J. Indian Inst. Sci., Nov.–Dec. 2006, **86**, 533–552

An approach to QoS-constrained path protection in MANETs using segment-backup paths

AJAY AGARWAL AND BIJENDRA N. JAIN*
Department of Computer Science and Engineering, Indian Institute of Technology, Delhi, New Delhi 110 016, India.
email: {aagar, bnj}@cse.iitd.ernet.in

Received on January 2, 2006; Revised on October 20, 2006.

Abstract

In the context of mobile ad hoc networks (MANETs), we consider the problem of identifying (a) an optimal primary path which satisfies the required QoS constraints, and (b) a set of alternate paths that may be used in case a link or a node on the primary path fails. The alternate paths are also required to satisfy the same set of QoS constraints as is the case with the primary path. In the paper, we have proposed that the traffic be re-routed along a subpath that bypasses a segment of the primary path that contains the failed link or node. The segments are determined based on (a) availability of alternate paths, so that (b) QoS constraints are met. This flexibility in identifying the segments can also be used to ensure that the delay in switching traffic over to an alternate path, and the resulting packet loss, are bounded. This paper is focused on proving that for a given (a) source–destination pair of nodes in a network, and (b) any primary path between them, the nodes on the primary path can be divided into a collection of segments such that for each segment there exists an alternate path which completely bypasses the segment **if and only if** there exist two or more node-disjoint paths between the source and destination nodes. This implies that if there exists a solution consisting of a set of alternate backup paths *for a given primary path* then such a solution can always be found for *any* primary path. We describe an algorithm to identify (a) a primary path, (b) the collections of segments, and (c) the corresponding set of alternate paths, one for each segment, each of which satisfies specified QoS constraints, so that the delay in switching traffic over to an alternate path is bounded.

Keywords: Ad hoc networks, path protection, segment-backup paths, QoS-routes.

1. Introduction

In a mobile ad hoc network (MANET) a routing protocol invariably results in identifying a route from a given source node S to a given destination node D. Such a route is *optimum* in that it satisfies a required set of QoS constraints, specified in terms of end-to-end delay, packet loss, jitter, etc. Such a path is referred to as the *primary* path. However, when the primary path fails, one would switch traffic over to an alternate path, if such a path has been identified. Otherwise, a new path is discovered. In *single path* routing schemes [1–4] a new path is discovered only after detecting failure of a node (or link) along the primary path. Clearly, the resulting switch-over delay and packet loss will be significant, and enough to render applications such as VoIP to be less meaningful if run over MANETs. If, instead, *multiple paths* are identified upfront, then a new route discovery is required only when all paths fail.

*Author for correspondence.

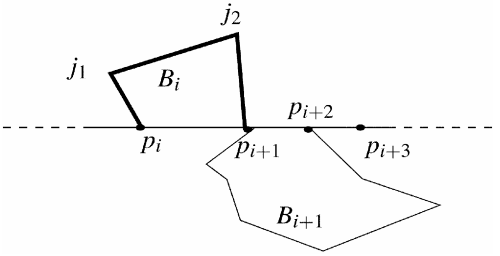

FIG. 1. Link bypass mechanism.

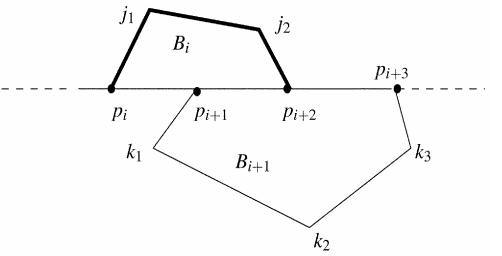

FIG. 2. Node bypass mechanism.

The solutions proposed in [5–14] suggest a variety of approaches to identifying (and setting up) alternate paths. These include re-routing packets along a subpath that bypasses the failed link or node, or one that bypasses the entire path (i.e. node-disjoint path) or the remaining portion thereof (Figs 1–3). Routing protocols such as TORA [11] and others [5–10, 12] (Fig. 4) call for reorganization of the identified directed-acyclic graph (DAG) rooted at destination D and which represents a host of alternate routes from S to D.

These routing protocols, as well as the one proposed in the paper, assume that if a node (or link) on the primary path fails then the other nodes (and links) continue to be operational. In that case the alternate path, whether based on link (or node) bypass or based on TORA, will continue to be available for use. This assumption can be justified on account of the facts that (a) the identification of the primary and alternate paths is done at the time data transfer is required to be carried out, and (b) a change in the network topology is isolated or that multiple changes in the topology are rare. These assumptions are also needed if one considers and insists on QoS to be guaranteed in respect of certain parameters.

QoS-aware ad hoc routing protocols [13, 14], on the other hand, are based on multipath routing that use multiple paths that are node-disjoint. This has several drawbacks.

(1) Usually, a working path that satisfies a QoS constraints is determined first. Then, subsequently, a node-disjoint path is identified as a backup path. In some cases, the working path so determined may be routed through the network in a manner such that it "blocks" all node-disjoint paths in the network. In Fig. 5, for example, there exist two node-disjoint paths $B1 = \langle S, B, D\rangle$ and $B2 = \langle S, C, D\rangle$ from source S to destination D. However, if the working path selected is $P = \langle S, B, C, D\rangle$, then clearly there is no path which is node-disjoint with the working path.

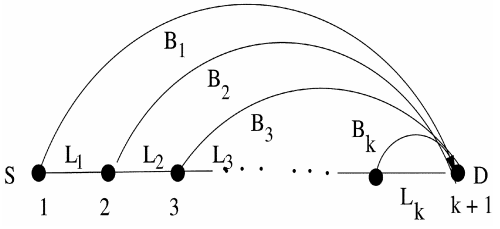

FIG. 3. Remaining-links bypass strategy (source courtesy [10]).

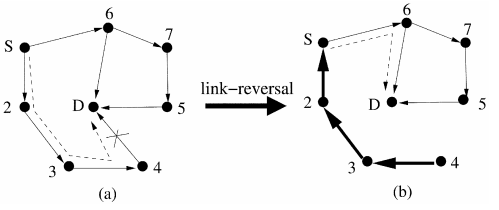

FIG. 4. A new DAG shown in Fig. (b) is formed by reversing links (shown in bold) if link (4, D) in a network shown in Fig. (a) fails.

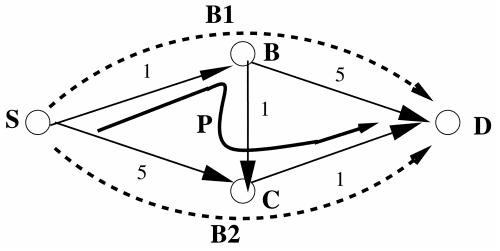

FIG. 5. Delay over each link. No end-to-end backup path, if working path is $\langle S, B, C, D \rangle$.

(2) The switch-over delay can be as high as the delay of the working path from the source to the destination. This happens when the last link fails.

