Shortest Multipath Routing Using Labeled Distances *

Chandramouli Balasubramanian J.J. Garcia-Luna-Aceves
Department of Computer Engineering
University of California
Santa Cruz, CA 95064, U.S.A.
Email: {chandrab, jj}@cse.ucsc.edu

Abstract

We present and verify SMLDR (Shortest Multipath Labeled Distance Routing), an on-demand loop free multipath routing protocol. It extends Labeled Distance Routing (LDR) to the multipath domain and enables loop freedom by maintaining the ordering of distance invariants. By modifying the route update conditions of LDR and by using the concept of limiting distance we demonstrate shortest multipath routing. Further we describe the fundamental multipath concepts for on-demand routing protocols and elucidate how SMLDR exercises each of these concepts in its routing mechanisms. The performance of SMLDR is compared against the performance of LDR, AODV and its multipath variant AOMDV. The simulation results corroborate the need for shortest multipath routing in terms of higher performance for the chosen metrics.

1. Introduction

On-demand routing protocols were designed to address the constraints of mobile ad hoc networks [3]. These protocols maintain routes to active destinations discovered on a need-to-know basis by broadcasting a source-initiated query request. In any network, there may be more than one route to the destination. Single path routing protocols record only the most feasible (primary) path that was discovered earliest. Some on-demand single path routing protocols that have been proposed include Ad hoc On-demand Distance Vector (AODV) routing [16], Dynamic Source Routing (DSR) [9] and Labeled Distance Routing (LDR) [7].

Multipath routing protocols work on the principle that higher performance can be achieved by recording more than one feasible path. Multipath routing in wired networks has been proposed to take advantage of network redundancy, reduce congestion, and address QoS issues. Lower delay, increased fault tolerance [21], lower power consumption [5], and higher security [11] are other compelling reasons that exist for discovering multiple paths in MANETs. Node mobility in ad hoc networks leads to frequent link breaks. This induces periodic route request broadcasts, resulting in both a higher routing overhead and route establishment delay. With both data and signaling packets competing for the same channel packet delivery is substantially reduced. However, when multiple routes are known, even if the primary path fails data forwarding can continue uninterrupted on the alternate available paths without waiting for a new route to be discovered.

Many on-demand multipath routing protocols have been proposed for ad hoc networks, including Split Multipath Routing (SMR) [10], Multipath Dynamic Source Routing (Multipath DSR) [14], Temporally Ordered Routing Algorithm (TORA) [15], Routing On-demand Acyclic Multipath (ROAM) [17], Ad hoc On-demand Multipath Distance Vector (AOMDV) [12] and Cooperative Packet Caching and Shortest Multipath (CHAMP) [18]. SMR and multipath DSR and are based on source routing while TORA, ROAM, AOMDV and CHAMP are distance-vector based.

Disjoint paths in SMR are determined on the basis of path information at the destination. The destination replies to the first request and waits until other requests have been received. It then chooses a maximally disjoint path from the one that has already been replied to and initiates a reply. The data traffic is split between the two available paths. Multipath DSR also extends DSR to incorporate multipath routing. Difference is that in the former approach the data traffic is split among the available paths, while the latter adopts an alternate path routing approach. In multipath DSR, intermediate nodes are equipped with multipath to prevent in-flight data from getting dropped.

TORA provides multiple alternate paths by maintaining a destination oriented directed acyclic graph (DAG) from the source. ROAM extends DUAL [6] to create routes on

* This work was funded in part by the Baskin Chair of Computer Engineering at UCSC.

0-7803-8815-1/04/$20.00 ©2004 IEEE

demand and maintain multiple loop free paths per destination. When links fail it initiates the diffusing computations to synchronize and update the upstream nodes of the new routing information. Both TORA and ROAM require reliable delivery of control packets. When a node is involved in a computation the routes are locked down until it has received replies from all its neighbors. The control overhead incurred by such mechanisms is not viable beyond low mobility.

AOMDV is a multipath mechanism based on AODV. To achieve multipath AOMDV accepts multiple reverse route requests and maintains a multipath table for each destination. A node advertises the hop count that is greater than all known distances at that node and hence maintains AODV's distance invariants. Though this maintains loop freedom the advertisements subsume the shorter routes causing them to become indisposed and unusable.

DASM (Diffusing Algorithm for Shortest Multipath) [22] is a multipath algorithm for wired networks that introduced the concept of shortest multipath as "a directed acyclic graph defined by the successor entries of the routing tables of routers in all the paths from the source to a destination that are guaranteed to be loop-free at any given instant." MDVA [20] is a proactive multipath distance vector routing protocol that considers the granularity of link costs to discover shortest multipath. By load-balancing traffic over the multiple successors discovered minimal delays are achieved [19]. CHAMP is similar to MDVA and uses a simple load balancing approach to route packets. However, CHAMP does not discover shortest multipath but merely considers shortest equal cost multipath routing based on paths of equal length. Cooperative packet caching and rerouting of data packets are used to improve packet delivery.

SMR, Multipath DSR and AOMDV employ disjoint path discovery mechanisms to provide for independent route failures. SMR and multipath DSR are source routing protocols and therefore use path information to determine disjoint paths. AOMDV uses the last hop as a path identifier to obtain disjoint paths. However, in the presence of route failures, an intermediate node that changes successors may not relay the new information to its predecessors. This causes an inconsistency in the state of the last hop maintained at the upstream nodes and link disjointness cannot always be guaranteed.

Section 2 identifies the fundamental concepts that represent the design choices in multipath routing. The above described multipath protocols are all tailored for either link costs or hop count as the distance metric. Section 3 presents shortest multipath labeled distance routing (SMLDR), which is a generalized framework for shortest multipath routing. We discuss the operation of SMLDR and illustrate the mechanisms used for attaining shortest multipaths. Section 4 presents the analysis and shows that SMLDR works correctly and is loop-free. Section 5 discusses simulation results of SMLDR and compares its performance with AODV, LDR and AOMDV. Simulation results clearly indicate the need for shortest multipath routing and show that SMLDR outperforms the other protocols. Section 6 concludes our work.

2. Fundamental Concepts of Multipath Routing

We identify the following seven concepts as fundamental to multipath routing algorithms. Although all seven need not be present in the routing schema, they ought to be considered in any multipath design. These abstractions can be used as building blocks for designing a new multipath protocol or as features of comparison among existing multipath routing protocols.

