An adaptive framework for QoS routing through multiple paths in ad hoc wireless networks

S.K. Das,[a,*,1] A. Mukherjee,[b,1] S. Bandyopadhyay,[c] D. Saha,[c] and K. Paul[d]

[a] *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*
[b] *IBM Global Services, Calcutta, India*
[c] *MIS Group, Indian Institute of Management Calcutta, Calcutta 700 104, India*
[d] *School of Information Technology, IIT-Bombay, Powai, Mumbai 400 076, India*

Abstract

We propose an adaptive framework for computing multiple paths in *temporal* and *spatial* domains to transmit a large volume of data packets from a source s to a destination d in ad hoc wireless networks. The objective is to achieve quality of service (QoS) by minimizing end-to-end delay for packet delivery. We consider two aspects in this framework. The first aspect is to perform preemptive route rediscoveries before the occurrence of route errors while transmitting a large volume of data from s to d. This helps us to find out dynamically a series of possible 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* is also defined, and a unified mechanism is proposed to address the above two aspects that relies on evaluating a path based on link and path stability. Our solution method uses Lagrangean relaxation and subgradient heuristics to solve an optimization formulation of the problem in order to compute the paths and the corresponding data distribution, both in temporal and spatial domains. Simulation experiments demonstrate that the proposed framework helps in significantly reducing the end-to-end delay and the required number of route-rediscoveries.

Keywords: Ad hoc networks; Link and path stability; Multipath routing; Performance optimization; Quality of service

1. Introduction

There has been a growing interest in ad hoc networks in recent years [2,4,10,13]. An ad hoc network can be envisioned as a collection of mobile nodes or routers (each equipped with a wireless transceiver), which are free to move about arbitrarily. Two nodes, willing to communicate, may be outside the transmission range of each other; however, they can communicate via 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 s or destination d during data transfer. This interruption in communication results in degraded QoS because the user has to wait for a subsequent rediscovery of another route between this (s–d) pair. Some path-maintenance algorithm has to be invoked to prevent the disruption of communication, which eventually increases the end-to-end delay.

The majority of the existing schemes proposed for ad hoc networks uses single-path routing [8,10,18], which

might not reduce the average end-to-end delay. However, once a set of (s–d) paths is discovered in anticipation, in some cases, it is useful to employ the concept of *multipath* communication [3]. In fact, earlier research [5,11,14] have shown that it is possible to improve end-to-end performance 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 as well as end-to-end delay. Although utilization of multiple paths has been explored in the wired networks [1,3,11], in order to provide improved performance against single path, it has also been shown in [14] that deployment of multiple paths does not necessarily result in a lower end-to-end delay. Furthermore, most of the existing work considers only spatial multipath, whereas our work takes into account both spatial and temporal multipaths.

In this paper, we propose an adaptive framework for computing multiple paths (both in temporal and spatial domains) to transmit a large volume of data between (s–d) pairs in ad hoc wireless networks. Data transmission in the *spatial* domain balances traffic load in the network, whereas that in the *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 rediscoveries before the occurrence of route errors while transmitting a large volume of data from s to d. Consequently, this helps us find out dynamically a series of paths in the temporal domain to complete the data transfer. The second aspect of the framework 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. Lagrangean relaxation [7] and subgradient heuristics [9] methods are used to compute multiple paths and data distribution along them, both in temporal and spatial domains. Simulation experiments demonstrate that the proposed framework helps in reducing the end-to-end delay by a factor of 0.7. The number of route-rediscoveries due to route errors is also reduced, which in turn reduces the network congestion due to control packets.

A preliminary version of this work has been presented in [5].

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. 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 the node pair (s–d) needs 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), say. This interruption in service eventually degrades the quality of service [6]. Instead, if the routing scheme considered the mobility pattern of the intermediate nodes and also the traffic congestion, it might be able to find better paths a priori, having a longer life and perhaps less congestion 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, where $s, d \in N$
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., $P = \{P_1, P_2, P_3, \ldots\} = \{P_i \mid i \in [1, \infty)\}$
P_i	ith set of stable paths, $P_i \in P$, for $i \in [1, \infty)$
p_{ij}	binary indication variable for jth path in $P_i \in P$ where $i \in [1, \infty)$ and $j \in [1, \lvert P_i \rvert]$, such that $p_{ij} = \begin{cases} 1, & \text{where path } p_{ij} \text{ is selected between } s, d \text{ for the set } P_i, \\ 0 & \text{otherwise.} \end{cases}$
D	super set of a set of data volume in packets, i.e., $D = \{D_1, D_2, D_3, \ldots\} = \{D_i \mid i \in [1, \infty)\}$
D_i	ith set of data volume in packets, where $D_i \in D$ for $i \in [1, \infty)$
Δ_{ij}	data distribution for the jth path in the set P_i and $\sum_{j \in P_i} \Delta_j = \lvert D_i \rvert$ for $i \in [1, \infty)$
R	transmission range of nodes (assumed to be equal for all nodes)
M	average velocity of nodes
a_{mn}	affinity of a link $l_{mn} \in L$
B	link bandwidth in packets/ms (assumed to be identical for all links)
H_{ij}	number of hops traversed by jth path in the set P_i, where $1 \leqslant j \leqslant \lvert P_i \rvert$
H_{ij}^{ave}	average number of hops traversed by any jth path in any set P_i
$H_{stab(j)}^{i}$	number of hops traversed by the most stable jth path in the set P_i
τ_{ij}	average delay per hop per packet for jth path in the set P_i
$\tau_{stab(j)}^{i}$	average delay per hop per packet for the most stable jth path in the set P_i

r_{ij} route discovery time for jth path in the set P_i

q_{ij} average queuing delay per packet per node for jth path in the set P_i

T_{ij} average path delay per packet for jth path in the set P_i such that $T_{ij} = H_{ij} * \tau_{ij}$

S_{ij} stability for jth path in the set P_i

$r^i_{stab(j)}$ route discovery time for the most stable jth path in the set P_i

T_{disc} route-request-time-out for any source node waiting for route reply packets after issuing a route-request.

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 $n \in N$ has a unique identifier. Two nodes n and m are connected by two unidirectional links $l_{nm} \in L$ and $l_{mn} \in L$ such that n can send message to m via l_{nm} and m can send message to n via l_{mn}. In this paper, for the sake of simplicity, we have assumed symmmetric link with same capacity between n and m.