The existing algorithms or protocols do not explicitly require that the alternate routes so identified also satisfy the same set of QoS constraints that are satisfied by the primary path. One is, therefore, tempted to re-visit the algorithms and ensure that the resulting alternate routes also satisfy the QoS constraints. However, the more fundamental concern has to do with the conditions under which such alternate paths exist. In particular, it is unclear as to "for a given primary path, what is sufficient to guarantee that there will exist a complete set of paths that bypass nodes along the primary path". Or, "for a given primary path, what is sufficient to guarantee that a node-disjoint path exists".

It is also unclear whether these sufficient conditions are necessary as well. The answer to one of these questions is clear. Even if there exist two node-disjoint paths between a given pair of source and destination nodes, for the particular selected primary path P, there may *not* exist a path that is node-disjoint to P (see Fig. 5, where P = $\langle S, B, C, D \rangle$). The difficulty is that the primary path is invariably identified first based on certain QoS preferences, and it is only later that one searches for an alternate node-disjoint path. In the context of MANETs, we have proposed that the traffic be re-routed along a subpath that bypasses a segment (or a portion) of the primary path that contains the failed link or node. The identification of the segments (and their size) is not fixed a priori but will be determined based on (a) the availability of alternate paths, so that (b) QoS constraints are met. This has several implications, the most significant of which has to do with the availability of alternate paths.

In Section 2, we formally define the notion of segment-backup paths. Section 3 shows the existence of a complete set of segment-backup paths and its one-to-one correspondence with a pair of node-disjoint paths. In Section 4, we argue that the proposed scheme based on segment-backup paths is capable of addressing QoS constraints in a comprehensive manner. Algorithm to compute these segment-backup paths is given in Section 5. Concluding remarks are made in Section 6.

2. Segment-backup paths

The notion of segments has simultaneously been introduced in MPLS [15] and wireline [16] networks wherein the network topology is known a priori. In MANETs, on the other hand, the network topology is not known a priori but using a suitable ad hoc routing protocol can significantly be determined by discovering a large number of loop-free paths between a given source and destination. In this section, we introduce the notion of *a complete set of*

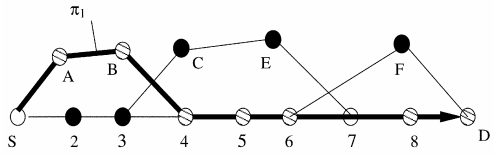

FIG. 6. A segment-backup path π_1 (shown bold) is used when either node 2 or 3 fails.

segment-backup paths which may also be applied suitably in the context of MANETs. A complete set of segment-backup paths protects against failure of one or more nodes/links in a segment of the primary path. A segment, in turn, consists of one or more contiguous nodes and associated links.

Consider, for example, a *portion* of the MANET given in Fig. 6. There, we assume that the primary path from source S to destination D is given by $\pi_0 = \langle S, 2, 3, \ldots, 8, D\rangle$. We view the primary path as made up of, for instance, three *segments*, viz. $\langle 2, 3\rangle$, $\langle 4, 5, 6\rangle$, and $\{7, 8\}$. If a node in a given segment fails, we consider this to mean that the corresponding segment has failed. (We assume that the source node S and the destination node D never fail. This is reasonable since if S or D fails then there is no way that communication from S to D can be maintained.)

In Fig. 6, we have also shown several other subpaths, one of which is $\langle S, A, B, 4\rangle$. This subpath together with $\langle 4, 5, 6, 7, 8, D\rangle$ may be used to route packet from S to D in case a node in segment $\{2, 3\}$ fails (or any of the associated links (S, 2) and (2, 3) fails). Such a subpath $\langle S, A, B, 4\rangle$ is referred to as a *bridge*, and the resulting alternative path $\pi_1 = \langle S, A, B, 4, 5, 6, 7, 8, D\rangle$ is called a *segment-backup path*. Clearly, and similarly, if node 4, 5, or 6 in the next segment $\{4, 5, 6\}$ fails then the segment-backup path $\pi_2 = \langle S, 2, 3, C, E, 7, 8, D\rangle$ may be used. Similarly, the segment backup path $\pi_3 = \langle S, 2, 3, 4, 5, 6, F, D\rangle$ is used in case node 7 or 8 fails. The collection of the three segment-backup paths is called complete *set* of segment-backup paths.

2.1. *Formal definitions*

For a given network, let G = (N, L), where $N = \{1, 2, \ldots, n\}$ is the set of nodes, and $L = \{\langle i, j\rangle | \ i \neq j, \ i, j \in N\}$ is the set of bidirectional links. Further, for a given source node, $S = p_1$ and destination node, $D = p_n$, let $\pi_0 = \langle p_1, p_2, \ldots, p_{n-1}, p_n\rangle$ or $\langle S, p_2, \ldots, p_{n-1}, D\rangle$, be the primary path.

Definition 1: For a given primary path $\pi_0 = \langle p_1, p_2, \ldots, p_{n-1}, p_n\rangle$, the collection of nodes

$$\sigma = \{p_i, p_{i+1}, \ldots, p_j\}, \tag{1}$$

where $1 < i \leq j < n$ is said to be a *segment* provided $p_i, p_{i+1}, \ldots, p_j$ are contiguous nodes along π_0. Further, the *ordered* collection of segments

$$\Omega = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}, m \geq 0 \tag{2}$$

is said to be a *complete set of segments* on π_0 provided segments $\sigma_1, \sigma_2, \ldots, \sigma_m$ are pairwise disjoint and for each segment

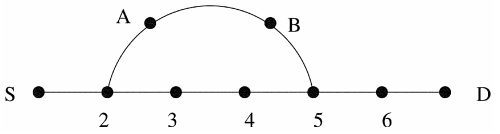

FIG. 7. A segment bypass P (σ) = $\langle S, 2, A, B, 5, 6, D\rangle$ corresponding to segment $\sigma = \{3, 4\}$.

$$\sigma_r = \{p_i^r, p_{i+1}^r, \ldots, p_j^r\},\ 1 \le r \le m, \tag{3}$$

$p_j^r = p_k$ implies $p_i^{r+1} = p_{k+1}$, for $r = 1, 2, \ldots, m-1$, $p_i^1 = p_2$, $p_j^m = p_{n-1}$. □

This definition of a *complete set of segments* ensures that every *intermediate* node on the primary path π_0 is contained in some segment. In Fig. 6, for example, $m = 3$ and $\sigma_1 = \{2, 3\}$, $\sigma_2 = \{4, 5, 6\}$, $s_3 = \{7, 8\}$.