Multiple Route Discovery Procedure is the process by which multiple paths are discovered. This is similar to the route discovery mechanism used in single path routing protocols viz. route discovery flood with the route replies backtracking to the source along the reverse routes established by the requests. However the nodes now treat each request received from distinct previous hops as potential multipaths. Similarly the destination may initiate a reply for each request received from distinct neighbors.

Filtering Provision is the option of choosing certain paths with higher utility value against choosing all the paths that become available. Some filtering provisions are i) feasible loop free paths, ii) shortest multipaths, iii) disjoint paths (SMR, multipath DSR, AOMDV), and iv) threshold on number of paths that are recorded in the routing table. In dense networks often a combination of one or more provisions provides the most effective route pruning.

Path Usage Policy describes whether all the paths (or a subset of them) would be used at once or one path at a time. The former requires data to be forwarded along all the paths (SMR, multipath DSR, and CHAMP). This lends itself naturally to load balancing and traffic engineering approaches. The latter forwards data only along the primary (potentially least cost) path and when the primary path fails alternate paths are employed.

Data Forwarding Mechanism refers to the way in which the data is to be forwarded over the multiple paths. This property is meaningful only when all the paths (or a subset of them) are used at the same time. Examples of a few schemes include the simple round robin (CHAMP) and heuristic forwarding based on path lengths (MDVA).

Multipath Maintenance Heuristic refers to how the multiple paths would be maintained. For example, if number of available paths goes below a threshold, initiating a new

route discovery ensures continuous availability of multiple paths to the destination.

Data Path Freshness Strategy answers the question "what mechanisms ensure the freshness of the data paths?" The freshness maintenance strategy is closely tied to the path usage policy. When all the paths are simultaneously used the data packets flowing along these paths automatically update the lifetime (CHAMP). When the paths are used one at a time the primary path's lifetime always gets updated but to keep the alternate paths alive hello packets are needed for link sensing by probing the path (AOMDV).

Underlying Single Path Routing Protocol comes into play when the multipath routing protocol being designed is an extension of a single path routing protocol. In certain cases the peculiarities of the single path protocol can affect the multipath mechanisms.

3. Shortest Multipath Labeled Distance Routing (SMLDR)

SMLDR discovers multiple loop free paths to the destination and employs LDR as the underlying single path routing protocol. LDR uses the notion of feasible distances to test the feasibility of a route. Each node in a path tests independently if the reported distance in the advertisement is lesser than its feasible distance in order to accept the advertisement. Further in LDR the feasible distance is always reset to the minimum distance. By employing a slightly relaxed policy on the feasible distance reset we allow for multiple paths to be accrued to the destination.

SMLDR differs from LDR in the route reset mechanism as well. In LDR when the route invariants cannot be satisfied the node initiates a unicast probe if it has a feasible path to the destination. We have modified this to be a broadcast mechanism. This change is introduced because it is infeasible to unicast along all the available paths and more paths can be explored by broadcasting the request.

In SMLDR a new distance metric termed *limiting distance* is introduced which is the minimum distance to the destination known at each node in the network. Using this concept provides the filtering mechanism to select shortest multipath for data forwarding. SMLDR uses alternate path routing and does not maintain disjoint paths. The routing table entries are ordered on the basis of the limiting distance to avail the shorter paths. Basic operation of SMLDR described in this paper does not use any multipath maintenance heuristic. Hellos or keep-alive packets are required to maintain the data path freshness of alternate paths.

Table 1 describes the notations used throughout this paper. The loop-free conditions and the operational procedures of SMLDR are outlined next.

Notation	Meaning
sn_D^A	Sequence number for destination D as known at node A.
lc_B^A	Link cost from node A to node B.
d_{DB}^A	Distance of node A to destination D via next hop B.
nh_D^A	Next hop for node A towards destination D.
ph_S^A	Previous hop for node A towards source S.
fd_D^A	Feasible distance for destination D as known to node A.
rd_D^A	Reported Distance for destination D advertised by node A.
ld_D^A	Limiting Distance for destination D advertised by node A.
S_D^A	Successor set at node A towards destination D.
rr_D^{rq}	Route reset bit for Destination D in the route request.
$rpld_{\{S,\,ph,\,rreqid\}}$	"replied" bit for tuple {S, ph, rreqid} in RREQ cache.
$bM_{\{S,\,ph,\,rreqid\}}$	"black mark" bit for tuple {S, ph, rreqid} in RREQ cache.

Table 1. Notations

3.1. Design

During route discovery the route request (RREQ) broadcast by the source is the tuple $\{dst,\ sn_{dst},\ rreqid,\ src,\ sn_{src},\ ph,\ fd_{dst},\ rd_{src},\ ld_{src},\ flags\}$. The field $rreqid$ is the unique broadcast identifier generated by the source src for each request initiated. sn_{dst} is the sequence number and fd_{dst} is the feasible distance for the destination with identifier dst. ph identifies the previous hop visited by the route request. The source sequence number sn_{src}, the reported distance rd_{src}, and the limiting distance ld_{src} are fields that help to establish the feasible routes back to the source if need be. The route reply (RREP) is the tuple $\{dst,\ sn_{dst},\ src,\ rreqid,\ rd_{dst},\ ld_{dst},\ lifetime,\ flags\}$ generated either by the destination or an intermediate node. The $rreqid$ is the value copied from the route request. rd_{dst} is the distance reported by the node for the destination and ld_{dst} is the corresponding limiting distance. The $lifetime$ field indicates the remaining time for the route to the destination and is the upper threshold for the timeout in the routing table. Flags contain the control bits. We assume symmetric link costs.

Each node upon receiving a request records the tuple $\{src,\ rreqid,\ ph,\ rpld,\ bM\}$ in the RREQ cache. If the route request is not a feasible route to the source then the black mark (bM) bit is set to $true$. Recording distinct previous hops helps to identify the multiple paths back to the

source along which the route replies can be relayed. Once a reply is relayed the replied ($rpld$) bit in the route request cache is set to $true$. Replies are relayed only to those previous hops whose entry in the request cache has bM bit set to $false$. This is necessary to avoid circular relaying of route replies.