In a wireless environment, each node n is characterized by a transmission range, R_n, which is assumed to be the same as R for all nodes. The neighbors of n are the set of nodes within its transmission range. It is assumed that when node n transmits a packet, it is broadcast to all of its neighbors. However, the strength of connections to all members of the neighbor set with respect to any node n may not be uniform. For example, a node m in the periphery of the transmission range of n is weakly connected to n compared to another node, say u, which is closer to n. Thus, the probability 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 simply *affinity* which is defined in terms of link affinity. Informally speaking, *link-affinity*, $a_{nm(t)}$, associated with a link l_{nm} at time t, estimates the life span of the link l_{nm}. It is a function of the current distance (δ_{nm}) between nodes n and m, the relative mobility of m with respect to n, and the transmission range of n. If the transmission range of n and m are different, $a_{nm(t)}$ may not be equal to $a_{mn(t)}$. The *node-affinity*, $\eta_{nm(t)}$, between n and its neighbor m is then defined as $\eta_{nm(t)} = \min[a_{nm(t)}, a_{mn(t)}]$ which determines the stability of connectivity between them. The unit of affinity is in seconds.

To compute the link-affinity, $a_{nm(t)}$, at any instant of time, node n sends a periodic beacon and node m samples periodically the signal strength received from node n. Since the signal strength of n as perceived by m is a function $f(R_n, \delta_{nm})$, the node m can predict the current distance δ_{nm}. If M is the average velocity of the nodes, the worst-case link-affinity is given by $a_{nm(t)} = (R_n - d_{nm})/M$, 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 $R = 300$ m, the average velocity is $M = 10$ m/s, $\delta_{nm} = 100$ m, then he life span of link l_{nm} (worst-case) is $a_{nm(t)} = 20$ s, 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 based on an optimistic assumption that node-distance can be deduced from signal strength. In reality, however, even if the transceivers have the same transmission range, the signal strength varies because of the differences in battery power. Therefore, a node m may also need to monitor the change in signal strength of n over time to estimate the link-affinity. (This idea was originally described in [12].)

Let $\Delta S_{nm}(t)$ be the change in signal strength at time t. Then $\Delta S_{nm}(t) = S_{nm}(t) - S_{nm}(t - \Delta t)$, 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 $S'_{nm}(t) = (\Delta S_{nm}(t)/\Delta t)$ be the rate of change of signal strength at time t and let $S'_{nm}(t)_{\text{avg}}$ be the average rate of change of signal strength at time t over the past few samples. Let S_t be the *threshold-signal-strength* such that when the S_{nm} associated with l_{nm} goes below S_t, we assume that the link l_{nm} is disconnected. Furthermore, we define

$$a_{nm}(t) = \begin{cases} \text{high} & \text{if } S'_{nm}(t)_{\text{avg}} \geqslant 0, \\ (S_t - S_{nm}(t))/S'_{nm}(t)_{\text{avg}} & \text{if } S'_{nm}(t)_{\text{avg}} < 0. \end{cases}$$

If $\Delta S_{nm}(t)_{\text{avg}} \geqslant 0$, it indicates that the link-affinity is either increasing or constant and the two nodes are either coming closer or static. Hence, link-affinity is termed as *high* at that instant of time. The value for *high* is computed as the ratio of the transmission range to the 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 $S'_{nm}(t)_{\text{avg}}$ is positive, 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. Hence, 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 u. Therefore, if $S'_{nm}(t)_{\text{avg}} > 0$, a correction factor, μ, is used to moderate this *high* value. This correction factor is given by $\mu = (1 - S_t/S_{nm}(t))$, implying $a_{nm}(t) = \mu * high$ if $S'_{nm}(t)_{\text{avg}}$ is positive. This indicates that if $S_{nm}(t) \approx S_t$, then $\mu \to 0$ and consequently $a_{nm}(t) \to 0$, even if $S'_{nm}(t)_{\text{avg}} > 0$. Recall that the *node-affinity* or *affinity* between two adjacent nodes n and m is defined as $\eta_{nm}(t) = \min[a_{nm}(t), a_{mn}(t)]$.

Given a path $\mathscr{P} = (s, i_1, i_2, \ldots, i_k, d)$ between s and d, its stability will be determined by the lowest-affinity link

(since that is the bottleneck for the path). It is defined as $\eta^{\mathscr{P}}_{sd}(t) = \min[\eta_{s_i}(t), \eta_{i_1 i_2}(t), .., \eta_{i_k d}(t)]$.

However, the notion of path stability is dynamic and service-specific. It is dynamic because the stability of a path is its life span from a given instant of time. Moreover, stability has to be seen in the context of providing service. A path between s and d is said to be stable at an instant of time, if its span of life is sufficient to complete such service as transferring the required volume of data between s and d.

If a path is not stable enough to carry a large volume of data between s and d but yet a communication is initiated, some form of route maintenance scheme has to be deployed to repair the path or find out an alternative path in case of a route error. This interruption in service and its resumption after route re-discovery eventually degrades the average QoS. Instead, if it is possible to predict the life span of a path between s and d, and accordingly preempt the process of route re-discovery so as to discover a new (s–d) path before the existing path breaks, it may be possible to provide an uninterrupted (better quality) service of data communication between s and d.

Additionally, once a set of paths between s and d has been discovered, in some cases, it is possible to improve the end-to-end delay by splitting the volume of data further into different blocks and send them via selected multiple paths between these two nodes. This eventually reduces the congestion and end-to-end delay. Of course, depending on the topology, we need to decide dynamically whether or not multipath is a preferred mode of communication.

2.3. Communication delay

To start with, let us assume a static network setting where nodes are not mobile. In this section, for simplicity, we will omit i from the subscripts (as defined in Section 2.1) when a generic path will be used and there will be no ambiguity.

If q_j is the average queuing delay (in ms) per packet per node in the jth path from s to d, B is the bandwidth (in packets/ms) and τ_j is the average delay (in ms) per hop per packet of a traffic stream for jth path from s to d, then $\tau_j = q_j + 1/B$, ignoring propagation and processing delays.

The first component q_j is the queuing delay and the second component $1/B$ 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 jth path from s to d with H_j number of hops, then $T_j = H_j\tau_j$.

