An Adaptive Framework for QoS Routing through Multiple Paths in Ad hoc Wireless Networks

S. K. Das [*1], A. Mukherjee [+2], S. Bandyopadhyay [^], D. Saha [^3], K. Paul [#]
das@cse.uta.edu, amitava.mukherjee@in.pwcglobal.com, {somprakash, ds}@iimcal.ac.in, krishna.paul@ccrle.nec.de

[*] Center for Research in Wireless Mobility and Networking (CreWMaN),
Department of Computer Science and Engineering,
The University of Texas at Arlington, Arlington, TX 76019-0015, USA

[+] PricewaterhouseCoopers Ltd., Sector V, Saltlake, Calcutta 700 091 India

[^] MIS Group, Indian Institute of Management Calcutta, Calcutta 700 104, India

[#] C&C Research Laboratories**,** NEC Europe Ltd.
69115 Heidelberg, Germany

***Abstract*:** We propose a mechanism for adaptive computation of multiple paths in temporal and spatial domain to transmit large volume of data packets from a source s to a destination d in ad hoc wireless networks. The objective of this adaptive framework is to achieve quality of service (QoS) by minimizing end-to-end (or delivery) delay. We consider two aspects in this framework. The first aspect is to perform preemptive route re-discoveries before the occurrence of route errors while transmitting large volume of data from s to d. Consequently, this helps to find out dynamically a series of multiple paths in temporal domain to complete the data transfer. The second aspect is to select multiple paths in spatial domain for data transfer at any instant of time and to distribute the data packets in sequential blocks over those paths in order to reduce congestion and end-to-end delay. A notion of link stability and path stability has been defined and a unified mechanism is proposed to address these two aspects that relies on evaluating a path based on link stability and path stability. Our simulation method uses Lagrangean relaxation and subgradient heuristics to solve an optimization problem in order to find out the paths and data distribution into those paths both in temporal and spatial domains. The performance of this approach has been evaluated on a simulation environment. It has been observed that the use of temporal multi-paths allows any source to transmit a large volume of data to a destination without degradation of performance due to route-errors. Additionally, the use of spatial multi-paths would help to significantly reduce the end-to-end delay and the number of route-rediscovery needed in this process.

[1] The first author is partially supported by a grant from Nortel Networks
[2] Corresponding author
[3] The fourth author is partially supported by grants from AICTE & UGC, Govt. of India

1. Introduction

There has been a growing interest in ad hoc networks in recent years [1]-[5]. An ad hoc network can be envisioned as a collection of mobile routers (each equipped with a wireless transceiver), which are free to move about arbitrarily. In an ad-hoc network, two nodes, willing to communicate, may be outside the transmission range of each other; they still are able to communicate in multiple hops, if other nodes in the network are willing to forward packets from them.

An important problem associated with routing in networks, particularly in ad hoc networks, is to employ methods that will ensure better quality of service (QoS). The successful operation of an ad-hoc network is disturbed, if an intermediate node, supporting communication between an (s-d) pair, moves out of range of either source and destination during data transfer. This interruption in communication results in degraded QoS because the user has to wait for a subsequent re-discovery of another route between this (s-d) pair. Otherwise some path-maintenance algorithm has to be invoked to prevent the disruption of communication that eventually increases the end-to-end delay.

The majority of earlier schemes proposed in the context of ad hoc networks, use single-path routing [1], [3]-[5], which might not reduce the average end-to-end delay. However, once a set of paths between (s-d) is discovered in anticipation, in some cases, it is useful to employ the concept of multipath communication as discussed in [10]. In fact, some research studies [6], [11], [17] have shown it is possible to improve end-to-end service by taking recourse to simultaneous data transfer over these paths. This parallelism in transmission is achieved by splitting the original data volume into smaller blocks and sending these blocks of data via selected multiple paths from s to d, which in turn reduces congestion and end-to-end delay. However, most of these works considers only spatial multipath, whereas our work takes both spatial and temporal multipaths into account. A preliminary version of this work has been presented in [16]. Utilization of multiple paths in order to provide improved performance against single path has been explored in wire networks [9]-[11]. However, it has also been shown that deployment of multiple paths does not necessarily result in a lower end-to-end delay [6].

In this paper, we propose a framework for adaptive computation of multiple paths both in temporal and spatial domains to transmit a large volume of data between (s-d) in ad hoc wireless networks. Data transmission in spatial domain balances traffic load in the network, while that in temporal domain gives the continuity of data transfer from s to d. We have considered two different aspects in this framework. The first aspect performs preemptive route re-discoveries before the occurrence of route errors while transmitting a large volume of data from s to d. Consequently, this has helped to find out dynamically a series of paths in the temporal domain to complete the data transfer. The second aspect selects multiple paths in the spatial domain for data transfer at any instant of time and to distribute the data packets in sequential blocks over those paths in order to reduce further congestion and end-to-end delay. We also define a notion of link stability and path stability, based on which are computed paths in the proposed framework. The simulation uses Lagrangean relaxation and subgradient heuristics [7]-[8] to find out the paths and data distribution into those paths both in temporal and spatial domains.

The organization of this paper is as follows. Section 2 describes a stability-based framework for QoS routing in ad hoc wireless networks. Section 3 enhances this framework to introduce an adaptive mechanism for multipath routing using a solution method based on Lagrangean relaxation and subgradient heuristics. Section 4 explains the simulation results followed by concluding remarks in Section 5.

2. A Stability-based Framework for QoS Routing

Let us assume that nodes (s-d) need to communicate and the routing scheme has detected a path (s-x-d), where x is an intermediate node. However, if x is highly mobile and tends to move outside the transmission range of s and/or d, the routing scheme soon has to find an alternative path (s-y-z-d). This interruption in service eventually degrades the quality of service [5]. Instead, if the routing scheme has considered the mobility pattern of the intermediate nodes and also considered the traffic congestion, it might be able to find a better path beforehand, that has a longer life and is also less congested in a specific context.

Before proceeding further, let us introduce the following notations used to describe the framework:

2.1 Notations

N: set of nodes
L: set of directed links
s, d: source and destination, s, d $\in$ **N**
l_{mn}: link from node m to node n, where m, n$\in$ **N** and $l_{mn} \in$ **L**
P: super set of a set of stable paths, i.e., P={$\boldsymbol{P_1}$, $\boldsymbol{P_2}$, $\boldsymbol{P_3}$,…} = {$\boldsymbol{P_i}$}, i$\in[1,\propto)$
$\boldsymbol{P_i}$: ith set of stable paths, $\boldsymbol{P_i} \in \mathbf{P}$, i $\in[1,\propto)$
p_j: binary indication variable for jth path in the set $\boldsymbol{P_i}$, where each set $\boldsymbol{P_i} \in \mathbf{P}$ and i $\in[1,\propto)$, j$\in[1, |\boldsymbol{P_i}|]$ and p_j = 1, where path p_j is selected for s,d for the set $\boldsymbol{P_i}$
= 0, otherwise
D: super set of a set of data volume in packets i.e., **D** = {$\boldsymbol{D_1}$, $\boldsymbol{D_2}$, $\boldsymbol{D_3}$,…} = {$\boldsymbol{D_i}$}, i$\in[1,\propto)$
$\boldsymbol{D_i}$: ith set of data volume in packets, $\boldsymbol{D_i} \in \mathbf{D}$, i $\in[1,\propto)$
Δ_j: data distribution for the jth path in the set $\boldsymbol{P_i}$ and $\sum_{j\in \boldsymbol{P_i}} \Delta_j = |\boldsymbol{D_i}|$ for i$\in[1,\propto)$
R: transmission range of nodes (assumed to be equal for all nodes)
M: average velocity of nodes
a_{mn}: affinity of link $l_{mn} \in$ **L**
B: bandwidth of link in packets/msec (assumed to be identical for all links)
H_j: number of hops traversed by jth path in the set $\boldsymbol{P_i}$
τ_j: average delay per hop per packet for jth path in the set $\boldsymbol{P_i}$
d_j: route discovery time for jth path in the set $\boldsymbol{P_i}$
q_j: average queuing delay per packet per node for jth path in the set $\boldsymbol{P_i}$
T_j: average path delay per packet for jth path in the set $\boldsymbol{P_i}$ and $T_j = H_j * \tau_j$

S_j: stability for jth path in the set $\boldsymbol{P_i}$ and $j \in \boldsymbol{P_i}$, $i \in [1,\infty)$
$d_{stab(j)}$: route discovery time for the most stable jth path in the set $\boldsymbol{P_i}$
$H_{stab(j)}$: number of hops traversed by the most stable jth path in the set $\boldsymbol{P_i}$
$\tau_{stab(j)}$: average delay per hop per packet for the most stable jth path in the set $\boldsymbol{P_i}$
$T^{\boldsymbol{disc}}$: route-request-time-out for any set $\boldsymbol{P_i}$
H^{ave}: average number of hops traversed by any jth path in any set $\boldsymbol{P_i}$

2.2 System Description

The network is modeled as a graph G = (**N**, **L**) where **N** is a finite set of nodes and **L** is a finite set of directed links. Each node $n \in \mathbf{N}$ is having a unique node identifier. Two nodes n and m are connected by two unidirectional links $l_{nm} \in \mathbf{L}$ and $l_{mn} \in \mathbf{L}$ such that n can send message to m via l_{nm} and m can send message to n via l_{mn}. However, in this study, we have assumed $l_{nm} = l_{mn}$ for the sake of simplicity.