Definition 2: For a given primary path $\pi_0 = \langle p_1, p_2, \ldots, p_{n-1}, p_n\rangle$, and a *segment* $\sigma = \{p_i, p_{i+1}, \ldots, p_j\}$, where $1 < i \le j < n$, a subpath

$$B(\sigma) = \langle x_1, x_2, \ldots, x_k\rangle \tag{4}$$

is said to be a *bridge* across the segment σ, if

(a) $x_1 = p_{i-1}$, $x_k = p_t$, $j + 1 \le t \le n$, and

(b) the subpath $\langle x_2, x_3, \ldots, x_{k-1}\rangle$ and π_0 have no common node(s). □

Definition 3: For a given primary path $\pi_0 = \langle S, p_2, \ldots, p_{n-1}, D\rangle$ and a *bridge* $B(\sigma) = \langle x_1, x_2, \ldots, x_k\rangle$ corresponding to a segment $\sigma = \{p_i, p_{i+1}, \ldots, p_j\}$, where $1 < i \le j < n$, the path

$$\text{P}\ (\sigma) = \langle S, p_2, \ldots, p_{i-2}, x_1, x_2, \ldots, x_k, p_{t+1}, \ldots, D\rangle, \tag{5}$$

where $x_1 = p_{i-1}$, $x_k = p_t$, $j + 1 \le t \le n$ is said to be a *segment bypass* corresponding to segment σ. □

The path P (σ) may or may not be useful in transferring packets from S to D in case any node $p_k \in \sigma = \{p_i, p_{i+1}, \ldots, p_j\}$ fails. In order to understand this fully, consider the network given in Fig. 7, where primary path $\pi_0 = \langle S, 2, 3, 4, 5, 6, D\rangle$, $\sigma_y = \{3, 4\}$, and P (σ_y) = $\langle S, 2, A, B, 5, 6, D\rangle$. It is tempting to suggest that if node 3 or 4 fails then node 2 will detect its failure or be notified of the node failure, and subsequently switch traffic over the bridge $\langle 2, A, B, 5\rangle$, resulting in all traffic going over P (σ_y) = $\langle S, 2, A, B, 5, 6, D\rangle$. This is perfectly acceptable, except that the strategy will not work in case node 6 fails. When that happens node 5 will detect failure of node 6 and notify node 2 which will redirect traffic over the bridge $\langle 2, A, B, 5\rangle$. Node 5 must necessarily drop all these re-routed packets. Alternatively, node 5 sends another notification to node 2, but along the upstream nodes B and A. Irrespective of the strategy used by node 5, the segment bypass P (σ_y) is useful only when node 6 is in some segment σ_z, where $z > y$, and there is a segment bypass, P (σ_z), that corresponds to σ_z.

This is equally true of a P (σ_k), $k = 1, 2, \ldots, z$. Therefore, to ensure that segment bypass, P (σ_k), $k = 1, 2, \ldots, z$, may be used to re-route traffic in case a node $p_i \in \sigma_k$ fails, it is necessary and sufficient that every node p_l on the primary path and beyond the nodes in σ_k is

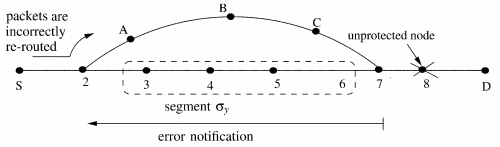

FIG. 8. A situation where packets are incorrectly re-routed to a segment bypass P (σ_k) = ⟨*S*, 2, *A*, *B*, *C*, 7, 8, *D*⟩ in case node 8 fails.

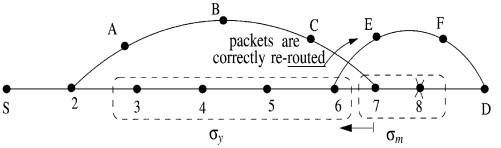

FIG. 9. A situation where packets are correctly re-routed to a segment-backup path $\pi(\sigma_k)$ = ⟨*S*, 2, *A*, *B*, *C*, 7, 8, *D*⟩ in case node 8 fails.

such that $p_l \in \sigma_y$, $y > k$. In that case a segment bypass P (σ_k) is referred to as a *segment-backup path*, $\pi(\sigma_k)$, and the collection of $\{\pi_k, k = 1, 2, \ldots, z\}$ is referred to as a *complete set of segment-backup paths* corresponding to *the* complete set of segments $\{\sigma_k, k = 1, 2, \ldots, z\}$.

Before stating the formal definition of a segment-backup path, it is worth re-stating that a segment bypass has the potential of being used as an alternate path, but only if certain other considerations are satisfied, the most important of which is that all nodes downstream along the primary path are also protected by some segment-backup path. Otherwise, the packets will be incorrectly re-routed along the bypass when an unprotected downstream node fails, as shown in Fig. 8. However, as shown in Fig. 9, if the downstream nodes are also protected then packets will be routed correctly along the segment-backup path corresponding to the segment that contains the failed node. Also note that the segment bypass corresponding to the last segment is by default a segment-backup path.

Definition 4: For a given primary path $\pi_0 = \langle S, p_2, \ldots, p_{n-1}, D\rangle$, a corresponding complete set of segments $\Omega = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$, and a segment σ_y, $1 \le y \le m$, a corresponding *segment bypass* P (σ_y) is a corresponding *segment-backup path*,

$$\pi(\sigma_y) = \text{P}\ (\sigma_y), \tag{6}$$

provided $y = m$ or if $y < m$, there exists a segment-backup path $\pi(\sigma_z)$, for every σ_z, $y < z \le m$. □

For brevity, $\pi(\sigma_y)$ is also written as π_y.

Definition 5: For a given primary path $\pi_0 = \langle p_1, p_2, \ldots, p_n\rangle$, a *segment* $\sigma = \{p_i, p_{i+1}, \ldots, p_j\}$ where $1 < i \le j < n$, and its corresponding bridge $B(\sigma) = \langle x_1, x_2, \ldots, x_k\rangle$, the first node x_1 of $B(\sigma)$) (where $x_1 = p_{i-1}$) is said to be a *segment switching router* (SSR) for the segment σ. Similarly, the last node x_k of $B(\sigma)$) (where $x_k = p_t$, $j + 1 \le t \le n$) is said to be the *segment merging router* (SMR) of the segment σ. □

The SSR for the segment σ has the responsibility to switch the traffic over the segment-backup path $\pi(\sigma)$ in case any component of the segment σ fails. Similarly, the SMR has the responsibility to merge the traffic coming from the source S over the remaining portion, viz. ⟨*SMR*, … , *D*⟩ of the primary path π_0.