As an example, consider the network in Fig. 1 where a source (14) is sending data to a destination (5) at 4 hops away from the source (say, via the path 14–1–2–3–5) using RTS/CTS/ACK protocol. When node 14 is transmitting a data packet, node 1 has to accept if it 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 of establishing a transmission in pipelined fashion, if the number of hops is more than three. Furthermore, there is also a chance that node 14 will complete its entire data transfer activity to node 1 before the latter gets chance to transmit. So, the worst-case delay per packet is $H_j\tau_j$.

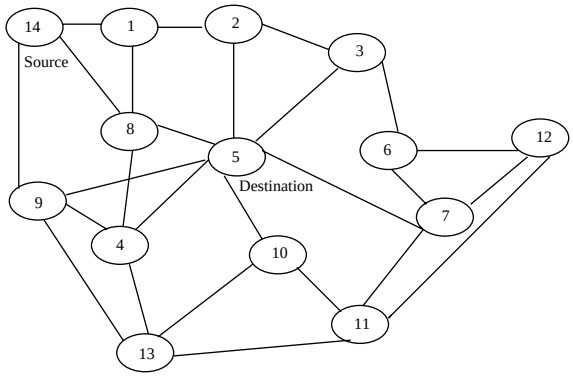

Fig. 1. Data communication from source (14) to destination (5).

Next, let us consider a dynamic scenario where nodes are mobile. It implies that an intermediate node in the jth path participating in a communication between s and d, may suddenly move out of range or switch itself off in between transfer of a message. In other words, the jth 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 D be the number of packets to be sent between s and d; let T_{sd} be the average path delay per packet between them; r_{sd} be the average route discovery time for discovering a set of paths between them; k be the number of times the route rediscovery is required for completing data transfer; 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 $|D|$ packets of data transfer along the (s–d) path is given by $|D|T_{sd} + kr_{sd} + kt_{sd}$.

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

- It performs preemptive route rediscoveries that overlap with data communication in temporal domain. In other words, $|D|T_{sd}$ and kr_{sd} overlap in time domain.
- Since the framework avoids the occurrence of route-errors, $t_{sd} = 0$.
- Because multiple paths are used in spatial domain, T_{sd} can be reduced significantly, as explained below.

3. Framework for multipath routing

This section describes an adaptive framework for multipath routing using a solution method based on Lagrangean relaxation and subgradient heuristics.

3.1. Problem formulation

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

Each temporal stable path set, say P_i, has a number of stable paths p_{ij} for $j = 1, 2\ldots, |P_i|$. The jth stable path is selected when $p_{ij} = 1$ and the maximum number of paths selected in the set P_i is equal to $|P_i|$. We call these as *spatial paths* in the temporal path set P_i. On these paths, the data volume $|D_i|$ is further distributed into Δ_{ij} (for $j = 1, 2\ldots$), called spatial data volume, and the sum of all Δ_{ij}'s is equal to $|D_i|$. The parameters S_{ij}, r_{ij} and H_{ij} for the path p_{ij} are as explained earlier.

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 is available before the completion of current data volume. Let us assume that T_{disc} (in ms) 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 P_i. Let us also assume that B is the bandwidth measured in terms of the number of packets per ms, and let H^{ave} be the average number of hops traversed by any jth path in any set P_i. For any set P_i of paths with data volume $|D_i|$ to be communicated, the route-rediscovery for the next set P_{i+1} are initiated after transmitting $(|D_i| - T_{\text{disc}}B/H^{\text{ave}})$ amount of data packets which ensure that the next set P_{i+1} is available to source just before completion of transmitting $|D_i|$ volume of data packets. For simplicity, we assume $B = 1$ packet/ms.

Based on the above notations and terminology, let us formally describe the problem as follows:

> Minimize the average delay of a wireless mobile ad hoc network by distributing data packets into a set of temporal paths for each source–destination pair (s–d). Each of these paths is further divided into a number of spatial paths through which smaller data blocks are sent. A set of stabilized paths with route discovery time and a number of hops traversed along these paths have been taken as input to the problem.

Formally, this optimization problem is defined as

Minimize

$$Z = \left(\sum_{i \in [1,\infty)} \sum_{j \in P_i} (r_{ij} + \Delta_{ij} H_{ij} \tau_{ij} p_{ij}) \right)^{\alpha} \tag{1}$$

for each (s–d) pair, where α is the degree of the complexity of the optimization problem,

Subject to

$$\sum_{i \in [1,\infty)} \sum_{j \in P_i} (r_{ij} + \Delta_{ij} H_{ij} \tau_{ij} p_{ij}) < \sum_{i \in [1,\infty)} \sum_{j \in P_i} (r^i_{stab(j)} + |D_i| H^i_{stab(j)} \tau^i_{stab(j)} p_{ij}), \tag{2}$$

$$\forall i \in [1, \infty) \left[\forall j \in P_i (r_{ij} + \Delta_{ij} H_{ij} \tau_{ij} p_{ij}) < S_{ij} - \sum_{k=1}^{j-1} (r_{ik} + \Delta_{ik} H_{ik} \tau_{ik} p_{ik}) \right], \tag{3}$$

$$\sum_{j \in P_i} \Delta_{ij} = |D_i| \tag{4}$$

and

$$\sum_{i \in [1,\infty)} |D_i| = |D|, \tag{5}$$

where

$$0 \leqslant p_{ij} \leqslant 1 \tag{6}$$

for each

$$j \in P_i, i \in [1, \infty) \quad \text{and} \quad \sum_{j \in P_i} p_{ij} \leqslant |P_i|. \tag{7}$$

The objective function (1) determines the average network delay for each source–destination pair. 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. Constraint (3) assures that the data distribution in multiple paths reaches the destination within the life of those paths. Constraint equation (4) mentions that the total data packets for each source–destination must be distributed in a set of paths. Constraint (5) indicates that the number of paths selected must not exceed the cardinality of the set 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. 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, 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. If maximum hop $= 0$, the route request packet is discarded.
3. If this node id is already listed in the route record in the request, the route request packet (to avoid looping) is discarded.
4. The node id to the route record in the route request packet is appended and the request is rebroadcast.

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 (r_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 above path-searching mechanism is the same as described in [10] 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 $m, n, \ldots$; 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. Multipath algorithm for data communication

The multipath routing algorithm is described as follows.