If a node A has routes to destination *D* it maintains in its routing table the sequence number originated by *D* (sn_D^A), the feasible distance to *D* (fd_D^A), the reported distance for *D* (rd_D^A), the limiting distance to *D* (ld_D^A) and a route list. The entries in the route list contain the set of next hops which form the successor set towards *D* and the distance to D via these next hops. The feasible distance fd_D^A is the first instance of distance to *D* recorded by *A* for the current sequence number. The limiting distance ld_D^A is the minimal distance and the reported distance rd_D^A is the maximum distance of the valid entries in the route list for destination D.

3.2. Loop-Free Invariant Conditions

The loop freedom conditions in SMLDR flow directly from LDR which is the underlying single path routing protocol. To provide routing table loop freedom LDR uses a combination of distance invariants and sequence numbers. Higher sequence number indicate a fresher advertisement and is used to reset the invariants. LDR describes three conditions; the numbered distance condition, the feasible distance condition and the start distance condition as sufficiency conditions for loop freedom [7]. As long as these conditions are satisfied it is not possible for a loop to be formed. Start distance condition specifies the requirements to be satisfied by an intermediate node to initiate a route reply. Numbered distance condition must be satisfied for a node to change its successor to the destination and for the route to be considered as a feasible route. Feasible distance condition ensures ordering of the feasible distances along any path to the destination. We restate the conditions modified to incorporate multipath routing.

Start Distance Condition (SDC): A node *I* can initiate a route reply for a request from originating node *A* for destination *D* if *I* has a valid and fresher route to *D*. The validity and freshness are determined if one of the following conditions are satisfied.

$sn_D^I > sn_D^{rq}$
$sn_D^I = sn_D^{rq} \wedge rd_D^I < fd_D^{rq} \wedge \neg rr_D^{rq}$

Numbered Distance Condition (NDC): A node "A" may update its route entry for a destination D upon receiving a route reply from B if one of the following cases is satisfied.

$sn_D^{rp} > sn_D^A$
$sn_D^{rp} = sn_D^A \bigwedge rd_D^{rp} < fd_D^A$

If node A already has valid fresh routes to the destination then $rd_D^{rp} + lc_B^A \leq rd_D^A$ must be satisfied for $sn_D^{rp} = sn_D^A$.

Feasible Distance Condition (FDC): An intermediate node *I* must set the route reset bit when it relays a route request which it cannot satisfy.

$$if\ (sn_D^I < sn_D^{rq}) \bigvee ((sn_D^I = sn_D^{rq}) \bigwedge (fd_D^I \geq fd_D^{rq}))$$
$$then\ rr_D^{rq} \leftarrow 1$$

3.3. Route Discovery and Maintenance

When a source node *A* needs to send data to destination *D* and does not have a valid path to *D*, it starts a timer and relays a route request for destination *D* with route request identifier ID_A. *A* is then said to be active for destination *D* for the computation (A, ID_A). When A receives a feasible reply for the destination it terminates the computation. In between if the timer expires A's computation results in a failure. A then increments ID_A and initiates a new request for the destination *D*.

Procedure 1:Initiate Route Request. A node *A* relaying a route request if active for *D* buffers the data packet. Otherwise it becomes active for *D*, relays a route request identified by (A, ID_A) and starts a timer with expiry time $2 \cdot ttl \cdot latency$ where *ttl* refers to the time-to-live of the broadcast and latency is the estimated per hop latency of the network. For each route request that *A* relays it increments its route request id (ID_A). The feasible distance fd_D^{rq} in the route request is set to the last known feasible distance for destination *D* at *A*. If no reply is received before the timer expires *A* may retry the request with an increased *ttl*. After a predetermined number of attempts if A is not able to find a route to *D* it should inform the upper layer about the inability to deliver and drop packets which have been queued for the destination.

Procedure 2: Relay Route Request. If an intermediate node *I* receives a route request (A, ID_A) from a previous hop *B* (possibly equal to *A)* for destination *D*, *I* first checks to see if it is passive for destination *D*. If it is passive it becomes engaged and records $\{A, ID_A, B\}$ in its route request cache for a sufficient period of time referred to as the *reverse route timeout*. If node I satisfies SDC and has at least one route which has not been used in a reply before *I* should initiate a reply; otherwise the route request is further relayed. If nrq denotes the new route request then

$$sn_D^{nrq} \leftarrow \begin{cases} sn_D^I & if\ sn_D^I > sn_D^{rq} \\ sn_D^{rq} & otherwise \end{cases}$$

$$fd_D^{nrq} \leftarrow \begin{cases} fd_D^I & if\ sn_D^I > sn_D^{rq} \\ min(fd_D^I,\ fd_D^{rq}) & if\ sn_D^I = sn_D^{rq} \\ fd_D^{rq} & otherwise \end{cases}$$

$$rr_D^{nrq} \leftarrow \begin{cases} 0 & if\ sn_D^I > sn_D^{rq} \\ rr_D^{rq} & if fd_D^I < fd_D^{rq} \bigwedge sn_D^I = sn_D^{rq} \\ 1 & otherwise \end{cases}$$

$$rd_S^{nrq} \leftarrow rd_S^I$$
$$ld_S^{nrq} \leftarrow ld_S^I$$

The feasible distance in the new route request is set to the minimum for the same sequence number as the reply must be satisfied at all nodes along the reverse path.

Procedure 3: Initiate Route Reply Destination. The destination *D* must initiate a route reply for each route request that it receives from distinct neighbors. If the route reply is denoted by *rp* then

$$\begin{aligned} sn_D^{rp} &\leftarrow \begin{cases} (sn_D^D + 1) & if\ sn_D^D = sn_D^{rq} \bigwedge rr_D^{rq} \\ sn_D^D & otherwise \end{cases} \\ rd_D^{rp} &\leftarrow 0 \\ ld_D^{rp} &\leftarrow 0 \end{aligned}$$

Procedure 4: Initiate Route Reply Intermediate Node. A node *I* should initiate a route reply if it has a valid route to the destination *D* and it satisfies SDC. The replied bit ensures that the node does not forward each available path more than once per (originator, rreqid) for each previous hop. If *ph* is the previous hop towards the source *S* stored in the route request cache, *nh* is the next hop towards the destination and if *rp* is the reply initiated then

$$\begin{aligned} sn_D^{rp} &\leftarrow sn_D^I \\ rd_D^{rp} &\leftarrow rd_D^I \\ ld_D^{rp} &\leftarrow ld_D^I \\ rpld_{\{S,ph,rreqid\}} &\leftarrow true \end{aligned}$$