It is important to note that while SSRs of two successive bridges $B(\sigma_z)$ and $B(\sigma_{z+1})$ cannot be the same, the SMRs of these bridges may be the same. In such case, the bridge $B(\sigma_{z+1})$ is

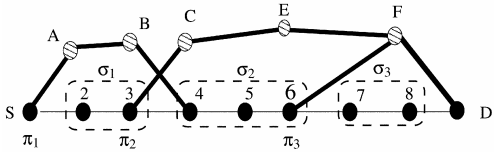

FIG. 10. Even if nodes 7 or 8 are protected by $\pi(\sigma_2)$ and $\pi(\sigma_3)$, path $\pi(\sigma_3)$ is used to switch traffic if node 7 or 8 fails.

said to be an *inner bridge* of the bridge $B(\sigma_z)$. To see the significance of *inner bridges*, consider the network given in Fig. 10, for example. Therein the segment-backup paths are $\pi(\sigma_1) = \langle S, A, B, 4, 5, 6, 7, 8, D\rangle$, $\pi(\sigma_2) = \langle S, 2, 3, C, E, F, D\rangle$ and $\pi(\sigma_3) = \langle S, 2, 3, 4, 5, 6, F, D\rangle$. While nodes in σ_2 are protected against failure by $\pi(\sigma_2)$, nodes in σ_3 may be protected by $\pi(\sigma_2)$ or by $\pi(\sigma_3)$. In other words, bridge $B(\sigma_3) = \langle 6, F, D\rangle$ is an *inner bridge* of the bridge $B(\sigma_2) = \langle 3, C, E, F, D\rangle$. In the proposed scheme it is implied that if a node in σ_3 were to fail then the segment-backup path $\pi(\sigma_3)$ must be used. That is, the responsibility of detecting failure and switching traffic over a bridge across σ_3 should be with the closest segment switching router, viz. node 6 in the above example. This is desirable since the time to detect and notify failure is likely to be minimum for node 6 (as opposed to node 3).

In this way, the maximum switch-over delay, defined as the delay between the time of node or link failure and the time when SSR switches data traffic over the corresponding segment-backup path, can be controlled since it depends significantly upon the segment size.

We now define a *complete set of segment-backup paths* in order to ensure that an alternative path is available in case any node along the primary path, viz. $p_k \in \{p_2, \ldots, p_{n-1}\}$, fails.

Definition 6: For a given primary path $\pi_0 = \langle S, p_2, \ldots, p_{n-1}, D\rangle$ and a *complete set of segments* $\Omega = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$, the set of segment-backup paths

$$\Pi(\pi_0) = \bigcup_{r=1,2,\ldots,m} \{\pi(\sigma_r)\}, \tag{7}$$

is said to be a *complete set of segment-backup paths* provided $\pi(\sigma_r)$ is a segment-backup path corresponding to segment σ_r for all $r = 1, 2, \ldots, m$. □

Apart from improved switch-over delay (as briefly mentioned earlier), the other major advantages of the scheme based on segment-backup paths are:

- A greater possibility of the existence of a complete set of segment-backup paths, and
- The possibility of ensuring that every alternate path also meets QoS constraints.

These are discussed in the sections that follow.

3. Existence of segment-backup paths

The proposed methodology has the advantage that the primary path may be selected based on QoS considerations rather than consideration of fault tolerance alone.

To illustrate, consider the network given in Fig. 11, where the delay over each link is identified. If the shortest delay path $\langle S, B, C, D\rangle$ is identified as the primary path from S to

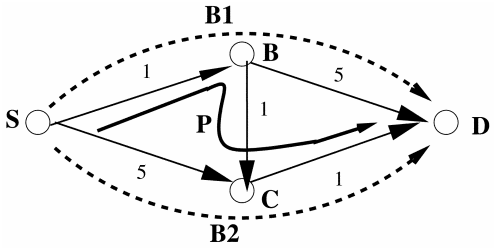

FIG. 11. No end-to-end backup path, if primary path is $\langle S, B, C, D\rangle$.

D, then there exists no alternate path that is node-disjoint with $\langle S, B, C, D\rangle$, even though there exist two node-disjoint paths between S and D. On the other hand, we can identify a complete set of segment-backup paths that protect against failure of node B or C. These are $\langle S, C, D\rangle$ and $\langle S, B, D\rangle$. In this section, we formally argue that for any selected primary path, there will always exist a complete set of segment-backup paths provided there exist two node-disjoint paths between the given source and destination nodes. This result is formally stated below as Theorem 1 (see also [16]).

Theorem 1. *In an undirected graph G, if* $\exists$ *two loop-free node disjoint paths Q and R from a given source node S to a given destination node D, then intermediate nodes on* any *given loop-free path* π_0 *from S to D can be broken into k primary segments,* $\Omega = \langle \sigma_1, \sigma_2, \ldots, \sigma_k\rangle$, $k \geq 0$, *such that corresponding to each* σ_i, $i = 1, \ldots, k$, $\exists$ *a segment-backup path* π_i.

Proof: Let $Q = \langle q_1, q_2, \ldots, q_b\rangle$ and $R = \langle r_1, r_2, \ldots, r_c\rangle$ be the given loop-free node disjoint paths from source S to destination D. Further, let $\pi_0 = \langle p_1, p_2, \ldots, p_a\rangle$ be any loop-free path from S to D. Note,

$$p_1 = q_1 = r_1 = S \text{ (source node)}, \tag{8}$$

and

$$p_a = q_b = r_c = D \text{ (destination node)}. \tag{9}$$

We now show that $\exists$ a set of segments, $\Omega = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$, consisting of intermediate nodes on the path π_0 and a set, $\pi = \{\pi_1, \ldots, \pi_k\}$, of segment-backup paths, where each π_i is a backup path for the corresponding segment σ_i. Since each segment-backup path π_i is different from a primary path only in respect of the bridge $B(\sigma_i)$ (Fig. 12), we may, equivalently, obtain a set of bridges $B(\sigma_i)$ in place of a set of paths, π_i.

Case 1: Length of π_0, $\ell(\pi_0) = 1$: Since $\pi_0 = \langle S, D\rangle$ has no intermediate node, there is no segment for which a segment-backup path needs to be identified. Therefore, $\Omega = \{\ \}$.

Case 2: Length of π_0, $\ell(\pi_0) > 1$: We consider the node p_2 on path π_0. Since Q and R are node disjoint, p_2 is either *not* on Q or *not* on R or both.

Let p_2 *not* be on R. In that case, $\exists$ a subpath of R, viz. $\langle r_1, r_2, \ldots, r_j\rangle$ for which $r_1 = p_1 = S$ and $r_j = p_k$ (for some $3 \leq k \leq a$ and $k = \min\{3, 4, \ldots, a\}$) such that $\langle r_1, r_2, \ldots, r_j\rangle$ and $\langle p_1, p_2, \ldots, p_k\rangle$ are node disjoint (Fig. 13). Note, in the worst case $r_j = p_k = D$. The segment, $\{p_2, p_3, \ldots, p_{k-1}\} = \sigma_1$, and $B(\sigma_1) = \langle r_1, r_2, \ldots, r_j\rangle$. That is, $\Omega = \{\sigma_1\}$ and $B(\Omega) = \{B(\sigma_1)\}$.

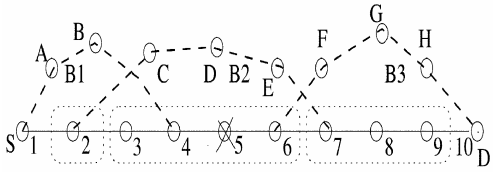

FIG. 12. Illustration of three bridges $B1$, $B2$ and $B3$ (shown dotted) corresponding to three segments $\sigma_1 = \{2\}$, $\sigma_2 = \{3, 4, 5, 6\}$ and $\sigma_3 = \{7, 8, 9\}$, respectively.