In a wireless environment, each node n is characterized by a transmission range. We have defined the neighbors of n as the set of nodes within the transmission range R of n. It is assumed that when node n transmits a packet, it is broadcast to all of its neighbors. However, in the wireless environment, the strength of connections to all members of the neighbor set with respect to any node n are not uniform. For example, a node m in the periphery of the transmission range of n is weakly connected to n compared to a node u, which is closer to n. Thus, the chance of m going out of the transmission range of n due to an outward mobility of either m or n is more than that of u.

The strength of relationship between two nodes over a period of time is defined as *node-affinity* or *affinity*. Informally speaking, *link-affinity* $a_{nm}(t)$, associated with a link l_{nm} at time t, is a prediction about the span of life of the link l_{nm} in a particular context. Link-affinity $a_{nm}(t)$ at that instant of time is a function of the current distance between n and m, relative mobility of m with respect to n, and the transmission range of n. If transmission range of n and m are different, $a_{nm}(t)$ may not be equal to $a_{mn}(t)$. The *node-affinity* or *affinity* $\eta_{nm}(t)$ between two nodes n and its neighbor m is defined as $\min[a_{nm}(t), a_{mn}(t)]$. The stability of connectivity between n and its neighbor m depends on η_{nm}. The unit of affinity is seconds.

To find out the link-affinity $a_{nm}(t)$ at any instant of time, node n sends a periodic beacon and node m samples the strength of signals received from node n periodically. Since the signal strength of n as perceived by m is a function $f(R_n, d_{nm})$ where R_n is the transmission range of n, and d_{nm} is the current distance between n and m at time t, the node m can predict the current distance d_{nm} at time t between n and m. If M is the average velocity of the nodes, the worst-case link-affinity $a_{nm}(t)$ at time t is $(R_n\text{-}d_{nm})/M$, assuming that at time t, the node m has started moving outwards with an average velocity M. For example, If the transmission range of n is 300 meters, the average velocity is 10m/sec and current distance between n and m is 100 meters, the life-span of link l_{nm} (worst-case) is 20 seconds, assuming that the node m is moving away from n in a direction obtained by joining n and m.

The above method is simple, but is based on an optimistic assumption that node-distance can be deduced from signal strength. In real life, even if the transceivers have the same transmission range, it will vary because of the differences in battery power in each of them. Therefore, it may be difficult for one node to estimate distance from another node by monitoring the current signal strength only. So, a node m needs to monitor the change in signal strength of n over time to estimate link-affinity, as described below [18]:

Let $\Delta S_{nm}(t)$ be the change of signal strength at time t and is defined as:
$\Delta S_{nm}(t) = S_{nm}(t) - S_{nm}(t- \Delta t)$, where $S_{nm}(t)$ is the current sample value of the signal strength of node n as perceived by node m at time t, $S_{nm}(t- \Delta t)$ is the previous sample value at time $(t- \Delta t)$ and Δt is the sampling interval. Let $S'_{nm}(t)$ be the rate of change of signal strength at time t and is defined as $S'_{nm}(t) = (\Delta S_{nm}(t) / \Delta t)$ and let $S'_{nm}(t)_{avg}$ is the average rate of change of signal strength at time t over the past few samples. Let S_t be the threshold-signal-strength : when the signal strength S_{nm} associated with l_{nm} goes below S_t, we assume that the link l_{nm} is disconnected. Further we define
$a_{nm}(t)$ = *high*, if $S'_{nm}(t)_{avg}$ is positive;
= $(S_t - S_{nm}(t)) / S'_{nm}(t)_{avg,}$ if $S'_{nm}(t)_{avg}$ is negative.

If $\Delta S_{nm(ave)}$ is positive, it indicates that the link-affinity is increasing and the two nodes are coming closer. Hence, link-affinity is termed as *high* at that instant of time. The value for *high* is computed as (transmission range / average node velocity) and is approximately equal to the time taken by a node m to cross the average transmission range of node n with an average velocity. However, as indicated earlier, even if $S'_{nm}(t)_{avg}$ is positive, a node $m \in N_n$ in the periphery of the transmission range of n is weakly connected to n compared to a node $p \in N_n$ which is closer to n. Thus, the chance of m going out of the transmission range of n due to a sudden outward mobility of either m or n is more than that of p. Thus, if $S'_{nm}(t)_{avg}$ is positive, a correction factor μ is used to moderate this *high* value. This correction factor μ is equal to $(1 - S_t / S_{nm}(t))$ and $a_{nm}(t) = \mu$* *high*, if $S'_{nm}(t)_{avg}$ is positive. This indicates that if $S_{nm}(t)$ is very close to S_t, μ will be close to zero and consequently $a_{nm}(t)$ will also become close to zero, even if $S'_{nm}(t)_{avg}$ is positive. As indicated earlier, the *node-affinity* or *affinity* between two nodes n and m, $\eta_{nm}(t)$, is defined as $\min[a_{nm}(t), a_{mn}(t)]$.

Given any path p = (i, j, k, …, l, m), the stability of path p will be determined by the lowest-affinity link (since that is the bottleneck for the path) and is defined as : $\min[\eta_{ij}(t), \eta_{jk}(t), \ldots, \eta_{lm}(t)]$. In other words, stability of path p at some instant of time t between source s and destination d, $\eta^p_{sd}(t)$, is given by: $\eta^p_{sd}(t) = \min[\eta_{ij}(t)], \forall i,j \in p$.

However, the notion of stability of a path is dynamic and service-specific. It is dynamic because, as indicated earlier, the stability of a path is the span of life of that path from a given instant of time. Moreover, stability has to be seen in the context of providing service. A path between (s-d) is stable for an instant of time, if its span of life is sufficiently stable to complete service such as transferring the required volume of data between (s-d). This path is sufficiently stable to transfer a small volume of data between (s-d); but this is unstable for a large volume of data to be transferred.

Thus, even if a path to find to be, it has not been sufficiently stable to carry a large volume of data in a highly dynamic environment of ad hoc wireless networks. In this situation, communication cannot be initiated for transferring a large volume of data because of the low stability in the path. Even if the communication has been initiated, some form of route maintenance scheme has to be deployed to repair the path or to find out an alternative path in case a route error has occurred. However, this interruption in service and its resumption after route re-discovery has eventually degraded the average QoS. Instead, if it is possible to predict the life span of a path between (s-d), and, accordingly preempt the process of route re-discovery and discover a new path between (s-d), before the existing path breaks, it is possible to provide uninterrupted (better quality) service of data communication between (s-d).

Moreover, once a set of paths between s to d has been discovered, in some cases, it is possible to improve end-to-end delay by splitting the volume of data further into different blocks and to send it via selected multiple paths between (s-d), which has eventually reduced the congestion and end-to-end delay. Of course, depending on the topology, we need to decide dynamically whether multi-path is a preferred mode or not.