Step 1: Initialize sets $\{D_i\} = \Phi$ and $\{P_i\} = \Phi$, where $i \in [1, \infty)$.

Step 2: While $\sum_{i\in[1,\infty)} |D_i| \leqslant |D|$.

Step 3: Call the path-finding algorithm (as given in Section 3.2) 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 4: Call the optimization problem Z with the set of input variables.

Step 5: The paths p_{ij} (for $j = 1, 2, \ldots$) are selected for the set P_i.

Step 6: Δ_{ij}'s are determined for all paths $p_{ij} (j = 1, 2, \ldots)$ for the set P_i such that $\sum_{j\in P_i} \Delta_{ij} = |D_i|$ for $i \in [1, \infty)$.

Step 7: If $\sum_{i\in[1,\infty)} |D_i| = |D|$ then Stop.

Step 8: Go to Step 3 after transmitting $(|D_i| - T^i_{\text{disc}} B/H^{\text{ave}})$ packets.

Step 9: Stop.

3.4. An illustrative example

Let us assume that the total data volume is $|D| = 5000$ packets that needs to be routed from source $s = 14$ to destination $d = 5$, as shown in the network in Fig. 1. We call a path-finding algorithm that selects an initial set of stable paths with route discovery time and number of hop-counts. In our example network, the path-finding algorithm results in five paths having route discovery time $r_j = 68$, 138, 201, 262, and 273 ms, respectively. Let these five paths be $p_1 = 14$–8–5, $p_2 = 14$–1–2–5, $p_3 = 14$–9–13–10–5, $p_4 = 14$–8–4–9–5 and $p_5 = 14$–1–2–3–5. Let they be stable for 1122, 1682, 3216, 4228, and 5011 ms with hop counts 2, 3, 4, 4 and 4, respectively. The optimization algorithm selects the path set $P_1 = \{p_1, p_2$ and $p_5\}$ to route data packets from source to destination. The set of data packets $D_1 = \{$ 527 $(= \Delta_1)$, 140 $(= \Delta_2)$ and 764 $(= \Delta_5)\}$ is distributed into the corresponding paths p_1, p_2 and p_5. A total $|D_1| = 1431$ packets of data is routed in the first iteration. Now $|D| - |D_1| = \sum_i \Delta_i) = (5000 - 1431) = 569$ packets 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 $(1431 - 300/2) = 1281$ packets where $|D_1| = 1431$, $T^{\text{disc}} = 300$ ms, $B = 1$ packet/ms and $H^{\text{ave}} = 2$.

The multipath algorithm distributes the remaining volume of data packets from source to destination into the next few iterations in the same way as described

above. The algorithm terminates when $|D| = |D_1| + |D_2| + |D_3| + \cdots$.

3.5. Solution using Lagrangean relaxation and subgradient heuristics

In this section we develop a heuristic solution to the (nonlinear) optimization formulation of our multipath algorithm. (Note that a decision analog of this formulation is an NP-hard problem.) The techniques used are Lagrangean relaxation [7] and subgradient heuristics [9,15].

The Lagrangean relaxation (L) of the optimization problem (Z) in Section 3.1 is obtained by multiplying constraints (2), (4) and (5) with vectors of Lagrangean multipliers λ_j, μ_j and β_j, respectively, and then adding them to the objective function. Thus,

$$
\begin{aligned}
L = & \left(\sum_{i \in [1,\infty)} \sum_{j \in P_i} (r_{ij} + \Delta_{ij} H_{ij} \tau_{ij} p_{ij}) \right)^{\alpha} \\
& + \lambda_j \left(\sum_{i \in [1,\infty)} \sum_{j \in P_i} r_{ij} + \Delta_{ij} H_{ij} \tau_{ij} p_{ij} \right) \\
& - \sum_{i \in [1,\infty)} \sum_{j \in P_i} (r^i_{stab(j)} + |D_i| H^i_{stab(j)} \tau^i_{stab(j)} * p_{ij})) \\
& + \mu_j \left(\sum_{j \in P_i} \Delta_{ij} - |D_i| \right) + \beta_j \left(\sum_{j \in P_i} p_{ij} - |P_i| \right)
\end{aligned}
$$

subject to constraint (3).

The set of feasible solutions for the Lagrangean relaxation, L, of the problem (Eq. (1)) is a super set of the set of feasible solutions for the problem. For the given vectors $\Gamma_j = \{\lambda_j, \mu_j, \beta_j\}$, if the problem has a feasible solution Δ_j and p_j for a given set P_i, then the following relationship holds:

$$L(\Delta_j, p_j, \Gamma_j) < Z(\Delta_j, p_j, \Gamma_j). \tag{8}$$

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

$$L(\Delta^*_j, p^*_j, \Gamma_j) < L(\Delta^*_j, p^*_j, \Gamma^*_j) < L(\Delta_j, p_j, \Gamma^*_j). \tag{9}$$

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

$$L(\Delta^*_j, p^*_j, \Gamma_j) < L(\Delta^*_j, p^*_j, \Gamma^*_j). \tag{10}$$

The solution of Lagrangean relaxation can be obtained by solving $\nabla \Delta_j L = 0$ and $\nabla p_j L = 0$, which computes Δ_j and p_j, respectively. A detailed description is given in [15]. The subgradient heuristic technique is outlined in the appendix. A flowchart is shown in Fig. 2.

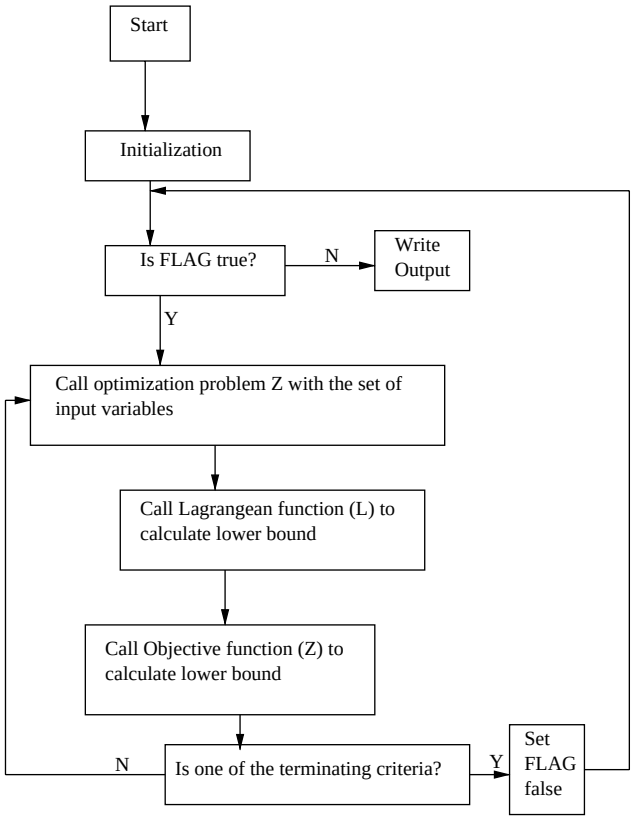

Fig. 2. Flow chart of subgradient algorithm.