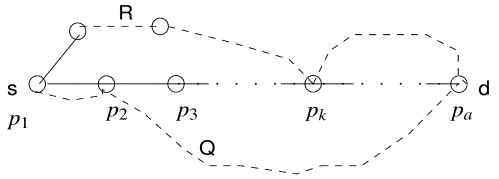

FIG. 13. If p_2 is not on R then $\exists$ a subpath of R which starts at S and ends at node p_k on primary path π_0 (for some $3 \le k \le a$ and $k = \min\{3, 4, \ldots, a\}$) such that it is disjoint with the path $\langle p_1, \ldots, p_k \rangle$.

Note the above discussion also takes care of the case where p_2 is neither on Q nor on R. The case when p_2 is not on Q can be similarly handled by interchanging the roles of Q and R.

If $k = a$, the sufficiency proof is complete. Otherwise, we need to identify the remainder of set Ω by considering the remaining subpath, $\langle p_{k-1}, p_k, \ldots, p_a \rangle$.

Now, consider node p_k. Since Q and R are node disjoint, p_k is *not* on Q (as p_k is on R). In that case, $\exists$ a subpath of Q, viz. $\langle q_i, q_{i+1}, \ldots, q_j \rangle$ for which $q_i = p_x$ (for some $1 \le x \le k-1$ and $x = \max\{1, 2, \ldots, k-1\}$) and $q_j = p_y$ (for some $k+1 \le y \le a$ and $y = \min\{k+1, k+2, \ldots, a\}$) such that path $\langle p_x, p_{x+1}, \ldots, p_y \rangle$ and $\langle q_i, q_{i+1}, \ldots, q_j \rangle$ are node disjoint (Note, in the worst case, $q_i = p_x = S$. Similarly, in the worst case, $q_j = p_y = D$). As a result, $\sigma_2 = \langle p_{x+1}, \ldots, p_{y-1} \rangle$ and the corresponding bridge, $B(\sigma_2) = \langle q_i, q_{i+1}, \ldots, q_j \rangle$. Now, consider $\Omega = \{\sigma_1, \sigma_2\}$. Clearly, since it may be the case that $\sigma_1 \cap \sigma_2 \ne \phi$, we redefine $\sigma_1 \leftarrow \sigma_1 - \sigma_2$. If $\sigma_1 = \phi$ then $\{\Omega \leftarrow \Omega - \{\sigma_1\}, B(\Omega) \leftarrow B(\Omega) - \{B(\sigma_1)\}\}$. Note: The operation $\leftarrow$ is an assignment, while $\sigma_1 - \sigma_2$ removes all nodes in σ_2 from σ_1.

Having shown how σ_2 and $B(\sigma_2)$ may be computed, we complete the proof using induction. We assume that at induction step $m \ge 1$, $\Omega = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$, such that σ_i, $i = 1, 2, \ldots, m$, are pairwise disjoint and $|\sigma_i| \ge 1$, and $B(\Omega) = \{B(\sigma_1), B(\sigma_2), \ldots, B(\sigma_m)\}$. Let $\sigma_m = \{p_{n+1}, p_{n+2}, \ldots, p_{t-1}\}$ and $B(\sigma_m) = \langle p_n, r_2, \ldots, p_t \rangle$. Now if $p_t = p_a$, the proof is complete. Otherwise, if p_t is *not* on R, we similarly identify a subpath $\langle r_f, r_{f+1}, \ldots, r_g \rangle$ for which $r_f = p_x$ (for some $1 \le x \le t-1$ and $x = \max\{1, 2, \ldots, t-1\}$), $r_g = p_y$ (for some $t+1 \le y \le a$ and $y = \min\{t+1, t+2, \ldots, a\}$), and path $\langle p_x, p_{x+1}, \ldots, p_y \rangle$ and $\langle r_f, r_{f+1}, \ldots, r_g \rangle$ are node disjoint. Note, in the worst cases, $r_f = p_x = S$ and $r_g = p_y = D$. Then $\sigma_{m+1} = \{p_{x+1}, \ldots, p_{y-1}\}$ and $B(\sigma_{m+1}) = \langle r_f, r_{f+1}, \ldots, r_g \rangle$. Let $\Omega \leftarrow \Omega \cup \{\sigma_{m+1}\}$. Then, unless $\sigma_1, \sigma_2, \ldots, \sigma_m, \sigma_{m+1}$ are pairwise disjoint, we re-compute

```
for all i ← 1 to m
   {σ_i ← σ_i − σ_{m+1},
      if σ_i = φ then {Ω ← Ω − {σ_i},
                       B(Ω) ← B(Ω) − {B(σ_i)}
                      }//end of if-then//
}//end of for all//
```

This completes the proof of sufficient condition. ■

The significance of this result is that one is not forced to select either of the two node-disjoint paths to be the primary path. The primary path may be selected based on considerations other than fault tolerance against node/link failure. Further, the above suggests that if the topology of the network is rich enough, as determined by whether or not there exists two or more node-disjoint paths between node S and D, then there will be at least one complete set of segment-backup paths. Hopefully there will be many more such complete segment-backup paths. And at least one of such complete sets will satisfy the QoS constraints.

The condition stated in Theorem 1 is more relevant in context of wireline networks [16] where network topology, and therefore the existence of two node-disjoint paths is known. On the other hand, in the context of MANETs the network topology, and therefore existence of two node-disjoint is not known a priori.

It is found that the converse of this result is also true. That is, for a given primary path, if there exists a corresponding complete set of segment-backup paths, then there exists a pair of node-disjoint paths between the source and destination nodes. It is more formally stated in Theorem 2.

Before stating Theorem 2, below we define a few operations and the notion of a *coalesced bridge*. This is required to prove the theorem below.

Let $\mathcal{N}(p) = \{x_1, x_2, \ldots, x_u\}$ denote the ordered set of nodes corresponding to a path $p = \langle x_1, x_2, \ldots, x_u \rangle$. Similarly, let $\mathcal{P}(X) = \langle x_1, x_2, \ldots, x_u \rangle$ denote the path corresponding to the ordered set $X = \{x_1, x_2, \ldots, x_u\}$. In other words, $\mathcal{N}(\mathcal{P}(X)) = X$, and $\mathcal{P}(\mathcal{N}(p)) = p$. Further, let $p_{(x_i, x_j)} = \langle x_i, x_{i+1}, \ldots, x_j \rangle$ denote the subpath of p that extends from x_i to x_j.

We now define $\oplus$ operation which joins two loop-free paths having at least one node in common.

Let $A = \langle a_1, a_2, \ldots, a_m \rangle$, and $B = \langle b_1, b_2, \ldots, b_n \rangle$ be two loop-free non-disjoint paths such that $\mathcal{N}(\mathrm{A}) \cap \mathcal{N}(\mathrm{B}) \neq \phi$. Then $A \oplus B$ is defined as:

$A \oplus B = \langle a_1, a_2, \ldots, a_i, b_{k+1}, \ldots, b_n \rangle$, where i is such that $a_i = b_k$, $1 \le i \le m$, $1 \le k \le n$ and $a_x \neq b_y$, $\forall$ x = 1, 2, … , $i - 1$, $y = 1, 2, \ldots, n$. Note, path $A \oplus B$ is also loop-free (it is because portions of the paths of A and B in $A \oplus B$ are node disjoint except for $a_i = b_k$).