2.3 Estimating Communication Delay between a Source and a Destination

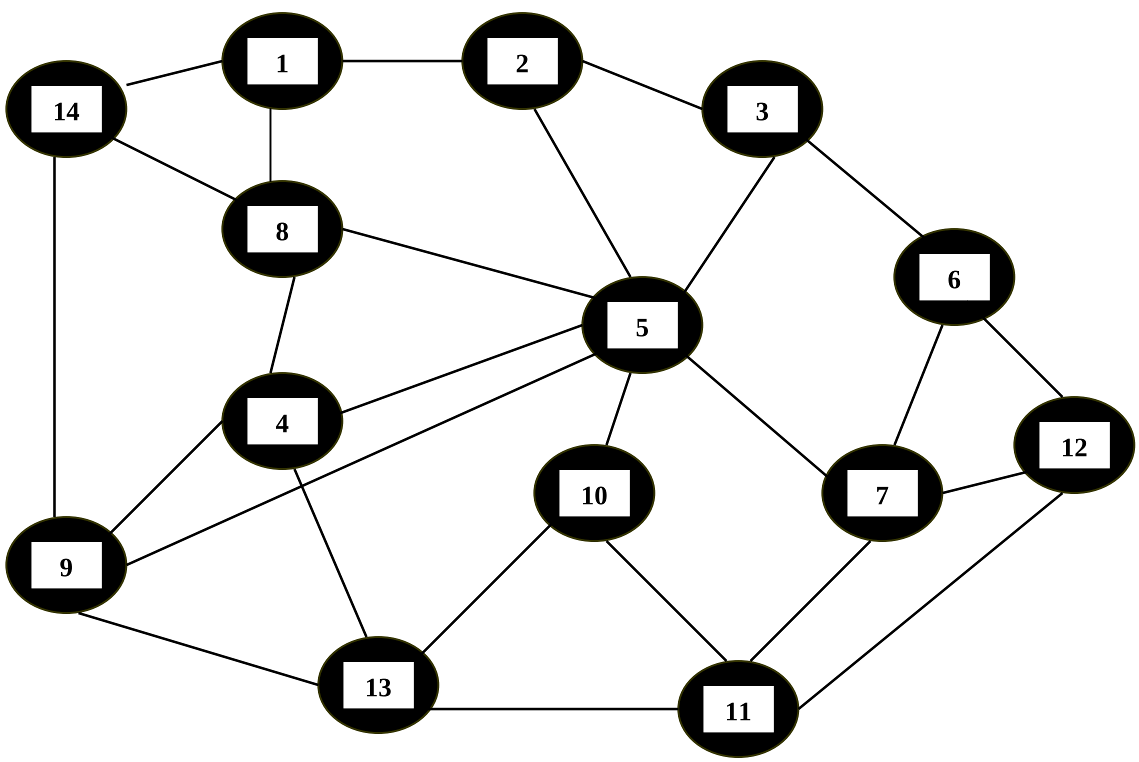

Figure 1: Data Communication from Source (14) to Destination (5)

To start with, let us assume a static network setting where nodes are not mobile. If q_j is the average queueing delay in msec per packet per node in the j^{th} path from s to d, B is the bandwidth in packets/msec and τ_j is the average delay in msec per hop per packet of a traffic stream for j^{th} path from s to d, then

$\tau_j = q_j + 1/B$, ignoring propagation processing delays.

The first component is the queuing delay and the second component is the delay due to packet transmission. We assume that the queuing delay is the most prominent factor in determining the overall QoS. If T_j is the total average delay per packet for the j^{th} path from s to d with H_j number of hops, then $T_j = H_j * \tau_j$.

Consider the situation in Figure 1 where source (14) is sending data to destination (5) which is 4 hops away from 14 (say, the path 14-1-2-3-5) using RTS / CTS / ACK protocol [15]. When 14 is transmitting a data packet, node 1 has to accept it, if 1 is in the mode of receiving packet; node 2 cannot transmit packet to avoid interference with node 1's reception. However, 3 can transmit simultaneously to node 5, if node 3 has any data packet to send. Thus, there is a possibility in establishing a transmission in pipelined fashion, if the number of hops is more than 3. Furthermore, there is also a chance that 14 will complete its entire data transfer activity to node a before a gets chance to transmit. So, the worst-case delay per packet is $H_j * \tau_j$.

Now, let us consider a dynamic scenario where nodes are mobile. It implies that an intermediate node in j^{th} path, participating in a communication between two nodes (s-d), may move out of range suddenly or may switch itself off in between transfer of a message. In other words, j^{th} path between s and d may not be stable enough to complete the desired communication and we may need to rediscover another path to complete the transfer of data. Let $|\mathbf{D}|$ be the number of packets to be sent between (s-d); let T_{sd} be the average path delay per packet between them d_{sd} the average route discovery time for discovering a set of paths between (s-d); k be the number of times route rediscovery is required for completing data transfer between (s-d) and let t_{sd} be the average time taken by the source to detect the occurrence of a route error. Then the total delay to complete $|\mathbf{D}|$ packets of data transfer between (s-d) is given by

$$\text{Total Delay} = |\mathbf{D}| * T_{sd} + k * d_{sd} + k * t_{sd}$$

In the proposed framework, we minimize the end-to-end delay by using the following conditions:

1. It performs preemptive route re-discoveries that overlap with data communication in time-domain. In other words, $|\mathbf{D}| * T_{sd}$ and $k * d_{sd}$ overlap in time-domain.
2. Since the framework avoids the occurrence of route-errors, $t_{sd} = 0$.
3. Because multiple paths are used in spatial domain, T_{sd} can be reduced significantly, as explained in the next section.

3. Framework for Multi-path Routing

This section describes an adaptive framework for multi-path routing using a solution method based on Lagrangean relaxation and subgradient heuristics. Subsection 3.1 illustrates the problem formulation while Subsection 3.2 explains the path finding algorithm. Subsection 3.3 depicts the multi-path algorithm for data communication and an example explains this algorithm in Subsection 3.4. Finally, Subsection 3.5 describes the solution method based on Lagrangean relaxation and subgradient heuristics.

3.1 Problem Formulation

Let us consider that the volume $|\mathbf{D}|$ of data packets needs to be routed from a source to destination pair (s-d). If $|\mathbf{D}|$ is very large, then it cannot always be possible to send data in a single path in the mobile environment. This is because of the fact that the stability of the path is not so high to keep the connectivity during the routing of the entire data volume. So, the total data volume $|\mathbf{D}|$ is divided into smaller size packets $|\boldsymbol{D_1}|$, $|\boldsymbol{D_2}|$… called as *temporal data sets*, which are obtained dynamically at different time intervals. The ith temporal data set $\boldsymbol{D_i}$, corresponds to the ith temporal stable path set $\boldsymbol{P_i}$, in which data packets are routed from s to d. Clearly, the corresponding temporal stable path sets {$\boldsymbol{P_1}$, $\boldsymbol{P_2}$,…} form the set **P**. That is, **P** and **D** are sets of temporal sets {$\boldsymbol{P_1}$, $\boldsymbol{P_2}$,…} and {$\boldsymbol{D_1}$, $\boldsymbol{D_2}$,…} respectively.

Each temporal stable path set $\boldsymbol{P_i}$, has a number of stable j-paths (j=1,2..). The jth stable path is selected when the value of p_j is 1 and the maximum number of paths selected in the set $\boldsymbol{P_i}$ is equal to $|\boldsymbol{P_i}|$. We call these j-paths (j=1,2..) as spatial paths in the temporal path set $\boldsymbol{P_i}$. On these j-spatial paths, the data $|\boldsymbol{D_i}|$ is distributed further in Λ_j (j=1,2…), called *spatial data volume* and the sum of all Λ_j's is equal to $|\boldsymbol{D_i}|$. The S_j, d_j and H_j for the path p_j are self-explained terms.

In order to perform a preemptive route rediscovery before the occurrence of route error, the source is able to initiate the route rediscovery in such a manner that the next set of paths are available before the completion of current data volume. Let us assume that T_{disc} in msec. is the route-request-time-out, i.e., the maximum time interval allocated from generating a route-request from a source and getting route replies back to source for any set $\boldsymbol{P_i}$. Let us also assume that B is the bandwidth in number of packets per msec. and H^{ave} be the average number of hops traversed by any jth path in any set $\boldsymbol{P_i}$. For any set $\boldsymbol{P_i}$ with data volume $|\boldsymbol{D_i}|$ to be communicated, the route-rediscovery for the next set $\boldsymbol{P}_{i+1}$ are initiated after transmitting ($|\boldsymbol{D}_i| - T_{disc} * B/H^{ave}$) amount of data packets which ensure that the next set $\boldsymbol{P}_{i+1}$ is available to source just before completion of transmitting $|\boldsymbol{D}_i|$ volume of data packets. For simplicity, we assume B= 1 packet per msec.