4. Simulation results

We note that existing simulators for wireless mobile networks are by and large either wireless link simulators to model wireless link characteristics, or network simulators to study algorithms and protocols in a static setting. Some efforts have recently been made to combine these two classes of simulators [16,17]. They are not inadequate to model and study multihop, ad hoc wireless networks. Although a recent release [2] of the network simulator, *ns*, provides support for modeling wireless LANs, it cannot be used directly for our purpose.

In order to study the performance of our proposed framework, we have developed a simulator [12] with the capability to model and study the following characteristics: (i) node mobility, (ii) link stability (affinity), (iii) affinity-based path search, (iv) pre-emptive route discovery with multipath routing, (v) dynamic network topology depending on mobility and transmission range, and (vi) physical and data link layers in wireless environment.

The proposed framework has been evaluated under a variety of conditions. The simulated environment is assumed to be a closed area of 1000×1000 m^2 in which the mobile nodes are distributed randomly. We have run simulations for networks with 10, 20, 30 and 40 mobile

hosts, operating at transmission ranges (R) varying from 150 to 400 m. The bandwidth (B) for transmitting data is assumed to be 1000 packets/s. 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 1 m (simulated). For example, if the bandwidth is assumed to be 1000 packets/s 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 ms) 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 to 20 m/s. 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 the worst-case scenario and a uniform node velocity of 10 m/s.

4.1. Unipath routing scheme

This section presents some observations on unipath routing scheme using a shortest path and a stable path algorithm. A shortest path algorithm captures the behavior of the general class of routing algorithms in the context of ad hoc networks. Whereas a 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 successful data communication 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 values $N = 4$, $R = 6$, $M = 3$ and $D = 3$ give rise to $(4 \times 6 \times 3 \times 3) = 216$ time steps. With each such step, we have studied shortest- and stable-path algorithms with 10 communication events per minute (C), initiated in the simulated environment with source and destination selected randomly. Thus, a total of 2160 communication events in the 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 the communication efficiency for both shortest-path and stable path algorithms, as shown in Table 1, and observe that the communication efficiency is much higher in the stable-path algorithm. The advantage is more pronounced in case of higher data volumes. As an example, the number of route error generated in shortest-path algorithm with high data volume (3000 packets) is about 32%.

However, it is also noted that the number of successful route discoveries is much less in the stable-path algorithm as compared to the shortest path algorithm. This is due to the fact that the stable-path algorithm evaluates the path for sufficient stability before initiating data communication. For instance, the possibility of finding 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).

These results indicate the effectiveness of stable-path algorithm in terms of reducing the route errors. However, we cannot send a large volume of data at a stretch in the unipath routing scheme. In fact, in order to get uninterrupted connectivity between s and d, we need to use a multipath scheme, as discussed below.

4.2. Proposed multipath 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 mobility. In other words, the intermediate nodes through which s and d are connected, may change with time but the (s–d) path should remain connected. The *connectivity efficiency* has been defined as the ratio of the total number of

Table 1
Communication efficiency for unipath routing

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
3000	720	389	266	68.4	720	214	210	98.1

connected node pairs (in single 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 in which all node-pairs are always connected in single or multiple hops, has an average connectivity efficiency of 100%. Based on our observation shown in Fig. 3, we have selected a transmission range of $R = 300$ meters for $N = 30$ nodes in the following analysis to get 100% average connectivity efficiency.

The transmission range cannot be allowed to increase further due to other two overheads. First, the cost (power consumption due to battery usage) increases as the transmission range is increased. Second, congestion and collision of control packets are the inevitable outcome of higher transmission range during data communication. Fig. 4 shows the variation of the average number of control packets generated per communication with transmission range for $max_{hop} = 4$. For $N = 30$, the control packets grows drastically beyond $R = 300$. In Fig. 5, we have shown the stability of the most stable path between two arbitrary source–destination (14 and 5) pair as depicted in Fig. 1, sampled at every 5 s interval of time. At each 5 s, a route discovery process is initiated from node 14 and the paths obtained after route-request-time-out (300 ms) is evaluated to compute the most stable path.

As shown in Fig. 5, no single path is stable throughout the span of 30 s. However, a sustained stability is obtained between nodes 14 and 5 through different intermediate nodes. This establishes the viability of our scheme. In other words, if we can perform preemptive

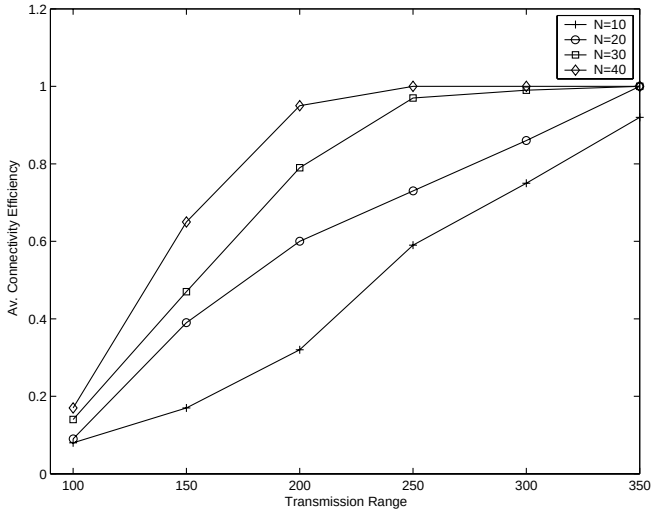

Fig. 3. Average connectivity efficiency vs. transmission range for different number of nodes.