Procedure 5: Update/Add Route Entry. Node A accepts a route reply from a node *B* if it satisfies NDC. The route entries are updated as below. As each node maintains only one sequence number per destination, a reply with a higher sequence number purges all the earlier entries.

$$\begin{aligned} sn_D^A &\leftarrow sn_D^{rp} \\ d_{DB}^A &\leftarrow rd_D^B + lc_B^A \\ S_D^A &\leftarrow \begin{cases} B & if\ sn_D^A < sn_D^{rp} \\ S_D^A \bigcup B & if\ sn_D^A = sn_D^{rp} \end{cases} \\ ld_D^A &\leftarrow min(d_{DI}^A)\,\forall I \mid I\,\varepsilon\,S_D^A \\ rd_D^A &\leftarrow max(d_{DI}^A)\,\forall I \mid I\,\varepsilon\,S_D^A \\ fd_D^A &\leftarrow d_{DB}^A\ if\ sn_D^A < sn_D^{rp} \end{aligned}$$

Procedure 6: Relay Route Reply. If a node *A* received a route reply *rp* and is not the originator *S* of the route request then *A* after updating its routing table may relay a new route reply *nrp* of the form

$$\begin{aligned} sn_D^{nrp} &\leftarrow sn_D^A \\ rd_D^{nrp} &\leftarrow rd_D^A \\ ld_D^{nrp} &\leftarrow ld_D^A \\ rpld_{\{S,ph,rreqid\}} &\leftarrow true \end{aligned}$$

where *ph* is the previous hop in the RREQ cache which has neither been replied to nor has the black mark bit set.

Procedure 7: Route Errors. A node *A* marks its route to destination *D* through next hop *B* as invalid if any of the following events occur:

- *A* receives a notification of link failure while sending data packets to *B* for destination *D*.
- *A* receives a route error from *B* for destination *D*.
- *A* does not route any data packets for destination *D* via *B* for a period of *active_route_timeout seconds* and has now received a data packet to be routed to destination *D*.

The successor set and distances at node A are then recomputed as

$$\begin{aligned} S_D^A &\leftarrow S_D^A - B \\ ld_D^A &\leftarrow min(d_{DI}^A)\,\forall I \mid I\,\varepsilon\,S_D^A \\ rd_D^A &\leftarrow max(d_{DI}^A)\,\forall I \mid I\,\varepsilon\,S_D^A \\ fd_D^A &\leftarrow min(fd_D^A,\, rd_D^A) \end{aligned}$$

If *A* does not have any more valid routes to *D* then *A* initiates a route error for destination *D* to all its precursors.

Procedure 8: Route Data Packet. When a node *A* receives a packet for destination *D*, A chooses I as the next hop to forward the data packet if $(ld_D^A = d_{DI}^A)\,\forall I \mid I\,\varepsilon\,S_D^A$.

3.4. Shortest Equal Cost Multipath Routing

By resetting the feasible distance in Procedure 5 and modifying NDC to accept only routes with equal or lesser limiting distances, the above procedures can be readily adapted for shortest equal cost multipath routing.

Modification in Procedure 5:

$$fd_D^A \leftarrow \begin{cases} d_{DB}^A & if\ sn_D^A < sn_D^{rp} \\ min(fd_D^A, d_{DB}^A) & if\ sn_D^A = sn_D^{rp} \end{cases}$$

Modification in NDC:

If node A already has valid fresh routes to the destination then $d_D^{rp} + lc_B^A \leq ld_D^A$ for $sn_D^{rp} = sn_D^A$.

3.5. Example

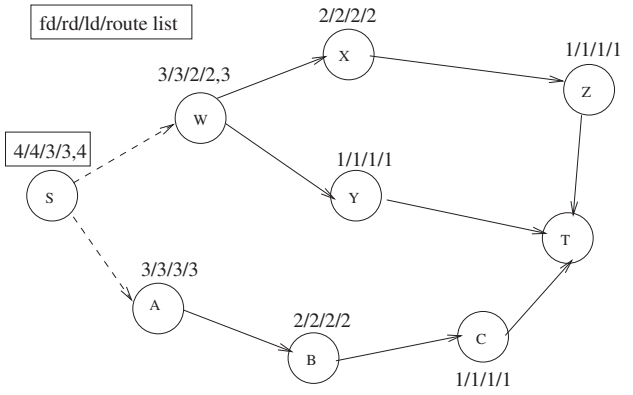


Figure 1. SMLDR Operation

Consider the network shown in Figure 1, initially without node *S*. The numbers stored represent the feasible distance, reported distance, limiting distance and the distances stored in route list entries of the nodes for the destination *T*. From the figure it can be seen that the shortest route from *S* to *T* is via *W*. We now illustrate using the concept of limiting distance how *S* can determine the shortest multipath

to *T*. Let *S* initiate a route request for *T*. Nodes *W* and *A* report distances 3 and 3 respectively. If *S* receives *A*'s reply first it updates its feasible, reported and limiting distances to 4 with *A* as the next hop towards destination *T* in its route list. When *S* receives *W*'s reply the reply is feasible at *S* and *W* is added to its route list. Though there is no change in the feasible and reported distances *S* updates it's limiting distance to 3. As *W* offers a lower limiting distance *S* chooses *W* as its successor to forward data packets in preference to *A*. Only when *S*' route to *T* via *W* fails *S* employs the route via *A* to reach *T*. The utility of limiting distance lies in the ability to discern optimal routes among those obtained during the multipath route discovery phase.

4. Analysis

We first prove loop freedom in SMLDR under the assumption that no node forgets the last sequence number it learns for a given destination. The proof requires that the relaying of replies and reply processing maintains the ordering of invariants. The ordering we want to establish is that the sequence numbers are nondecreasing and for the same sequence numbers the feasible distances are nonincreasing as we move away from the destination i.e. along any successor path *P* = $\{n_k, ..., n_1\}$ for all i ε [2, k] towards destination D, the ordering $(sn_D^{ni} < sn_D^{ni-1}) \vee (sn_D^{ni} = sn_D^{ni-1} \wedge fd_D^{ni} > fd_D^{ni-1})$ is to be maintained. We then prove that each source initiating a request is guaranteed a feasible reply in finite time if the network is stable for a sufficient period of time. Further we show that SMLDR is live.