Based on the above notations and terminology, we describe the problem as follows.

Minimize the average delay of a wireless mobile ad-hoc network by sending data packets dividing them into a set of several temporal paths for each source-destination (s-d) pair. Again each of these temporal paths is divided into a number of spatial paths through *which smaller data blocks are sent. Several stabilized paths with route discovery time and a number of hops traversed into these paths have been taken as input to the problem*.

Formally this optimization problem is defined as:

Minimize: $Z = (\sum_{i\in[1,\infty)} \sum_{j\in \boldsymbol{P_i}} (d_j + \Delta_j H_j \tau_j p_j))^{\alpha}$ for each s-d pair (1)
where α is the degree of the complexity of the optimization problem.

Subject to:

$$\sum_{i\in[1,\infty)} \sum_{j\in \boldsymbol{P_i}} (d_j + \Delta_j H_j \tau_j p_j) < \sum_{i\in[1,\infty)} \sum_{j\in \boldsymbol{P_i}} (d_{stab(j)} + |\boldsymbol{D_i}| H_{stab(j)} \tau_{stab(j)} p_j) \quad (2)$$

$$\forall_{i\in[1,\infty)} [\forall_{j\in \boldsymbol{P_i}} (d_j + \Delta_j H_j \tau_j p_j) < S_j - \sum_{k=1}^{j-1} (d_k + \Delta_k H_k \tau_k p_k)] \quad (3)$$

$$\sum_{j\in \boldsymbol{P_i}} \Delta_j = |\boldsymbol{D_i}| \text{ and } \sum_{i\in[1,\infty)} |\boldsymbol{D_i}| = |\mathbf{D}| \quad (4)$$

$$0 \le p_j \le 1 \text{ for each } j\in \boldsymbol{P_i} \text{ and } i\in[1,\infty) \text{ and } \sum_{j\in \boldsymbol{P_i}} \lceil p_j \rceil \le |\boldsymbol{P_i}| \quad (5)$$

The objective function (1) determines the average network delay for each source-destination pair. The constraint (2) guarantees that the data distribution for each s-d pair in a set of paths be less than that over a single, most stable path. The constraint (3) assures that the data distribution in multiple paths reaches the destination within the life of those paths. The constraint equation (4) mentions that the total data packets for each source-destination must be distributed in a set of paths. The constraint (5) indicates that when the value of the path p_j lies between 0 and 1, the path will be selected for the set $\boldsymbol{P_i}$, and the number of paths selected must not exceed the cardinality of the set $\boldsymbol{P_i}$.

3.2 Path Finding Algorithm

In this scheme, a source initiates a route discovery request when it needs to send data to a destination [1]. The source broadcasts route request packet to all neighboring nodes. Each route request packet contains source id, destination id, a request id with a locally maintained time-stamp, a route record to accumulate the sequence of hops through which the request is propagated during the route discovery, and a count of maximum hop (maximum hop = 4 is taken as an initial value in the simulation) which is decremented at each hop as it propagates. When maximum hop = 0, the search process terminates. The count of maximum hop thus limits the number of intermediate nodes (hop-count) in a path.

When any node receives a route request packet, it decrements maximum hop by 1 and performs the following steps:
1. If the node is the destination node, a route reply packet is returned to the source along the selected route, as given in the route record that now contains the complete path information between s and d.
2. Otherwise, if maximum hop = 0, the route request packet is discarded.

3. Otherwise, if this node id is already listed in the route record in the request, the route request packet (to avoid looping) is discarded.
4. Otherwise, the node id to the route record in the route request packet is appended and the request is re-broadcast.

When any node receives a route reply packet, it performs the following steps:

1. If the node is the source, it records the path to destination along with its time of arrival from locally maintained time. Thus, the time-delay between route request and route reply for a path is determined. This is the time required for the route discovery (d_j) between s and d. This is an indicator of the delay caused due to traffic congestion, packet transmission time and number of hops in the path under consideration.
2. If it is an intermediate node, it appends the value of affinity and propagates the packet to the next node listed in the route record to reach the source node.

The path-searching mechanism given above is same as described in [1] with three differences:

- the search is not restricted to finding the shortest path only; if multiple paths exist between source and destination, the source receives multiple path information from destination in sequence;
- the route reply packet from destination to source would collect the most recent value of affinity a_{mn} for all intermediate nodes m, n,…; and
- each path between source and destination is associated with a time-delay to estimate the delay associated with that path due to traffic congestion.

3.3 Multi-path Algorithm for Data Communication

The multi-path design algorithm is described as follows:

step I: initialize sets $\{\boldsymbol{D_i}\}=\Phi$ and $\{\boldsymbol{P_i}\}=\Phi$ where $i \in [1,\propto)$
step II: while $\sum_{i\in[1,\propto)} |\boldsymbol{D_i}| \leq |\mathbf{D}|$
step III: call the path-finding algorithm that gives a set of stable paths with route-discovery-time and number of hop-counts for those paths. This is the set of input variables to the optimization problem
step VI: call optimization problem Z with the set of input variables
step V: the paths p_j (j=1,2,…) are selected for the set $\boldsymbol{P_i}$
step VI: Δ_j's are determined for all paths p_j (j=1,2,…) for the set $\boldsymbol{P_i}$ and $\sum_{j\in \boldsymbol{P_i}} \Delta_j = |\boldsymbol{D_i}|$ for $i \in [1,\propto)$
step VII: if $\sum_{i\in[1,\propto)} |\boldsymbol{D_i}| = |\mathbf{D}|$ then go to step IX
step VIII: go to step III after transmitting ($|\boldsymbol{D_i}|$-T^{disc}*B/H^{ave}) packets
step IX: stop

3.4 An Illustrative Example

Let us assume that the total data volume is 5000 packets that needs to be routed from a source (s) to destination (d). We refer to figure 1 for explaining this algorithm. Let the (s-d) pair be (14-5) given in Figure 1. We call a path-finding algorithm that finds initially a set of stable paths with route discovery time and number of hop-counts of those paths. This path-finding algorithm finds five paths having route discovery time 68 ms, 138 ms, 201 ms, 262 ms and 273 ms respectively. Let these five paths be $path_1$: 14-8-5, $path_2$: 14-1-2-5, $path_3$: 14-9-13-10-5, $path_4$: 14-8-4-9-5 and $path_5$: 14-1-2-3-5; and these paths be stable for 1122 ms, 1682 ms, 3216 ms, 4228 ms and 5011 ms and have 2, 3, 4, 4 and 4 hops respectively. The optimization algorithm selects three appropriate paths. The path set $\boldsymbol{P_1}$ ={p_1, p_2 and p_5} is selected to route data packets from source-to-destination. The set of data packets $\boldsymbol{D_1}$ ={527 (Λ_1), 140 (Λ_2) and 764 (Λ_5)} is distributed into paths p_1, p_2 and p_5 respectively by solving the optimization algorithm. That is, the total 1431 packets of data have routed in the 1st iteration. Now $|\mathbf{D}| - |\boldsymbol{D_1}|$ (=$\sum_i \Lambda_i$), i.e., (5000-1431) will be distributed in the next few iterations. The path-finding algorithm is again called to find out a set of paths with route discovery time and number of hops traversed in those paths for the next iteration after transmitting (1431-300/2), i.e. 1281, packets where $|\boldsymbol{D_1}|$= 1431, T^{disc} = 300ms, B=1 packet per msec and H^{ave}=2.

The multi-path algorithm distributes the remaining amount of data packets from source to destination into the next few iterations in the same way described above. The algorithm terminates when $|\mathbf{D}| = |\boldsymbol{D_1}| + |\boldsymbol{D_2}| + |\boldsymbol{D_3}| + \ldots.$

3.5 A Solution Method using Lagrangean Relaxation and Subgradient Heuristics

The optimization problem in our multi-path algorithm has been solved by standard techniques such as Lagrangean relaxation [7] and subgradient heuristics [8], [12] which is described below.