Further, let ($\bullet$) operation "appends" two loop-free paths $P_1 = \langle a_1, a_2, \ldots, a_m \rangle$ and $P_2 = \langle a_m, a_{m+1}, \ldots, a_n \rangle$ resulting in a new loop-free path $P_1 \bullet P_2 = \langle a_1, a_2, \ldots, a_m, a_{m+1}, \ldots, a_n \rangle$ provided $\forall\ i, j, i \neq j, a_i \neq a_j$.

The notion of coalesced bridge is now defined below.

Definition: Let a primary path $\pi_0 = \langle S, p_2, \ldots, p_{n-1}, D \rangle$ and a corresponding complete set of segments $\Omega = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$. Further, let corresponding to two segments σ_i and σ_j there exist two bridges, viz. B_i, and B_j such that they have at least one common intermediate node (note the common node is not on the primary path π_0). Then a *coalesced bridge* of B_i and B_j is defined as $B_{ij} = B_i \oplus B_j$.

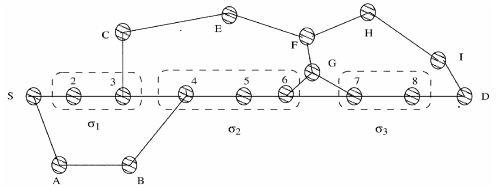

FIG. 14. An example network with three bridges, viz. $B(\sigma_1) = \langle S, A, B, 4\rangle$, $B(\sigma_2) = \langle 3, C, E, F, G, 7\rangle$ and $B(\sigma_3) = \langle 6, G, F, H, I, D\rangle$ wherein last two bridges have two nodes F and G in common.

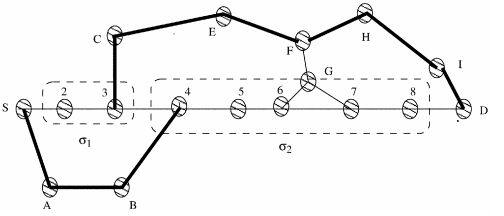

FIG. 15. B_is of Fig. 14 are made pairwise disjoint. Finally, two B_is, viz. $\langle S, A, B\rangle$ and $\langle 3, C, E, F, H, I, D\rangle$ are obtained (shown bold).

Figure 15 shows two bridges, viz. $B_1 = \langle S, A, B_i\rangle$ and $B_2 = \langle 3, C, E, F, H, I, D\rangle$ where bridge B_2 is a coalesced bridge of two bridges $B_2 = \langle 3, C, E, F, G, 7\rangle$ and $B_3 = \langle 6, G, F, H, I, D\rangle$ shown in Fig. 14.

Theorem 2 and its proof is given below.

Theorem 2. *If intermediate nodes on a given loop-free path π_0 from source node S to destination node D can be broken into segments, $\Omega = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$, for some $k \geq 1$, such that corresponding to each σ_i, $i = 1, \ldots, k$, $\exists$ a segment-backup path π_i, then $\exists$ two loop-free node-disjoint paths, say, Q and R from S to D.*

Proof: Case 1: $|\Omega| = 1$: Since there is only one segment, the segment-backup path π_1 and π_0 are node disjoint. Therefore, $Q = \pi_0$ and $R = \pi_1$.

Case 2: $|\Omega| > 1$: Let $\pi_0 = \langle p_1, p_2, \ldots, p_a\rangle$ be the given loop-free path from node S to node D, where $p_1 = S$ and $p_a = D$. Since each segment-backup path π_i differs from the primary path π_0 in respect of a bridge $B(\sigma_i)$ (or B_i, for short), we may, equivalently, consider the set of bridges, $B(\Omega) = \{B_1, \ldots, B_k\}$ corresponding to the set of paths $\pi = \{p_1, p_2, \ldots, \pi_k\}$.

Step 1: Select minimum number of bridges

Re-label all nodes on the path π_0 such that they are numbered 1 to a, S = 1, D = a. Now use Algorithm 1 to obtain a minimal set of segment-backup paths. Let $SMR(B_i)$ denote the last node of the bridge B_i while $SSR(B_i)$ denote the first node of B_i.

Algorithm 1: Computation of a minimal set of segment-backup paths corresponding to the path π_0.

```
1:  i ← 1
2:  while (SMR(B_i) < D) do
3:      Select B_j, j > i, such that SSR(B_i) < SSR(B_j) < SMR(B_i) < SMR(B_j), and j is the maximum.
4:      σ_i = σ_i ∪ σ_{i+1} ∪ … ∪σ_{j−1}          {size of σ_i is increased}
5:      Ω = Ω − {σ_{i+1}, σ_{i+2}, … , σ_{j−1}}    {delete merged segments}
6:      B(Ω) = B(Ω) − {B_{i+1}, B_{i+2}, … , B_{j−1}}  {delete corresponding bridges}
7:      Π = Π − {π_{i+1}, π_{i+2}, … , π_{j−1}}    {delete corresponding π_i s}
8:      i ← j
9:  end while
10: return, Ω, B(Ω),Π
```

Note, now a segment-backup path of Π (whose corresponding segment size is increased) may or may not satisfy the required switch-over delay. However, the set Π still protects all nodes on π_0 and, therefore, is *complete*. The segment-backup paths (or corresponding bridges) which have been deleted by Algorithm 1 are, in fact, redundant as regard to the construction of disjoint paths.

Algorithm 2: Joining every pair of nondisjoint bridges

```
1:  i ← 1                                              {initialize i}
2:  while (i ≤ |B(Ω)| − 1) do
3:    j ← i+1                                          {initialize j}
4:    while (j ≤ |B(Ω)|) do
5:      if (N(B_i) ∩ N(B_j) − N(π_0) ≠ φ) then
6:        B_i = B_i ⊕ B_j, B(Ω) = B(Ω) − {B_{i+1}, B_{i+2}, … , B_j}
7:        σ_i = σ_i ∪ σ_{i+1} ∪ …σ_j, Ω = Ω − {σ_{i+1}, σ_{i+2}, … , σ_j}
8:        π_i = π_{0(S,SSR(Bi))}·B_i·π_{0(SMR(Bi), D)}   {Note: B_i has been updated above}
9:        Π = Π − {π_{i+1}, π_{i+2}, … , π_j}.
10:       Relabel Ω, B(Ω) and Π
11:     end if
12:     j ← j + 1
13:   end while
14:   i ← i + 1
15: end while
```

Step 2: Relabel the sets Ω*,* $B(\Omega)$ *and* Π

Let the size of the each set, viz. Ω, $B(\Omega)$ and Π be x. Clearly, $x \le k$. The sets Ω, $B(\Omega)$ and Π are relabeled such that they are numbered 1 to x. Now, if $x = 1$ then go to Case 1 above, else go to Step 3.