Lemma 1: If a node updates its routing table by Procedure 5 and initiates replies by Procedures 3 or 4 or relays replies by Procedure 6 then NDC ensures that the ordering criteria is maintained when successor path is established assuming no node changes successor on the path.

Proof: We note that only the destination can increase its own sequence numbers and by Procedure 5 the feasible distance always decreases or remains same in a node for a particular sequence number. Let $P = \{n_k, ..., n_1\}$ be a successor path from the node $n_k$to n_1. The proof is by induction on the number of hops starting with the destination as the node generating the route reply and is similar to the proof in LDR [7, Lemma 1, pp.58].

Lemma 2: Given an established path that obeys the ordering criteria any change of successor (not in the successor list) according to NDC and Procedure 6 maintains the ordering along that path.

Proof: Let time $t^s_{n_i}$ be when node n_i switches successor to node n_{i-1} and let time $t^c_{n_i}$ be when node n_i switches successor from the earlier established path $P = \{n_k, ..., n_1\}$ that obeys the ordering criteria. Let m_j be the node in $\{m_j, ..., m_1, n_1\}$ which is in order to which n_i switches. This means that $sn^{n_i}_{n_1}(t^c_{n_i}) < sn^{m_j}_{n_1}(t^c_{n_i})$ or $sn^{n_i}_{n_1}(t^c_{n_i}) = sn^{m_j}_{n_1}(t^c_{n_i}) \wedge fd^{n_i}_{n_1}(t^c_{n_i}) > fd^{m_j}_{n_1}(t^c_{n_i})$ must be the ordering according to Procedure 6. Now n_i would change to m_j when m_j forwards a reply rp_{m_j} in $[t^s_{n_i}, t^c_{n_i}]$. If n_i accepts rp_{m_j}, it means the reply is feasible at n_i and in accordance with NDC. Thus $sn^{n_i}_{n_1}(t^c_{n_i}) < sn^{m_j}_{n_1}(t^c_{n_i})$ or $sn^{n_i}_{n_1}(t^c_{n_i}) = sn^{m_j}_{n_1}(t^c_{n_i}) \bigwedge fd^{n_i}_{n_1}(t^c_{n_i}) > rd^{m_j}_{n_1}(t^c_{n_i}) \geq fd^{m_j}_{n_1}(t^c_{n_i})$. This shows that Procedure 6 does not violate the ordering criteria.

Lemma 3: If a node A changes successor to one of the nodes in its successor list; NDC and Procedures 5, 6, and 7 maintain the ordering of the invariants.

Proof: Let time $t^s_{n_i}$ be when node n_i switches successor to node n_{i-1} and let time $t^c_{n_i}$ be when node n_i switches successor from the earlier established path $P = \{n_k, ..., n_1\}$ that obeys the ordering criteria. Let m_j be the node in the ordered path $\{m_j, ..., m_1, n_1\}$ to which n_i switches and let $t^o_{n_i}$ be the time when n_i received a reply from m_j. This means that $sn^{n_i}_{n_1}(t^o_{n_i}) < sn^{m_j}_{n_1}(t^o_{n_i})$ or $sn^{n_i}_{n_1}(t^o_{n_i}) = sn^{m_j}_{n_1}(t^o_{n_i}) \bigwedge fd^{n_i}_{n_1}(t^o_{n_i}) > rd^{m_j}_{n_1}(t^o_{n_i}) \geq fd^{m_j}_{n_1}(t^o_{n_i})$ must be the ordering if m_j relayed a reply and was added to $n_i's$ successor list according to Procedures 6 and 5 respectively. In time $t\ \varepsilon\ [t^o_{n_i}, t^c_{n_i}]$ consider these two possibilities:

i) $sn^{n_i}_{n_1}(t) > sn^{m_j}_{n_1}(t) > sn^{n_i}_{n_1}(t^o_{n_i})$ in which case m_j would no longer be a part of the successor set at $t^c_{n_i}$according to Procedure 5.

ii) $sn^{n_i}_{n_1}(t) = sn^{m_j}_{n_1}(t) = sn^{n_i}_{n_1}(t^o_{n_i})$ but $m'_j s$ reported distance has increased. At node m_j according to Procedure 7 for the same sequence number the feasible distance remains the same or is decreased. This implies the new ordering is $sn^{n_i}_{n_1}(t^c_{n_i}) = sn^{m_j}_{n_1}(t^c_{n_i}) \bigwedge fd^{n_i}_{n_1}(t^c_{n_i}) > fd^{m_j}_{n_1}(t^c_{n_i})$ which does not violate the ordering requirement.

For all other possibilities it is easily seen that the ordering is not violated. Thus Procedure 5 and 7 do not violate the ordering criteria.

Theorem 1: SMLDR is loop free at every instant as long as the nodes update their routing tables according to NDC and Procedures 5, and 7 and relay messages according to Procedures 2, 3, 4 and 6.

Proof: Let *I* be a downstream node for *A* towards the destination *D*. If *A* relays a reply and *I* accepts it then a loop would be formed. We prove this can never happen by contradiction. Initially at time t_0 let us assume the path from *A* to *I* to *D* is loop free. By the ordering criteria $sn^A_D(t_0) < sn^I_D(t_0)$ or $sn^A_D(t_0) = sn^I_D(t_0) \bigwedge fd^A_D(t_0) > fd^I_D(t_0)$. At time t_1 let *I* receive a reply relayed by *A* as per Procedures 4 or 6. At *I* $sn^I_D(t_1) \geq sn^I_D(t_0)$ as the sequence number is a nondecreasing function. If $sn^I_D(t_0) > sn^A_D(t_0)$ it means $sn^{rp}_D < sn^I_D(t_1)$ and *I* will not accept the *A*'s reply. Similarly if $sn^A_D(t_0) = sn^I_D(t_0) < sn^I_D(t_1)$ *I* cannot accept *A*'s reply. If $sn^I_D(t_0) = sn^A_D(t_0) = sn^I_D(t_1)$ we would have $rd^{rp}_D \geq rd^A_D(t_0) \geq fd^A_D(t_0) > fd^I_D(t_0) \geq fd^I_D(t_1)$ and *I* will not accept *A*'s reply as it will violate NDC.