The model described above is a nonlinear combinatorial optimization problem. Since the decision analog of this model belongs to the NP-complete class of problems, the Lagrangean relaxation is used to develop a heuristic solution of this problem. The Lagrangean relaxation (L) of the present problem (G) is obtained by multiplying constraints (2), (4) and (5) with vectors of Lagrangean multipliers λ_j, μ_j and η_j, respectively, and then by adding them to the objective function.

$$L = (\sum_{i\in[1,\infty)} \sum_{j\in \boldsymbol{P_i}} (d_j + \Delta_j H_j \tau_j p_j))^{\alpha} + \lambda_j (\sum_{i\in[1,\infty)} \sum_{j\in \boldsymbol{P_i}} (d_j + \Delta_j H_j \tau_j p_j) - \sum_{i\in[1,\infty)} \sum_{j\in \boldsymbol{P_i}} (d_{stab(j)} + |\boldsymbol{D_i}| H_{stab(j)} \tau_{stab(j)} p_j)) + \mu_j(\sum_{j\in \boldsymbol{P_i}} \Delta_j - |\boldsymbol{D_i}|) + \eta_j(\sum_{j\in \boldsymbol{P_i}} \lceil p_j \rceil - |\boldsymbol{P_i}|)$$

subject to the constraint (3)

The set of feasible solutions for the Lagrangean relaxation L of the problem (equation 1) is a super set of the set of feasible solutions for the problem. For the given vectors Γ_j {=λ_j, μ_j, η_j}, if the problem has a feasible solution Λ_j and p_j, then the following relationship holds:

$$L(\Lambda_j, p_j, \Gamma_j) < Z(\Lambda_j, p_j, \Gamma_j)$$

Thus, $L(\Lambda_j, p_j, \Gamma_i)$ is the lower bound of $Z(\Lambda_j, p_j, \Gamma_j)$ for each for each Γ_j. The best possible bound for such procedure is given by the vector Γ_R satisfying the following conditions:

$$L(\Lambda_j^*, p_j^*, \Gamma_j) < L(\Lambda_j^*, p_j^*, \Gamma_j^*) < L(\Lambda_j, p_j, \Gamma_j^*)$$

The point $(\Lambda_j^*, p_j^*, \Gamma^*_j)$ is called the optimal point of the Lagrangean because, for a unique point (Λ_j, p_j) the following relation holds:

$$L(\Lambda_j^*, p_j^*, \Gamma_j) < L(\Lambda_j^*, p_j^*, \Gamma_j^*)$$

The detail description of Subgradient heuristic is given in the appendix. A flow-chart is given below :

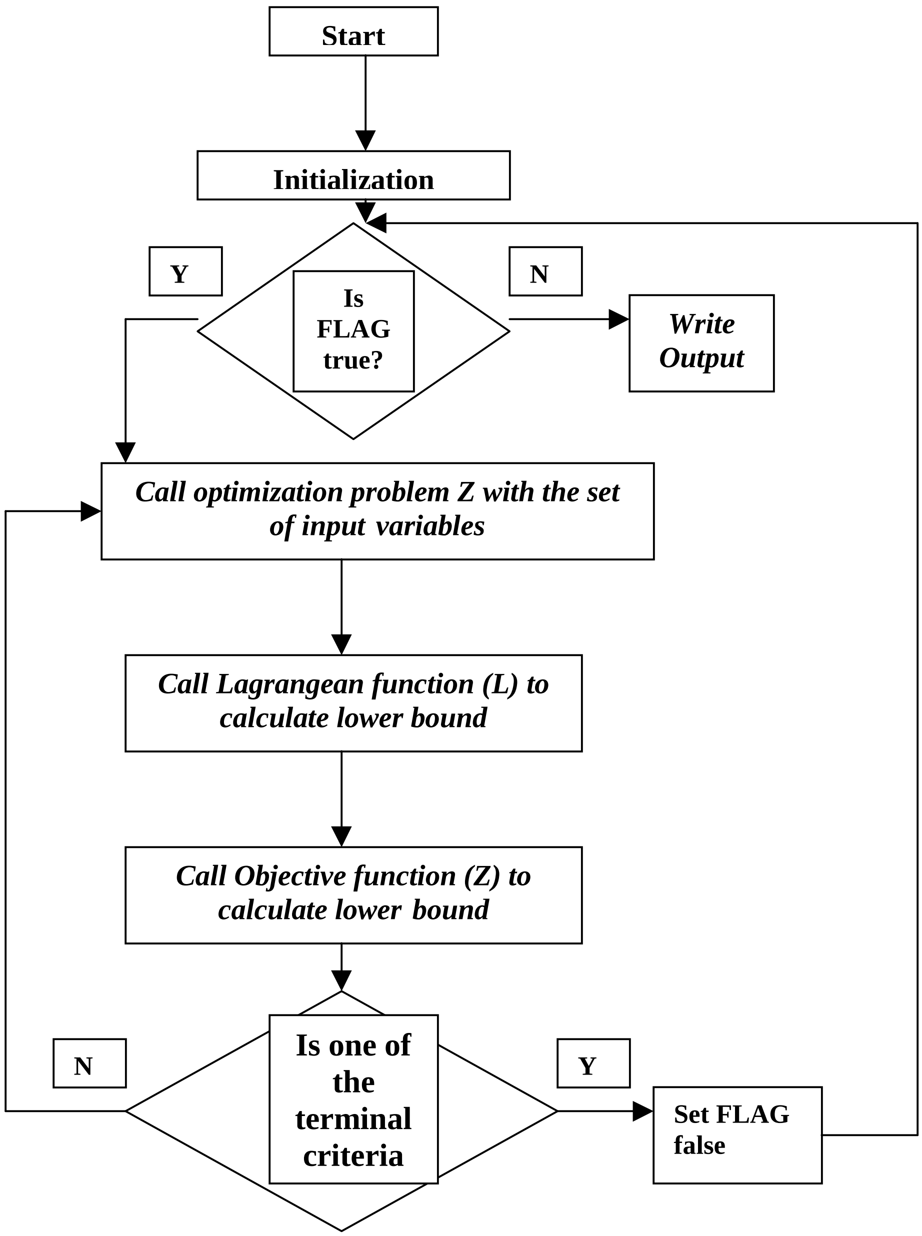

Figure 2 : Flow chart of subgradient algorithm

4. Simulation Results

The simulators suitable for the present context available in the market by and large are either wireless link simulators to model wireless link characteristics or network simulators to study networking algorithms and protocols in a static setting. Some efforts, in recent years, have been made to combine these two classes of simulators to model and study wireless mobile network characteristics [13]-[14]. These simulators are inadequate to model and study ad hoc wireless networks. The simulator, ns, for example, has initially provided no direct support for mobility or shared wireless radio channel. A recent release of this simulator [15] provides support for modeling wireless LANs; it cannot be used for studying multi-hop ad hoc networks directly. That's why we develop a simulation, which will study the different characteristics of mobile ad hoc networks.

In order to model and study extensively the performance of the proposed framework in the context of ad hoc wireless networks, we have developed a simulator [18] with the capability to model and study the following characteristics:

- Node mobility
- Link stability (*affinity*)
- Affinity- based path search
- Pre-emptive route discovery with multi-path routing
- Dynamic network topology depending on mobility and transmission range
- Physical and data link layers in wireless environment

The proposed framework has been evaluated on the above-mentioned simulated environment under a variety of conditions. The environment is assumed to be a closed area of (1000 x 1000) square meters in which mobile nodes are distributed randomly. We have run simulations for networks with 10, 20, 30 and 40 mobile hosts, operating at transmission ranges varying from 150 meters to 400 meters. The bandwidth for transmitting data is assumed to be 1000 packets / sec. The packet size is dependent on the actual bandwidth of the system. The propagation and processing delays are considered negligible.

In order to study the delay, throughput and other time-related parameters, every simulated action is associated with a simulated clock. The clock period (time-tick) is assumed to be one millisecond (simulated). For example, if the bandwidth is assumed to be 1000 packets per second and the volume of data to be transmitted from one node to its neighbor is 100 packets, it will be assumed that 100 time-ticks (100 milliseconds) are required to complete the task. The size of both control and data packets are same and one packet per time-tick will be transmitted from a source to its neighbors.