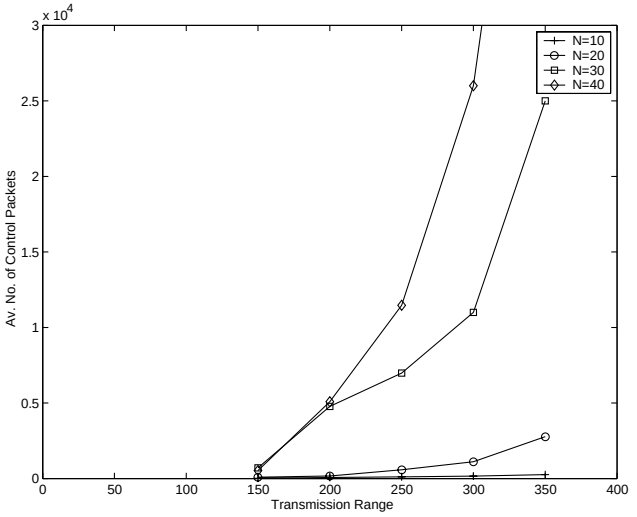

Fig. 4. Average number of control packets generated per communication vs. transmission range.

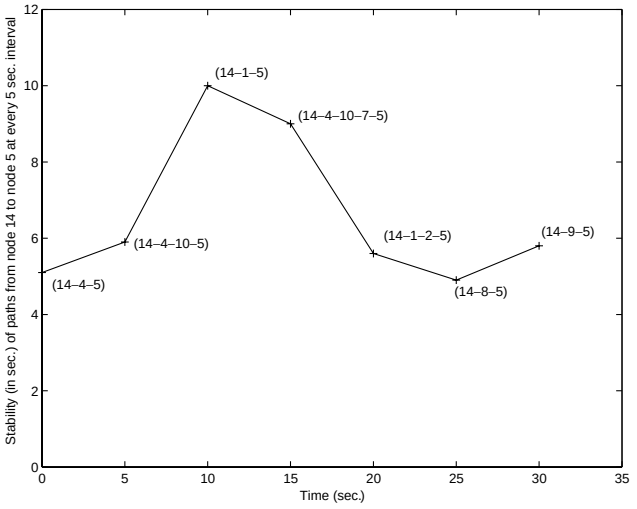

Fig. 5. Stability of maximum stable path between nodes 14 and 5, sampled at each 5-s interval of time (the path is given in the boxes).

Table 2
Total time required for sending 10,000 data packets using temporal multipaths only ($|D| = 10,000$)

P_i $\{r_j, S_j, p_j\}$	$\|D_i\|$	Time (in ms) required for sending $\|D_i\|$ packets of data (assuming $\tau_j = 1$ ms/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	$D = \sum \|D_i\| = 10,000$ packets	30, 238 ms

Table 3
Improvement in total time required for sending 10,000 data packets using both spatial and temporal multipaths ($|D| = 10,000$)

$P_i\{d_j, S_j, p_j\}$			Λ_j			$\lvert D_i \rvert$	Time for sending D_i pkts of 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	11,673
117, 2409, 24–25–5}	251, 4388, 24–32–39–5}	×	1145	615	×	1760	4135
238, 2845, 24–25–5}	×	×	1303	×	×	1303	2606
289, 3200, 24–26–9–5}	×	×	900	×	×	900	2700
Total						$\lvert D \rvert = \sum \lvert D_i \rvert =$ 10,000 pkts	21,114 ms

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 to illustrate the advantage of using temporal multipath only, disregarding the spatial multipath for the time being. The set P_i in this case consists of only one path, which is the most stable path. The total data volume of $D = 10,000$ packets from source (node 24) to destination (node 5) in a 30-node network with an average mobility of 10 m/s, 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 6 times at different intervals to find out dynamically a series of multiple paths in the temporal domain to complete the data transfer. Each route given in the table is the most stable one at that instant of time.

Table 3 shows the same example using both spatial and temporal multipaths. It shows significant improvement over the first scheme that uses temporal multipath alone. First, the number of route discovery required to complete the data transfer process has been reduced to four (from six in the earlier case), which implies creation of less congestion due to control packet propagation. Second, the time required to complete the data transfer is now 21,114 ms, which is significantly less than that without spatial multipath (30,283 ms). Therefore, the average delay per packet decreases from 3 to 2.1 s. This illustrates the efficacy of our proposed scheme in reducing end-to-end delay while transmitting a large volume of data. It is worth noting that the addition of spatial multipath (Table 3) eliminates the longest path (viz., 24–31–37–33–5 in the fourth 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 the paths used, the lesser is the chance of blocking perceived by the end-users.

5. Conclusions

In this paper, we have introduced a notion of temporal and spatial multipath routing in ad hoc wireless networks and described an adaptive framework to evaluate the suitability of using spatial multiple paths with an objective to minimize the end-to-end delay. We have observed that (i) use of temporal multipaths allows a source to transmit a large volume of data to a destination without much degradation of performance due to route-errors, and (ii) use of spatial multipaths helps reduce route-rediscovery and end-to-end delay significantly. The average per-hop delay per packet (τ_j) has been assumed to be 1 ms/packet, which might increase because of different load situation at different nodes. Currently, we are simulating the multipath algorithm under different load situations to see the effect on end-to-end delay. However, even if τ_j is more, it is not expected to affect the effectiveness of the proposed scheme.

Acknowledgments

We thank the anonymous referees for valuable comments which helped us improve the quality of the paper. Thanks are also due to the Guest Editor of this special issue.

The research of S. K. Das was partially supported by Nortel Networks and NSF Grants EIA-0086260 and EIA-0115885. The work of D. Saha was partially supported by grants from AICTE and UGC, Govt. of India.