Theorem 2: SMLDR ensures that a node A *initiating a*

route discovery for destination D *identified by the computation* (A, ID_A) *in an error free stable connected network receives a feasible route reply for destination* D.

Proof: Let node *A* initiate a request for destination *D* and let the request traverse the path $P = \{n_1, ..., n_{k-1}\}$ arriving at node n_k (possibly equal to *D*) before satisfying SDC. Node n_k initiates a reply for *D* towards *A*. The proof must show that the reply relaying mechanism ensures that the reply is usable by all relaying nodes. Let us first consider the case that no node along the solicited path *P* is affected by another route discovery event for *D* during the route request and establishment phase. Each node $n \varepsilon P$ must be in one of the three sub cases otherwise that node would have responded to the request instead of relaying it on towards n_k. *i)* n has no information about D, *ii)* n's invariants are invalid, *iii)* n has a valid route but its invariants cannot satisfy SDC. In case *i)* node n can use any reply sent by n_k. In case *ii)* the reply sent by n_k will satisfy the invariants at *n* as it satisfies *A*.

For case *iii)* the proof is by induction that the reply issued by n_k in response to *A*'s request satisfies all nodes along *P* if they followed Procedure 6. Consider for base case the node n_1, node *A*'s immediate neighbor. Node $n_1$would relay the request with $sn_D^{n_1}$and $fd_D^{n_1}$ if either $sn_D^{n_1} > sn_D^{rq}$ or $sn_D^{n_1} = sn_D^{rq} \bigwedge fd_D^{n_1} < fd_D^{rq}$ by Procedure 2. As the invariants in the request are only strengthened along the path any reply will satisfy both $n_1$and *A*. By the inductive hypothesis all nodes *A*, .., n_{i-1} will be able to use the reply. The proof that the reply will be satisfied at node n_i is identical to the base case (as the proof does not depend on the identity of the nodes).

Now suppose that some node $n_i \varepsilon P$ is affected by some independent route discovery or maintenance event for destination *D* during this phase. If one or more nodes $n_i \varepsilon P$ have a valid route to *D* their invariants were weaker than *A*'s invariants else they would not have relayed *A*'s request. Consider the event that one or more nodes $n_i \varepsilon P$ learn of a new route to *D* during this period. The new route will either have strengthened n_i's invariants or they remain the same. When a node receives a reply if the reply's invariants are weaker than the nodes invariants the node must discard the old reply and initiate a new reply with its stronger invariants according to Procedure 6. Thus even if the new discovery does strengthen the invariants of one node say n_t, the presence of the latest reply (with weaker invariants) will still cause the node n_t to issue a reply with its stronger invariants. It is possible for a node say n_p to receive two replies with the second reply having stronger invariants than the first though it has only one route request entry in its cache for an unique (A, ID_A) computation. This can cause n_p to not relay the second reply as the replied flag would have already been set for the route request entry. However from the above discussion we know that if n_p relays the first reply it satisfies the invariants of all nodes from n_p to *A* in the reverse path and hence is a feasible reply at *A*.

Theorem 3: Every active phase has a finite duration and SMLDR is live.

Proof: An active phase may never end either due to a deadlock or a livelock. The only way a deadlock can occur is when a node A is active for a computation (A, ID_A) without ever receiving a feasible reply. From Theorem 2 we know that each request is guaranteed a feasible reply if part of the network between the source and destination remains connected. Once a feasible reply is received the node reverts to passive state. Additionally whenever a node initiates a request it starts a timer. If the destination is unreachable the timer as described in Procedure 1 would expire causing the node to revert back to passive state. The node can then retry its request for the destination by entering the active phase again. The back-to-back active phase is prevented by setting a finite limit on the number of route request retries. When the limit is reached, the destination is declared unreachable to the higher layer and the node reverts to passive state thus preventing livelocks.

5. Simulations

The simulation results for SMLDR against AODV, AOMDV and LDR are presented. GloMoSim [1] was used for simulating the protocols. The IP layer uses a net queue size of 100 packets. All other simulation set up parameters are as described in [4] and [2]. AOMDV implementation is according to [13]. The multipath routing protocols use a threshold of 2 routes per destination stored in the routing table.

5.1. Simulation Environment

We consider simulation over a 2200m x 600m network containing 100 nodes. For the traffic load we use two scenarios of 30 flows with each flow sending packets of fixed size 512 bytes. To consider the effect of variation in load, in the first scenario the sources generate packets at the rate of 2 packets per second (pps) while in the second scenario they generate at the rate of 4 packets per second. This constitutes a net load of 60 packets per second (moderate load) and 120 packets per second (high load) respectively. The traffic flows start at a random time in the first 100 seconds and stays active till the end of the simulations. Both link layer feedback and Hellos are used for detecting link failures. A Hello loss of 2 is chosen to indicate link failure. The MAC layer used is 802.11 with a 250m transmission range and a throughput of 2 Mbps. To represent mobility we choose the random way point mobility model with each node moving at a random speed between 1-20 meters/sec. The simu-

lations were run for 900 simulated seconds with pause times 0s, 50s, 100s, 200s, 300s, 500s, 700s and 900s.

We choose packet delivery ratio, average end-to-end delay of data packets, normalized routing load, data hop count and the normalized route availability latency as the performance metrics of interest. Packet Delivery ratio is the fraction of CBR data packets received at the destination. Average data latency or the end-to-end delay includes the average of all possible delays for the data packets; from the time the data is transmitted to till it is received. Normalized routing load is the total number of control packets (route requests, route replies, route errors, and Hellos) divided by total number of received data packets. Data hop count is the average number of hops traversed by each packet which is computed as the number of data packets transmitted by each node over the total number of data packets received at the destination. The normalized route availability latency is the average sum of waiting time of all packets in the buffers of source nodes over the total number of packets sent.

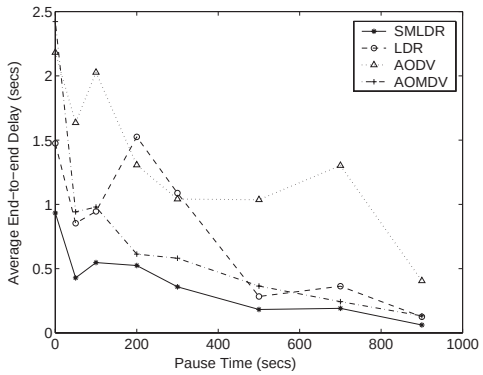
5.2. Simulation Results

Table 2 summarizes the results of the different metrics by averaging over all pause times for the two different traffic loads. The columns show the mean value and the 95% confidence interval.