The speed of movement of individual nodes ranges from 5 m/sec to 20 m/sec. Each node starts from a home location, selects a random location as its destination and moves with a uniform, predetermined random velocity towards the destination. Once it reaches the destination, it waits there for a pre-specified amount of time, selects randomly another location and moves towards that. However, in the present study, we have assumed zero waiting time to analyze worst-case scenario and a uniform node velocity of 10 m/sec.

4.1 Uni-path Routing Scheme

In this section, we have presented some observations on unipath routing scheme using shortest path and stable path algorithms. Shortest path algorithm has captured the behavior of the general class of routing algorithms in the context of ad-hoc networks. Stable path algorithm relies on evaluating a path based on both link stability and path stability before initiating data communication.

Once the route discovery is successful and the data communication is initiated, the completion of data communication depends on the stability of the selected path. ***Communication Efficiency*** is defined as the ratio of the number of data communication successful to the number of data communications initiated. We have taken several run of the simulator, each time with a particular setting of number of nodes (N), transmission range (R), mobility (M) and data volume in packets (D). The N = 4, R = 6, M = 3 and value of D = 3 give rise to (4 x 6 x 3 x 3), i.e. 216 **time steps**. With each time step, we have studied shortest-path and stable-path algorithms with 10 communication events per minute (C), initiated in the simulated environment with source and destination selected randomly. So, a total of 2160 communication events in uni-path routing have been studied with shortest-path and stable-path algorithms.

At each data volume (100 packets, 1000 packets, 3000 packets), we have evaluated Communication Efficiency for both shortest-path and stable path algorithms, as shown in Table 1. The Communication Efficiency is much higher in stable-path algorithm. The advantage is more pronounced in case of higher data volumes: the number of route error generated in shortest- path algorithm with high data volume (3000 packets) is about 32%.

Table 1: Communication Efficiency for Uni-path Routing using shortest-path and Stable-path algorithm at different data volumes

Data Volume (packets)	Shortest-Path Routing				Stable-Path Routing			
	Route Discovery initiated	Route Discovery Successful	Comm Successful	Comm Efficiency	Route-Discovery initiated	Route-Discovery Successful	Comm Successful	Comm Efficiency
100	720	441	428	97.1 %	720	416	414	99.5 %
1000	720	434	371	85.5 %	720	315	312	99.0 %
3000	720	389	266	68.4 %	720	214	210	98.1 %

However, it is noted that the number of successful route discoveries is much less in stable path algorithm as compared to shortest path algorithm. This is because of the fact that the stable-path algorithm evaluates the path for sufficient stability before initiating data communication. For example, the possibility of finding out a stable path for sending 3000 packets of data is far less (214 out of 720 initiations in our case) as compared to that for sending 100 packets of data (416 out of 720 initiations).

The results indicate the effectiveness of stable-path algorithm in terms of reducing the route errors. However, we cannot send large volume of data at a stretch in uni-path routing scheme. Thus, in order to get uninterrupted connectivity between s and d, we need to use multi-path scheme, as explained next.

4.2 Proposed Multi-Path Routing Framework

The success of the framework relies on the fact that nodes s and d are always connected through some intermediate nodes, in spite of their motility. In other words, the intermediate nodes through which s-d are connected, may change with time, but s-d should remain connected. *Connectivity efficiency* in this context has been defined as the ratio of total number of connected node-pairs (in single hop or multiple hops) and the total number of active node pairs at any instant of time. This fraction captures the degree of connectivity among the nodes in any snapshot of the mobile environment. The efficiency values obtained over several snapshots (taken at intervals of one second from the simulator) of the dynamic environment have been finally averaged to yield the *average connectivity efficiency* of the network. A network, where all node-pairs are always connected in single or multiple hops, has an average connectivity efficiency of 100%. Based on our observation shown in Figure 3, we have selected a transmission range of 300 meters for N=30 in the following analysis to get 100% average connectivity efficiency**.**

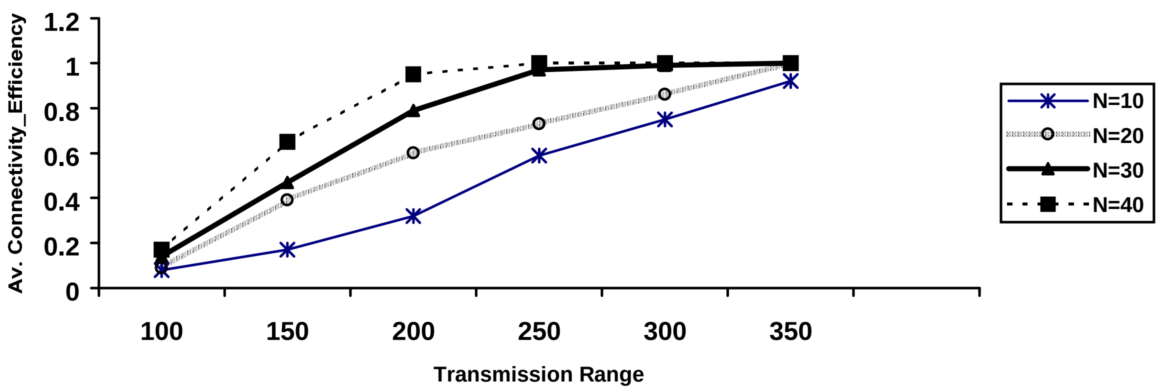

Figure 3. Average Connectivity Efficiency vs. Transmission Range for different number of nodes

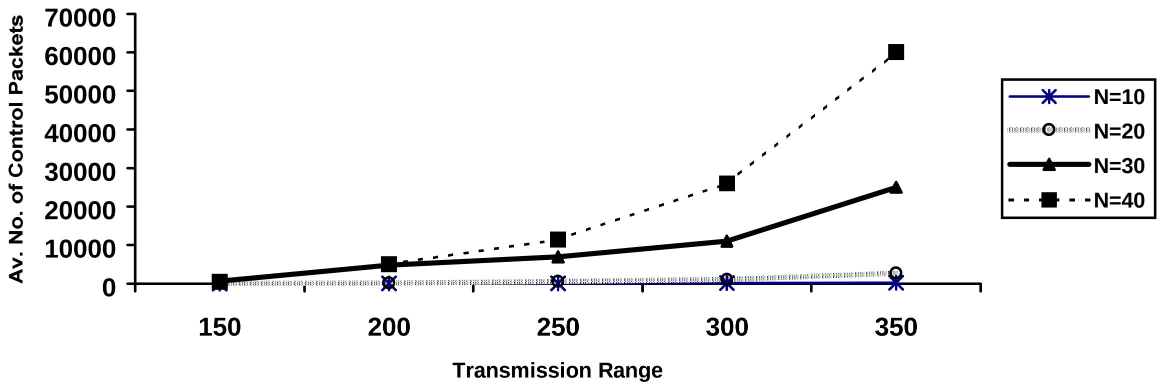

Figure 4. Average Number of Control Packets Generated per Communication vs. Transmission Range

R cannot be allowed to increase further due to other two overheads. First, cost (power consumption due to battery usage) increases as the transmission range is raised. Second, congestion and collision of control packets are the inevitable outcome of higher transmission range during data communication. Figure 4 shows the variation of average number of control packets generated per communication with transmission range with max_hop=4. For N=30, control packets grows drastically beyond R=300. In Figure 5, we have shown the stability of most stable path between two arbitrary (s-d) nodes 14 and 5 refereed the network shown in Figure 1, sampled at every 5-second interval of time. At each 5 seconds, a route discovery process is initiated from node 14 and the paths obtained after route-request–time-out (300 msec) is evaluated to find out the most stable path. As