Step 3: Join all pairs of non-disjoint bridges *into a coalesced bridge*

Algorithm 2 joins every pair of B_i and B_j, i.e. $B_i \oplus B_j$, $i > j$ provided B_i and B_j have at least one common intermediate node. Finally, it provides $B(\Omega)$, a set of (possibly coalesced) bridges which are pairwise-disjoint.

Step 4: Construct two node-disjoint paths

At the end of the Step 3, we have, say m, pairwise disjoint B_is, viz. $B_1, B_2, \dots, B_m$ corresponding to the segments $\sigma_1, \sigma_2, \dots, \sigma_m$. Then,

if m (number of B_is) is odd, let
$Q = \langle S, p_2\rangle \cdot \mathcal{P}(\sigma_1) \cdot B_2 \cdot \mathcal{P}(\sigma_3) \cdot B_4 \dots \mathcal{P}(\sigma_m) \cdot \langle p_{a-1}, D\rangle$,
and
$R = B_1 \cdot \mathcal{P}(\sigma_2) . B_3 \cdot \mathcal{P}(\sigma_4) \dots B_m$.
else let
$Q = \langle S, p_2\rangle \cdot \mathcal{P}(\sigma_1) \cdot B_2 \cdot \mathcal{P}(\sigma_3) \cdot B_4 \dots B_m$,

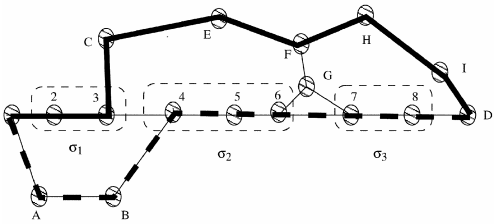

FIG. 16. Existence of two node disjoint backup paths (one shown as dotted and other as bold).

and

$$R = B_1 \cdot \mathcal{P}(\sigma_2) \cdot B_3 \cdot \mathcal{P}(\sigma_4) \ \ldots \ \mathcal{P}(\sigma_m) \cdot \langle p_{a-1}, D \rangle.$$

Now we claim that the paths Q and R, thus formed, are node disjoint.

Note,
1. B_i is node disjoint to $\mathcal{P}(\sigma_i)$ (by definition of bridges),
2. $\mathcal{P}(\sigma_i)$ is node disjoint to $\mathcal{P}(\sigma_j)$ (by definition of segments), and
3. B_i is node disjoint to B_j (since B_is are pairwise disjoint)

where $i \neq j$, $i = 1, \ldots, m$, and $j = 1, \ldots, m$.

Since both paths Q and R have either B_i or $\mathcal{P}(\sigma_i)$, $\forall\ i = 1, 2, \ldots, m$, then from the above three conditions, it can be inferred that Q and R are two node disjoint paths from S to D.

This completes the proof of necessary condition. ■

Corresponding to the minimal set of coalesced bridges shown in Fig. 15, Fig. 16 shows the two disjoint paths.

The significance of Theorem 2 is that if there exists a complete set of segment-backup paths for a **given** *primary path then there exists one or more complete sets of segment-backup paths for* **any** *primary path that one chooses. This is so since Theorem 2 implies that there exists two node-disjoint paths between S and D, and from Theorem 1,* **irrespective of what the specified primary path is***, there exists a complete set of segment-backup paths.*

This result is more significant in the context of MANETs where network topology and, therefore, existence of two node disjoint paths is not known a priori.

The second consequence of Theorems 1 and 2 is given in Corollary 1 and Corollary 2.

Corollary 1: *In an undirected graph G, if a given loop-free path π_0 from S to D can be broken into segments, $\Omega = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$, $k \geq 0$, each of which is of size one, such that corresponding to each σ_i, $i = 1, \ldots, k$, $\exists$ a node bypass, then $\exists$ two loop-free node-disjoint paths from node S to node D.*

The significance of this corollary is that if there exists a re-routing scheme based on using subpaths that bypass individual nodes on the primary path then the existence of two node-disjoint paths is assured. The latter in turn (from Theorem 1) implies the existence of a complete set of segment-backup paths for any primary path irrespective of the basis used to obtain the primary path. However, the converse is not true. That is, if there exists a

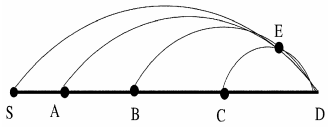

FIG. 17. A network where there does not exist a node bypass for every node on the primary path $\pi_0 = \langle S, A, B, C, D \rangle$. However, a complete set of segment-backup paths exists.

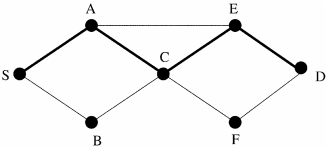

FIG. 18. A network where there does not exist a remaining-links bypass for every node on the primary path $\pi_0 = \langle S, A, C, E, D \rangle$. However, there exists a complete set of segment-backup paths.

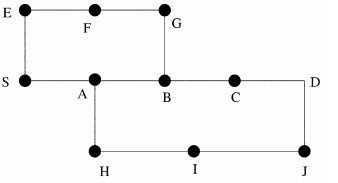

FIG. 19. A network where neither a node bypass nor there exist a remaining-links bypass for every node on the primary path $\pi_0 = \langle S, A, B, C, D \rangle$. However, a complete set of segment-backup paths exists.

complete set of segment-backup paths then there is no guarantee that there will exist a set of subpaths that bypass individual nodes on the primary path. The same can also be said about the re-routing scheme based on 'remaining-links bypass' (see corollary 2 below). In Fig. 17, for example, a network is shown where there does not exist a set of subpaths that bypass individual nodes on the primary path $\pi_0 = \langle S, A, B, C, D \rangle$. However, a complete set of segment-backup paths, viz. $\Pi = \{\langle S, E, D \rangle\}$ corresponding to π_0 exists. Similarly, in network (shown in Fig. 18) there does not exist a set of subpaths that bypass the remaining-links on the primary path $\pi_0 = \langle S, A, C, E, D \rangle$. However, a complete set of segment-backup paths viz., $\Pi = \{\langle S, B, C, E, D \rangle, \langle S, A, E, D \rangle, \langle S, A, C, F, D \rangle\}$ corresponding to π_0 exists. Moreover, another network, shown in Fig. 19 depicts a situation where there does not exist a set of subpaths that bypass individual nodes or that bypass the remaining-links on the primary path $\pi_0 = \langle S, A, B, C, D \rangle$. However, a complete set of segment-backup paths, viz. $\Pi = \{\langle S, E, F, G, B, C, D \rangle, \langle S, A, H, I, J, D \rangle\}$ corresponding to π_0 still exists. It is because, in all the three cases, the network is such that there exists two node-disjoint paths between S and D.