Appendix. Subgradient heuristic technique

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 this approach, a

gradient method is adapted by replacing the gradients with subgradients. If an initial multiplier vector Γ_j^0 is given, a sequence of multipliers is generated using the following expressions:

$$\lambda_j^{m+1} = \lambda_j^m + t_m \left(\sum_{i\in[1,\infty)} \sum_{j\in P_i} (r_{ij} + \Delta_{ij} H_{ij} \tau_{ij} p_{ij}) - \sum_{i\in[1,\infty)} \sum_{j\in P_i} (r^i_{stab(j)} + |D_i| H^i_{stab(j)} \tau^i_{stab(j)} p_{ij}) \right),$$

$$\mu_j^{m+1} = \mu_j^m + t_m \left(\sum_{j\in P_i} \Delta_{ij} - |D_i| \right),$$

$$\eta_j^{m+1} = \eta_j^m + t_m \left(\sum_{j\in P_i} p_{ij} - |P_i| \right),$$

where (r_{ij}, p_{ij}) are obtained from an optimal solution to Lagrangean relaxation L. The positive scalar, t_m, is a stepsize and given as

$$t_m = \psi_m (Z - L) \Bigg/ \Bigg[\Bigg\{ \left(\sum_{i\in[1,\infty)} \sum_{j\in P_i} (r_{ij} + \Delta_{ij} H_{ij} \tau_{ij} p_{ij}) - \sum_{i\in[1,\infty)} \sum_{j\in P_i} (r^i_{stab(j)} + |D_i| H^i_{stab(j)} \tau^i_{stab(j)} p_{ij}) \right)^2 + \left(\sum_{j\in_i} \Delta_{ij} - |D_i| \right)^2 + \left(\sum_{j\in_i} p_{ij} - |P_i| \right)^2 \Bigg]^{1/2},$$

where ψ_m is a scalar satisfying $0 < \psi_m < 2$. Initially, this scalar is set equal to 2, and 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 below.

Algorithm of subgradient optimization problem

Step 0: *Initialization*
set Z to an arbitrary large value;
select an initial set of multipliers λ_j, μ_j and β_j;
initialize the iteration counter n to 0;
set improvement counter $\mathscr{C}$ to 0;
set λ_j^*, μ_j^* and η_j^* to λ_j, μ_j and β_j respectively;
set the current best value of $L(\lambda_j$, μ_j and $\eta_j)$ to 0;
set stepsize ψ_m to ψ_m^0;

Step $k(k \geqslant 1)$: *Solving the Lagrangean Relaxation.*
(k.1) increment the improvement counter $\mathscr{C} \leftarrow \mathscr{C} + 1$;
(k.2) find out (Δ_j, p_j) by solving $L(\lambda_j$, μ_j and $\beta_j)$;
(k.3) Updating the parameters;

(k.3.1) If $L(\lambda_j^n$, μ_j^n and $\eta_j^n)$ is greater than the current best value of $L(\lambda_j, \mu_j, \beta_j)$, then $L(\lambda_j, \mu_j, \beta_j)$ is replaced by $L(\lambda_j^n, \mu_j^n, \beta_j^n)$. Also set λ_j^*, μ_j^* and β_j^* to λ_j^n, μ_j^n and β_j^n, respectively, where n is the iteration number.

(k.3.2) If (Δ_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 the current best value of Z, then Z is set to this value.

(k.3.3) If improvement has reached a prespecified upper limit, then set ψ_m by $\psi_m/2$ and the improvement counter $\mathscr{C}$ is set to 0 and the procedure restarts from Step (k.1).

(k.3.4) If iteration counter $>$ a prespecified limit
or, if $\psi_m < a$ prespecified limit
or, if $t_m < a$ prespecified limit
or, if $Z - L(\Delta_j, p_j, \Gamma_j)/||L(\Lambda_j, p_j, \Gamma_j)||^2 <$ a prespecified error tolerance then the process terminates.

(k.4) Upgrading the multipliers.

(k.4.1) Compute the new subgradients as

$$\gamma_m^A = \left(\sum_{j\in P_i} \left(r_{ij} + \Delta_{ij} H_{ij} \tau_{ij} p_{ij} - \sum_{j\in P_i} (r^i_{stab(j)} + |D_i| H^i_{stab(j)} \tau^i_{stab(j)} p_{ij}) \right) \right),$$

$$\gamma_m^B = \left(\sum_{j\in P_i} \Delta_{ij} - |D_i| \right),$$

$$\gamma_m^C = \left(\sum_{j\in P_i} \lceil p_j \rceil - |P_i| \right).$$

(k.4.2) Compute the stepsize as

$$t_m = \psi_m (Z - L) \Bigg/ \Bigg[\Bigg\{ \left(\sum_{j\in P_i} (r_{ij} + \Delta_{ij} H_{ij} \tau_{ij} p_{ij}) - \sum_{j\in P_i} (r^i_{stab(j)} + |D_i| H^i_{stab(j)} \tau^i_{stab(j)} p_{ij}) \right)^2 + \left(\sum_{j\in P_i} \Delta_{ij} - |D_i| \right)^2 + \left(\sum_{j\in P_i} p_j - |P_i| \right)^2 \Bigg]^{1/2},$$

(k.4.3) Compute the new multipliers as

$$\lambda_{i,m+1} = \min(-1, \lambda_{i,m} + t_m \gamma_m^A),$$
$$\mu_{i,m+1} = \min(-1, \mu_{i,m} + t_m \gamma_m^B),$$
$$\beta_{i,m+1} = \min(-1, \beta_{i,m} + t_m \gamma_m^C).$$

(k.4.4) Increment iteration counter by 1.
(k.5) go to Step (k.1).

References

[1] S. Bahl, M. El-Zarki, Dynamic multi-path routing and how it compares with other dynamic routing algorithms for high speed wide-area networks, Proceedings of the ACM SIGCOM, 1992.

[2] J. Broch, D.A. Maltz, D.B. Johnson, Y. Hu, J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing protocols, Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking, Mobicom'98, Dallas, Texas, October 25–30, 1998.

[3] H. Chu, K. Nahrstedt, Dynamic multi-path communication for video traffic, Proceedings of the Hawaiian International Conference on System Science, HICSS, 1997.

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

[5] 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, Proceeding of IEEE GLOBECOM, San Francisco, November 2000, pp. 261–265.

[6] R. Dube, C.D. Rais, K. Wang, 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, August 1996.

[7] M.L. Fisher, An applications oriented guide to Langrangean relaxation, Interfaces 15 (1985) 10–21.

[8] Z.J. Haas, A new routing protocol for the reconfigurable wireless networks, Proceedings of ICUPC'97, San Diego, October 1997.

[9] H. Held, P. Wolfe, H.P. Crowder, Validation of subgradient optimization, Math. Program 5 (1974) 62–68.