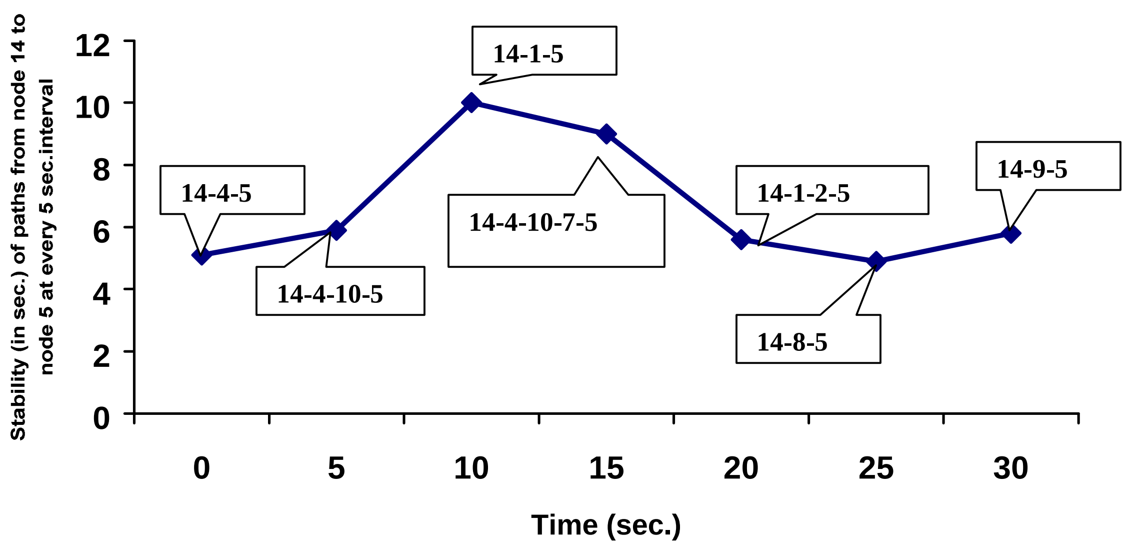

Figure 5. Stability of maximum stable path between node 14 and 5, sampled at each 5-second interval of time (the path is given in the boxes).

shown in the figure, it has been observed that no single path is stable throughout the span of 30 sec. However, we are getting a sustained stability between node 14 and 5 through different intermediate nodes. This establishes the viability of our scheme. In other words, if we can perform preemptive route re-discoveries before the occurrence of route errors while transmitting a large volume of data from s to d, it is possible to find out dynamically a sequence of multiple paths in temporal domain to complete a large volume of data transfer.

Table 2 shows an example case to illustrate the advantage of using temporal multi-path only, disregarding the spatial multi-path for the time being. So, the set $\boldsymbol{P_i}$ in this case consists of only one path, which is the most stable path**.** The total data volume, 10000 packets from source (node 24) to destination (node 5) in a 30-node system with an average mobility of 10m/sec, is communicated. No single path is found to be sufficiently stable to complete this large volume of data transfer. Thus, the source 24 needs to perform preemptive route discovery six times at different time intervals to find out dynamically a series of multiple paths in temporal domain to complete the data transfer. Each route given in the table is the most stable route at that instant of time.

Table 2. Total time required for sending 10000 data packets using Temporal Multi-paths only ($|D|$ = 10000)

$\boldsymbol{P_i}$ $\{d_j, S_j, p_j\}$	$\lvert \boldsymbol{D_i} \rvert$	**Time (in msec.) required for sending $\lvert \boldsymbol{D_i} \rvert$ packets of data (assuming $\tau_j = 1$ msec/packet)**
{249, 11922, 24-31-16-5}	3891	11673
{251, 4388, 24-32-39-5}	1379	4137
{238, 2845, 24-25-5}	1303	2606
{272, 6437, 24-31-37-33-5}	1541	6164
{297, 2958, 24-11-9-5}	887	2661
{247, 3358, 24-10-9-5}	999	2997
TOTAL	**$D=\sum \lvert D_i \rvert$ = 10000 packets**	**30238 msec.**

Table 3 shows the same example using both spatial and temporal multi-paths. It shows significant improvement over the first scheme that uses temporal multi-path alone. First, the number of route discovery required to complete the data transfer process has been reduced to four (from six in earlier case), which implies creation of less congestion due to control packet propagation. Second, the time required to complete the data transfer is 21114 msec, which is significantly less than that without spatial multi-path (30283 msec). So the average delay per packet decreases from 3 sec to 2.1 sec to illustrates the efficacy of the proposed scheme in reducing end-to-end delay while transmitting large volume of data. It is interesting to note that the addition of spatial multipath (Table 3) eliminates the longest path (viz., 24-31-37-33-5 in the 4th row of Table 2) used in the temporal multipath. This is another good indication of QoS improvement because longer paths are more vulnerable in ad hoc networks. Also, the shorter are the paths used, the less is the chance of blocking perceived by end-users.

Table 3. Improvement in total time required for sending 10000 data packets using both Spatial and Temporal Multi-paths ($|D|$ = 10000)

P_i {d_j, S_j,p_j}			Λ_j			$\lvert D_i \rvert$	**Time reqd for sending $\lvert D_i \rvert$ pkts data (τ_j = 1 ms/pkt)**
p_1	p_2	p_3	Λ_1	Λ_2	Λ_3		
{2, 3096, 24-5}	{56, 6864, 24-39-5}	249, 11922, 24-31-16-5}	3095	1058	2154	6307	11673
{117, 2409, 24-25-5}	{251, 4388, 24-32-39-5}	x	1145	615	x	1760	4135
{238, 2845, 24-25-5}	x	x	1303	x	x	1303	2606
{289, 3200, 24-26-9-5}	x	x	900	x	x	900	2700
TOTAL						**$\lvert D \rvert = \sum \lvert D_i \rvert$ = 10000 pkts**	**21114 msec**

5. Conclusion

In this paper, we have introduced a notion of temporal and spatial multi-path routing in ad hoc wireless network and described an adaptive framework to evaluate the suitability of using spatial multiple paths with an objective to minimize end-to-end delay. We have found that:

- Use of temporal multi-paths allows a source to transmit a large volume of data to a destination without much degradation of performance due to route-errors;
- Use of spatial multi-paths help reduce route-rediscovery and end-to-end delay significantly.

The average per-hop delay per packet (τ_j) has been assumed to be 1 msec/packet, which might increase because of different load situation at different nodes. Currently, we are on simulating the multi-path algorithm under different load situations to see the effect on end-to-end delay. However, even if τ_j is more, it will not affect the effectiveness of the proposed scheme.

References

[1] D. B. Johnson and D. Maltz, “Dynamic source routing in ad hoc wireless networks”, T. Imielinski and H. Korth, eds., *Mobile computing*, Kluwer Academic Publ. 1996.

[2] S. Corson, J. Macker and S. Batsell, “Architectural considerations for mobile mesh networking”, Internet Draft RFC Version 2, May 1996.

[3] Z.J.Haas, “A new routing protocol for the reconfigurable wireless networks”, *ICUPC’97*, San Diego, CA, Oct. 1997.

[4] C-K Toh, “A novel distributed routing protocol to support ad-hoc mobile computing”, *IEEE International Phoenix Conference on Computer & Communications (IPCCC’96*).

[5] R. Dube, C.D. Rais, K. Wang and S.K. Tripathi, “Signal stability based adaptive routing for ad hoc mobile networks”, Technical Report CS-TR-3646, UMIACS-TR-96-34, Institute for Advanced Computer Studies, Department of Computer Science, University of Maryland, College Park, MD 20742, USA, August, 1996.

[6] N.S.V.Rao and S.G. Batsell, “QoS Routing via multiple paths using bandwidth reservation”, in Proc. of the *IEEE INFOCOM*, 1998.

[7] M.L. Fisher, “An applications oriented guide to Langrangean relaxation”, *Interfaces*, vol 15, pp 10-21, l985.

[8] H. Held, P. Wolfe, and H.P. Crowder, “Validation of subgradient optimization”, *Math. Program*, vol 5, pp 62-68, 1974.

[9] S. Bahk and W. El-Zarki, “Dynamic multi-path routing and how it compares with other dynamic routing algorithms for high speed wide-area networks”, in Proc. of the *ACM SIGCOM*, 1992

[10] H. Chu and K. Nahstedt, “Dynamic multi-path communication for video traffic”, In Proc. of the *Hawaiian Int. Conf. On System Science*, 1997.

[11] S. Murthy and J.J. Garcia-Luna-Aceves, “Congestion-oriented shortest multi-path routing”, in Proc. of the *IEEE INFOCOM*, 1996.