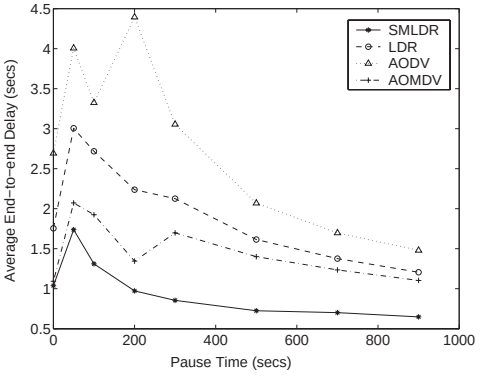
Multipath protocols on the whole perform significantly better than their single path counterpart. From the simulation results it is evident that the multipath mechanism chosen in the multipath routing protocols contribute to higher performance than those made by the corresponding underlying single path routing mechanisms.

Though LDR has a lower route availability latency than that of AODV for both traffic scenarios; there is still a need for the route discovery procedure to take place when the primary routes fail. Multipath routing protocols with their knowledge of additional routes avoid this phase and hence have the lowest route availability latency. SMLDR in particular upon detecting congestion due to Hello loss, is capable of switching to the minimal delay path among the available routes by the virtue of its shortest multipath design. This enables SMLDR to have a minimal normalized route availability latency of 71.335±28.33 (ms) for moderate load and 75.45±18.03 (ms) for the high load scenario.

Both route availability delay and propagation delay of data packets contribute to the data latency. Figures 2(a) and 2(b) show the average end-to-end delay or the data latency. For both traffic conditions and most pause times SMLDR exhibits less than two-third the data latency of AOMDV. For moderate load SMLDR has a latency of 0.4029±0.272 (s) compared to AOMDV's 0.7844±0.798 (s) and for high load SMLDR has a latency of 0.998±0.4045 (s) compared to 1.484±0.4082 (s) of AOMDV. AODV and LDR have significantly higher data latencies for both scenarios. The multipath mechanisms AOMDV and SMLDR exhibit about half the delay compared to their single path routing protocols.



(a) Moderate Load, 30 Flows, 60pps



(b) High Load, 30 Flows, 120pps

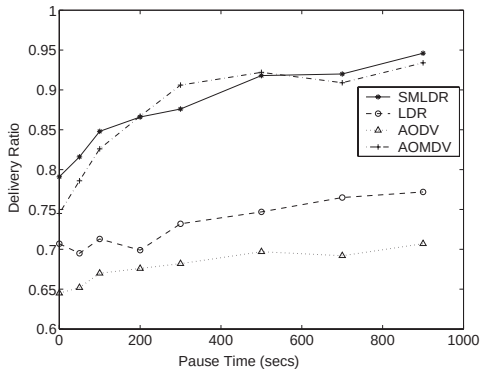
Figure 2. Data Latency

Figures 3(a) and 3(b) indicate the packet delivery ratio. For the moderate load scenario SMLDR has data delivery 0.872±0.0588 which is statistically equivalent to that of AOMDV. LDR delivers 0.728±0.0327 and AODV manages 0.676±0.0235. For the high load scenario SMLDR distinctly delivers higher than AOMDV however with only a slight improvement on the average. Higher delivery ratio of SMLDR is due to the fact that the route selection is more optimized by choosing shortest multipaths against the longer paths.

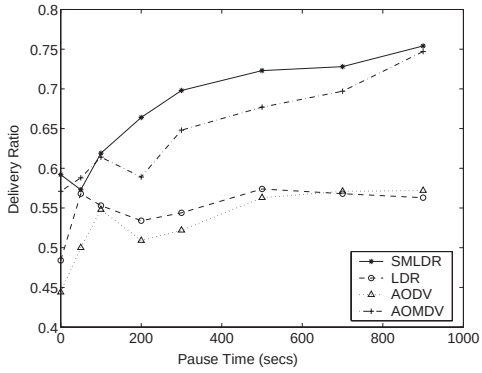
The normalized routing overhead is depicted in Figures 4(a) and 4(b). For the single path routing protocols the overhead for high-load scenario is compounded by the false negatives introduced by the keep-alive packets. SMLDR has lower routing overhead than AOMDV and the single path routing protocols. SMLDR exhibits the least overhead of

Name	Load (pps)	Delivery Ratio	Data Latency (s)	Routing Load	Data Hops	Route Latency (ms)
SMLDR	60	0.872±0.0588	0.403±0.272	7.221±2.800	5.457±0.360	71.33±28.33
AOMDV	60	0.862±0.7606	0.784±0.798	9.058±3.212	5.646±0.413	104.57±38.19
LDR	60	0.728±0.0327	0.832±0.585	8.626±2.643	5.523±0.415	201.84±101.53
AODV	60	0.676±0.0235	1.366±0.631	8.891±2.692	5.787±0.301	238.98±86.41
SMLDR	120	0.668±0.074	0.998±0.404	4.981±1.758	5.650±0.470	75.45±18.03
AOMDV	120	0.641±0.067	1.484±0.408	5.416±2.281	6.045±0.488	111.54±37.53
LDR	120	0.548±0.032	2.005±0.696	6.703±1.711	5.763±0.619	274.79±118.51
AODV	120	0.528±0.048	2.839±1.159	8.181±2.264	6.080 ± 0.910	382.91±172.68