[10] D.B. Johnson, D. Maltz, Dynamic source routing in ad hoc wireless networks, in: T. Imielinski, H. Korth (Eds.), Mobile Computing, Kluwer Academic Publishers, Dordrecht, 1996.

[11] S. Murthy, J.J. Garcia-Luna-Aceves, Congestion-oriented shortest multi-path routing, Proceedings of the IEEE INFOCOM, 1996.

[12] K. Paul, S. Bandyopadhyay, A. Mukherjee, D. Saha, A stability-based distributed routing mechanism to support unicast and multicast routing in ad hoc wireless network, Comput. Commun. 24 (December 2001) 1828–1845.

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

[14] N.S.V. Rao, S.G. Batsell, QoS Routing via multiple paths using bandwidth reservation, Proceedings of the IEEE INFOCOM, 1998.

[15] D. Saha, A. Mukherjee, On the multidensity gateway location problem for a multilevel high speed network, Comput. Commun. 20 (1997) 576–587.

[16] J. Short, R. Bagrodia, L. Kleinrock, Mobile wireless network system simulation, Wireless Networks 1 (4) (1995) 451–967.

[17] M. Srivastava, P. Mishra, P. Agrawal, G. Nguyen, Ethersim: a simulator for application-level performance modeling of wireless and mobile ATM networks, Comput. Networks ISDN Systems 29 (1998) 2067–2090.

[18] C.-K. Toh, A novel distributed routing protocol to support ad-hoc mobile computing, IEEE International Phoenix Conference on Computer and Communications (IPCCC), 1996.

Sajal K. Das is a Professor of computer science and engineering and also the Founding Director of the Center for Research in Wireless Mobility and Networking (CReWMaN) at the University of Texas at Arlington. A recipient of numerous awards for best teaching and scholarly research, he has visited numerous universities, research organizations, government and industry research labs worldwide for collaborative research and delivering invited seminar talks. He is also frequently invited as a speaker at international conferences and symposia. His current research interests include resource and mobility management in wireless networks, mobile and pervasive computing, wireless multimedia and QoS provisioning, mobile Internet architectures and protocols, and grid computing. He has published over 200 research papers in these areas, directed numerous funded projects, and holds four US patents in wireless mobile networks. He received the Best Paper Awards in ACM MobiCom'99, ICOIN-2002, ACM MSWiM 2000, and ACM/IEEE PADS'97. He serves on the Editorial Boards of IEEE Transactions on Mobile Computing, ACM Wireless Networks, Computer Networks, Journal of Parallel and Distributed Computing, Parallel Processing Letters, Journal of Parallel Algorithms and Applications. He served as General Chair of IEEE MASCOTS-2002; General Vice Chair of IEEE PerCom-2003, ACM MobiCom-2000 and HiPC-2001; General Chair of ACM WoWMoM 2000-2002, Program Chair of WoWMoM'98-99, TPC Vice Chair of ICPADS-2002; and as TPC member of numerous IEEE and ACM conferences. He is a member of the IEEE TCPP Executive Committee and Advisory Boards of several cutting-edge companies.

Amitava Mukherjee received his Ph.D. degree in Computer Science from Jadavpur University, Calcutta, India. He was in the Department of Electronics and Telecommunication Engineering at Jadavpur University, Calcutta, India from 1983 to 1995. Since June 1995, he is a Principal Consultant in erstwhile PwCC India, which is now IBM Business Consulting Services, part of IBM Global Services India Pvt. Ltd. from October 2002. From January 2003, he has joined the School of CSE, UNSW Sydney in a Faculty position on a sabbatical from IBM Global Services, Calcutta, India. His research interests are in the areas of Mobile Computing and Communication, Pervasive Computing and Mobile Commerce, Optical Networks, Combinatorial Optimization and Distributed Systems. His interests also include the Mathematical Modeling and its applications in the fields of Societal Engineering and International Relations. He is the author of over 75 technical papers, one monograph and four books. He is a member of IEEE, IEEE Communication Society.

Somprakash Bandyopadhyay is currently an Associate Professor in MIS and computer science group at the Indian Institute of Management, Calcutta. He has more than 20 years of academic and industrial experience in PricewaterhouseCoopers Limited, Indian Institute of Technology at Kharagpur, Indian Institute of Technology at Bombay, Tata Institute of Fundamental Research, and Jadavpur University, Calcutta. He is a Ph.D. in computer science from Jadavpur University (1985) and B.Tech in electronics and electrical communication engineering from Indian Institute of Technology, Kharagpur (1979). He was a research fellow of the Japan Trust International Foundation and worked in Advanced Telecommunication Research Institute in Kyoto, Japan in 2001 in the area of Ad Hoc Wireless Networks. He was also a fellow of the Alexander von Humboldt Foundation, Germany, and was involved in post-doctoral research at the German Research Centre for Artificial Intelligence, Saarbrucken during 1989–90.

Debashis Saha received his Bachelor of Electronics & Telecommunication Engineering degree in 1986 from Jadavpur University, Calcutta, and his M.Tech. and Ph.D. in 1988 and 1996, respectively, from Indian Institute of Technology (IIT) at Kharagpur. During 1990–2001, he was a faculty of Computer Science and Engineering at Jadavpur University. Currently, he is an Associate Professor in the Indian Institute of Management (IIM) Calcutta. His research areas are pervasive computing, wireless/mobile communication and Computing, and WDM optical networks. He has published more than 100 papers

in various conferences and journals. He has co-authored five books and a monograph. He is a recipient of the prestigious Career Award for Young Teachers (1997) from the AICTE, Govt. of India, and is a SERC Visiting Fellow (1999) and a BOYSCAST Fellow (2000) of Dept. of Science and Technology (DST), Govt. of India.

Krishna Paul received her MCA in 1993 and Ph.D. in 2001, both from Jadavpur University, Calcutta. Currently, she is an Assistant Professor in the School of Information Technology at Indian Institute of Technology, Bombay. She was a Senior Research Associate at Indian Institute of Technology, Kharagpur, for four years. She worked as Software Engineer in Techna Digital Systems, India for 2 years and with Cognizant Technology Solutions, India as Senior Associate for one and half years in the area of E-Commerce and Web Technology. She was also a Research Staff Member in Mobile Internet group of Computer and Communication Laboratory, NEC Europe Ltd. during 2001–2002. Her research interests include mobile and wireless computing.