simulation environment. Technical Report 990027, UCLA, May, 1999.

[12] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nataraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-Peer computing. Technical Report HPL-2002-57, HP Laboratories Palo Alto, Mar. 2002.

[13] C. Perkins and E. Royer. Ad Hoc On-Demand Distance Vector Routing. In *Proceedings 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99)*, 1999.

[14] C. R. Perkins and P. Bhagwat. Highly Dynamic Destination Sequenced Distance Vector Routing (DSDV) for Mobile Computers. In *ACM SIGCOMM*, Oct. 1994.

[15] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proceedings of ACM SIGCOMM*, 2001.

[16] S. Basagni, I. Chlamtac, V. R. Syrotiuk and B. A. Woodward. A Distance Routing Effect Algirithm for Mobility (DREAM). In *Proc. ACM/IEEE Mobicom*, Oct 1998.

[17] S. Capkun, M. Hamdi and J. P. Hubaux. GPS-Free Positioning in Mobile ad-hoc Networks. In *HICSS*, 2001.

[18] Seung-Chul M. Woo and Suresh Singh. Scalable Routing Protocol for Ad Hoc Networks. *Journal of Wireless Networks*, 7(5), Sep. 2001.

[19] Yaron Y. Goland, Ting Cai, Paul Leach, Ye Gu. Simple Service Discovery Protocol/1.0. *Internet Draft*, Oct. 1999.