Table 2. Performance Average over all pause times

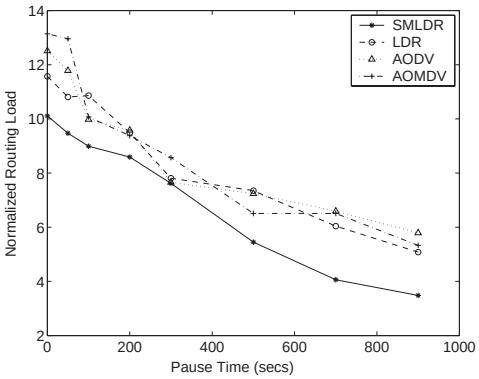


(a) Moderate Load, 30 Flows, 60pps

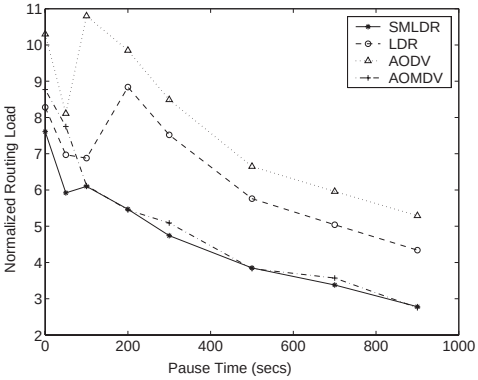


(b) High Load, 30 Flows, 120pps

Figure 3. Packet Delivery Ratio



(a) Moderate Load, 30 Flows, 60pps



(b) High Load, 30 Flows, 120pps

Figure 4. Normalized Routing Load

7.221±2.800 for moderate load and 4.981±1.758 for the high load scenarios. AOMDV's overhead of 5.416±2.281 is lower than that of LDR and AODV for high load scenario. However for the moderate load scenario AOMDV displays a high overhead of 9.058±3.212. In the higher mobility cases with low data rate, path failures and route discoveries are more frequent and hence the routing overhead is higher. All protocols have comparable data hop counts though for the high traffic load scenario AODV and AOMDV have slightly higher data hop counts. Higher hop counts imply that the routes taken by the data packets are not optimal.

6. Conclusions

We have presented a generalized framework to incorporate shortest multipath routing and identified the basic seven classes that define multipath routing mechanisms. Detailed procedures for the operation of SMLDR are outlined and the correctness established through formal proofs. By using the limiting distance information we have shown that it is possible to attain minimal delays for distance vector protocols. Though SMLDR does not restrict itself to shortest equal cost paths we have shown that SMLDR can be easily modified for shortest equal cost multipath routing.

We have demonstrated the effectiveness of shortest multipaths over both single path and multipath routing. Simulation results clearly show that shortest multipath routing distinctly outperforms other routing protocols. SMLDR has the least data latency and highest packet delivery for both moderate and high traffic loads for most pause times while exhibiting low network overhead. The data hop count in the case of shortest multipath routing is comparable and for most cases lesser than the other protocols. Low route availability latency and low data hop count identify SMLDR as a fast responsive routing protocol.

A key assumption we have made for the correct operation of SMLDR is that routers never forget the last sequence number they learn for a given destination. In practice, however, this may not be the case. The proposed approach can be modified based on the framework for destination-based sequence numbers proposed by Garcia-Luna-Aceves and Rangarajan [8].

References

[1] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla. Glomosim: A scalable network simulation environment. *Tech. Report. 990027, UCLA Computer Science Department*, 1999.

[2] 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. *Proceedings of ACM MOBIHOC*, pages 85–97, 1998.

[3] S. Corson and J. Macker. Mobile ad hoc networking (manet): Routing protocol performance issues and evaluation considerations. *IETF RFC 2501*, 1999.

[4] S. R. Das, C. E. Perkins, and E. M. Belding-Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. *Proceedings IEEE INFOCOM*, pages 3–12, 1999.

[5] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient energy-efficient multipath routing in wireless sensor networks. *Proceedings ACM MOBIHOC*, pages 251–253, 2001.

[6] J. J. Garcia-Luna-Aceves. Loop-free routing using diffusing computations. *IEEE/ACM Transactions on Networking*, 1(1):130–141, 1993.

[7] J. J. Garcia-Luna-Aceves, M. Mosko, and C. E. Perkins. A new approach to on-demand loop-free routing in ad hoc networks. *Proceedings PODC*, pages 53–62, 2003.

[8] J. J. Garcia-Luna-Aceves and H. Rangarajan. A new framework for loop-free on-demand routing using destination sequence numbers. *Proceedings IEEE MASS*, 2004.

[9] D. B. Johnson, D. A. Maltz, and Y. Hu. The dynamic source routing protocol for mobile ad hoc networks (dsr). *IETF Internet draft, draft-ietf-manet-dsr-09.txt*, 2003.

[10] S. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. *Proceedings IEEE ICC*, pages 3201–3205, 2001.

[11] W. Lou, W. Liu, and Y. Fang. Spread: Enhancing data confidentiality in mobile ad hoc networks. *Proceedings IEEE INFOCOM*, 2004.

[12] M. K. Marina and S. R. Das. On-demand multipath distance vector routing in ad hoc networks. *Proceedings IEEE ICNP*, pages 14–23, 2001.

[13] M. K. Marina and S. R. Das. Ad hoc on-demand multipath distance vector routing. *Technical Report, Computer Science Department, Stony Brook University*, 2003.

[14] A. Nasipuri, R. Castaneda, and S. R. Das. Performance of route caching strategies in dynamic source routing for on-demand protocols in mobile ad hoc networks. *ACM/Kluwer Mobile Networks and Applications (MONET)*, 6(4):339–349, 2001.

[15] V. D. Park and M. S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. *Proceedings IEEE INFOCOM*, 3:1405–1413, 1997.

[16] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc on demand distance vector (aodv) routing. *IETF RFC 3561*, 2003.

[17] J. Raju and J. J. Garcia-Luna-Aceves. A new approach to on-demand loop-free multipath routing. *Proceedings IC3N*, pages 522–527, 1999.

[18] A. Valera, W. Seah, and S. V. Rao. Cooperative packet caching and shortest multipath routing in mobile ad hoc networks. *Proceedings IEEE INFOCOM*, pages 260–269, 2003.

[19] S. Vutukury and J. J. Garcia-Luna-Aceves. A simple approximation to minimum-delay routing. *Proceedings ACM SIGCOMM*, pages 227–238, 1999.

[20] S. Vutukury and J. J. Garcia-Luna-Aceves. Mdva: A distance-vector multipath routing protocol. *Proceedings IEEE INFOCOM*, pages 557–564, 2001.

[21] Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi. A framework for reliable routing in mobile ad hoc networks. *Proceedings IEEE INFOCOM*, pages 270 – 280, 2003.

[22] W. T. Zaumen and J. J. Garcia-Luna-Aceves. Loop-free multipath routing using generalized diffusing computations. *Proceedings IEEE INFOCOM*, 3:1408–1417, 1998.