[12] D. Saha and A. Mukherjee, “On the multidensity gateway location problem for a multilevel high speed network”, *Computer Communications*, vol 20, pp576-87, 1997.

[13] J. Short, R. Bagrodia and L. Kleinrock. “Mobile Wireless Network System Simulation.” *Wireless Network Journal* 1, no. 4, 1995.

[14] M. Srivastava, P. Mishra; P. Agrawal and G. Nguyen. “Ethersim: A simulator for application-level performance modeling of wireless and mobile ATM networks” *Computer Networks and ISDN Systems* 29: 2067-2090, 1998.

[15] J. Broch, D.A. Maltz, D.B. Johnson, Y. Hu and J. Jetcheva, “A performance comparison of multi-hop wireless ad hoc network routing protocols” In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking* (Mobicom’98), Dallas, Texas, Oct. 25-30, 1998.

[16] S.K. Das, A. Mukherjee, S. Bandyopadhyay, K. Paul, D. Saha. “Improving quality-of-service in ad hoc wireless networks with adaptive multi-path routing”, in the Proceeding of *GLOBECOM 2000*, USA.

[17] K. Paul, S. Bandyopadhyay, A. Mukherjee and D. Saha, “A stability-based distributed routing mechanism to support unicast and multicast routing in ad hoc wireless network”, *Computer Communications*, vol. 24, December 2001, pp 1828-1845.

[18] K. Paul, “Some studies on the dynamics of mobile hosts for effective communication in ad-hoc wireless environment”, *Ph. D Dissertation*, CSE Dept, Jadavpur Univ., Calcutta, India, 2001.

Appendix

Subgradient Heuristic

Let us suppose that Γ^{*}_{j} is an optimal solution of the Lagrangean relaxation L. A subgradient optimization algorithm is used to derive lower bounds on the optimal primal objective value using L. In the subgradient optimization procedure, gradient method is adapted by replacing the gradients with subgradients. If an initial multiplier vector Γ^{0}_{j} is given, a sequence of multipliers is generated using the following expressions:

$$\lambda^{m+1}_{j} = \lambda^{m}_{j} + t_m \left(\sum_{i\in[1,\propto)} \sum_{j\in \boldsymbol{P_i}} (d_j + \Delta_j H_j \tau_j p_j) - \sum_{i\in[1,\propto)} \sum_{j\in \boldsymbol{P_i}} (d_{stab(j)} + \left|\boldsymbol{D_i}\right| H_{stab(j)} \tau_{stab(j)} p_j)\right)$$

$$\mu^{m+1}_{j} = \mu^{m}_{j} + t_m \left(\sum_{j\in \boldsymbol{P_i}} \Delta_j - \left|\boldsymbol{D_i}\right|\right)$$

$$\eta^{m+1}_{j} = \eta^{m}_{j} + t_m \left(\sum_{j\in \boldsymbol{P_i}} \lceil p_j \rceil - |\boldsymbol{P_i}|\right)$$

where (d_i, p_i) are obtained from an optimal solution to Lagrangean relaxation L, and t_m, a positive scalar, is a stepsize, which is given as follows :

$$t_m = \delta_m (G - L) / [\{(\sum_{i\in[1,\propto)} \sum_{j\in \boldsymbol{P_i}} (d_j + \Delta_j H_j \tau_j p_j) - \sum_{i\in[1,\propto)} \sum_{j\in \boldsymbol{P_i}} (d_{stab(j)} + \left|\boldsymbol{D_i}\right| H_{stab(j)} \tau_{stab(j)} p_j))^2 + (\sum_{j\in \boldsymbol{P_i}} \Delta_j - \left|\boldsymbol{D_i}\right|)^2 + (\sum_{j\in \boldsymbol{P_i}} \lceil p_j \rceil - |\boldsymbol{P_i}|)^2]^{1/2},$$

where δ_m is a scalar satisfying $0< \delta_m <2$. Initially, this scalar is set equal to 2, and it is then halved when the lower bound does not improve in a given number of consecutive iterations. The subgradient algorithm is terminated, if either the gap between the upper bound (the value of L) and the best primal feasible solution value are within a given specified limit. The algorithm for subgradient optimization problem is given next.

Algorithm of Subgradient Optimization Problem

Step 0: Initialization:

set G to an arbitrary large value
select an initial set off multipliers λ_j, μ_j and η_j
initialize the iteration counter n to 0
set improvement counter Δ to 0
set λ^{*}_{j}, μ^{*}_{j} and η^{*}_{j} to λ_j, μ_j and η_j respectively
set the current best value of L(λ_j, μ_j and η_j) to 0
set stepsize δ_m to δ^{0}_{m}

Step k (k>1): Solving the Lagrangean Relaxation

(k.1) increment the improvement counter by 1: $\Delta \leftarrow \Delta+1$
(k.2) find out (Λ_j, p_j) by solving $L(\lambda_j, \mu_j$ and $\eta_j)$
(k.3) Updating the parameters

(k.3.1) If $L(\lambda^n{}_j, \mu^n{}_j$ and $\eta^n{}_j)$ is greater than the current best value of $L(\lambda_j, \mu_j$ and $\eta_j)$ then $L(\lambda_j, \mu_j$ and $\eta_j)$ is replaced by $L(\lambda^n{}_j, \mu^n{}_j$ and $\eta^n{}_j)$. Also set $\lambda^*{}_j, \mu^*{}_j$ and $\eta^*{}_j$ to $\lambda^n{}_j, \mu^n{}_j$ and $\eta^n{}_j$ respectively

(k.3.2) If (Λ_j, p_j) is feasible to the minimization problem, then it its associated objective function for the minimization problem is computed. If this value is less than current best value of G then G is set to this value

(k.3.3) If improvement has reached a prespecified upper limit then set δ_m by $\delta_m/2$ and the improvement counter Δ is set to 0 and the procedure restarts from step (k.1)

(k.3.4) If iteration counter > a prespecified limit
or, if δ_m < a prespecified limit
or, if t_m < a prespecified limit
or, if $G- L(\Lambda_j, p_j, \Gamma_j) / \lVert L(\Lambda_j, p_j, \Gamma_j) \rVert^2$ < a prespecified error tolerance then
goto end step i.e., the process terminates

(k.4) Upgrading the multipliers

(k.4.1) Compute the new subgradients as:
$\gamma^m{}_A = (\sum_{j\in \boldsymbol{P_i}} (d_j + \Delta_j H_j \tau_j p_j) - \sum_{j\in \boldsymbol{P_i}} (d_{min(j)} + |\boldsymbol{D_i}| H_{min(j)} \tau_{imin(j)} p_j))$
$\gamma^m{}_B = (\sum_{j\in \boldsymbol{P_i}} \Delta_j - |\boldsymbol{D_i}|)$
$\gamma^m{}_C = (\sum_{j\in \boldsymbol{P_i}} \lceil p_j \rceil - |\boldsymbol{P_i}|)$

(k.4.2) Compute the stepsize as:
$t_m = \delta_m (G - L) / [\{(\sum_{j\in \boldsymbol{P_i}} (d_j + \Delta_j H_j \tau_j p_j) - \sum_{j\in}(d_{stab(j)} + |\boldsymbol{D_i}| H_{stab(j)} \tau_{stab(j)} p_j))^2 + (\sum_{j\in \boldsymbol{P_i}} \Delta_j - |\boldsymbol{D_i}|)^2 + (\sum_{j\in \boldsymbol{P_i}} \lceil p_j \rceil - |\boldsymbol{P_i}|)^2]^{1/2}$,

(k.4.3) Compute the new multipliers as:

$$\lambda^{m+1}{}_i = \min(-1, \lambda^m{}_i + t_m \gamma^m{}_A)$$
$$\mu^{m+1}{}_i = \min(-1, \mu^m{}_i + t_m \gamma^m{}_B)$$
$$\eta^{m+1}{}_i = \min(-1, \eta^m{}_C + t_m \gamma^m{}_C)$$

(k.4.4) Increment iteration counter by 1

$$k \rightarrow k+1$$

(k.5) goto step (k.1)

Solution of Lagrangean Relaxation

The solution of Lagrangean relaxation can be obtained by solving $\nabla\Delta_j L = 0$ and $\nabla p_j L = 0$, which find out Δ_j and p_j respectively. A detailed description is given in [12].