Corollary 2: *In an undirected graph G, if a given loop-free path π_0 from S to D can be broken into segments, $\Omega = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$, $k \geq 0$, each of which is of size one, such that corresponding to each σ_i, $i = 1, \ldots, k$, $\exists$ remaining-links bypass then $\exists$ two loop-free node-disjoint paths from node S to node D.*

To that extent the proposed scheme based on segment bypass provides greater flexibility in choosing the primary path as well as the corresponding segment-backup paths. Alternatively, one may view the proposed scheme based on segment bypass to be a generalization of the two schemes based respectively on (a) node bypass and (b) complete path bypass. This is so since the size of individual segment is a priori *not* constrained to be 1 (as in node bypass) or $n - 2$ (as in complete-path bypass).

Below we argue that the proposed scheme based on segment-backup paths is capable of addressing QoS constraints in a comprehensive manner.

4. QoS-Constraints on alternate paths

Let $\pi_0 = \langle S, p_2, p_3, \ldots, p_{n-1}, D \rangle$ be the primary path with $n - 2$ intermediate nodes, viz. p_2, $p_3, \ldots, p_{n-1}$. It is important to note that the $n - 2$ intermediate nodes on the path π_0 can be

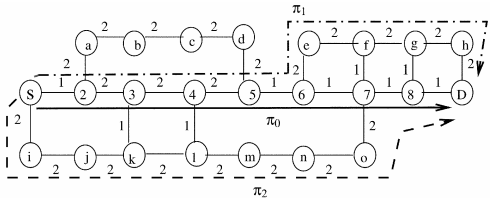

FIG. 20. Improved QoS guarantees.

partitioned into a complete set of segments in 2^{n-3} distinct ways. (The number of different complete sets of segments that can be formed using $m = n - 2$ intermediate nodes is given by $T(m) = \sum_{k=1}^{m} T(m-k)$; $T(0) = T(1) = 1$, whose solution is $T(m) = 2^{m-1}$; $T(0) = T(1) = 1$.) For instance, if there are 11 intermediate nodes, we can have 1024 different complete sets of segments.

Based on the network topology, a given complete set of segments may or may not satisfy the following properties:

(1) corresponding to each segment, there exists a segment-backup path,
(2) each segment meets specified switch-over delay, and
(3) each segment-backup path also meets specified QoS-constraints.

Since there are 2^{n-3} different ways to form the complete set of segments, there is greater probability of the above three constraints being met. This is not so when one is limited to working with schemes based on link or node bypass or complete route bypass.

Consider, for instance, the network given in Fig. 20, wherein we have also indicated the delay for each link. There are two node-disjoint paths $\pi_1 = \langle S, 2, 3, 4, 5, 6, e, f, g, h, D\rangle$ and $\pi_2 = \langle S, i, j, k, l, m, n, o, 7, 8, D\rangle$, and the corresponding delays are $\delta(\pi_1) = \delta(\pi_2) = 18$. If the stated QoS constraint on end-to-end delay is 17 then clearly both π_1 and π_2 do not meet the requirement. If, however, the primary path is chosen to be $\pi_0 = \langle S, 2, 3, 4, 5, 6, 7, 8, D\rangle$ then there is no other path which is node-disjoint from this primary path, π_0. However, there may still be ways to identify and construct a complete set of segment-backup paths. For instance, if $\sigma_1 = \{2\}$, $\sigma_2 = \{3, 4\}$, $\sigma_3 = \{5, 6\}$, $\sigma_4 = \{7\}$ and $\sigma_5 = \{8\}$, then one set of the corresponding segment-backup paths are:

$\pi(\sigma_1) = \langle S, i, j, k, 3, 4, 5, 6, 7, 8, D\rangle$,

$\pi(\sigma_2) = \langle S, 2, a, b, c, d, 5, 6, 7, 8, D\rangle$,

$\pi(\sigma_3) = \langle S, 2, 3, 4, l, m, n, o, 7, 8, D\rangle$,

$\pi(\sigma_4) = \langle S, 2, 3, 4, 5, 6, e, f, g, 8, D\rangle$, and

$\pi(\sigma_5) = \langle S, 2, 3, 4, 5, 6, 7, f, g, h, D\rangle$.

The corresponding delay for these segmented backup paths are $\delta(\pi(\sigma_1)) = 15$, $\delta(\pi(\sigma_2)) = 15$, $\delta(\pi(\sigma_3)) = 16$, $\delta(\pi(\sigma_4)) = 16$, $\delta(\pi(\sigma_5)) = 16$. Clearly, all of these meet the required delay constraint. Alternatively, if we select $\sigma_1 = \{2, 3\}$, $\sigma_2 = \{4, 5, 6\}$, $\sigma_3 = \{7\}$, $\sigma_4 = \{8\}$, together with corresponding segmented backup paths $\pi(\sigma_1) = \langle S, i, j, k, l, 4, 5, 6, 7, 8, D\rangle$, $\pi(\sigma_2) = \langle S, 2, 3, k, l, m, n, o, 7, 8, D\rangle$, $\pi(\sigma_3) = \langle S, 2, 3, 4, 5, 6, e, f, g, 8, D\rangle$, $\pi(r_4) = \langle S, 2, 3, 4, 5, 6, 7, f, g, h, D\rangle$, then the delay for these segmented backup paths is $\delta(\pi(\sigma_1)) = 15$,

$\delta(\pi(\sigma_2)) = 16$, $\delta(\pi(\sigma_3)) = 16$, $\delta(\pi(\sigma_4)) = 16$. These backup paths again meet the delay constraint.

An interesting and important point that needs to be made is that one can be sure that for any given primary path, π_0, between a source S and a destination D one will be able to identify at least one set of segments and corresponding a complete set of segment-backup paths. This is so if and only if the network is such that there are two or more node-disjoint paths between S and D. As a consequence, one is free to identify and use any route as the primary path, and then identify a complete set of alternate paths to address fault tolerance.

5. Computation of QoS-aware segment-backup paths

In this section, we illustrate a possible centralized algorithm to compute a primary path and the corresponding complete set of segment-backup paths that satisfy QoS constraints. But before we do so we describe below (a) model of the network (particularly from the view of performance parameters), and (b) the formal problem for which an algorithm is sought. (See also [17] for a distributed version of the algorithm.)

We represent the *network* as an undirected graph G = (N, L), where N is the finite set of nodes and L, the finite set of links. We denote by $|N|$ and $|L|$ the number of network nodes and links, respectively. A path from a node S to D is a finite sequence of distinct nodes $\pi = \langle p_1, p_2, \ldots, p_n \rangle$, where $p_1 = S$, $p_n = D$, $(p_i, p_{i+1}) \in$ L for all $1 \le i \le n - 1$ (Fig. 21). Since the QoS constraints we consider are all related to (a) an end-to-end path $\pi = \langle p_1, p_2, \ldots, p_n \rangle$, and (b) the network layer performance, we have identified two points X and Y in Fig. 21. All parameters related to the performance over the path are, therefore, specified or measured between points X and Y. Thus, and for instance,

- the end-to-end delay over π,

$$\delta(\pi) = \sum_{i=1,\ldots,n-1} \delta((p_i, p_{i+1})), \tag{10}$$

where $\delta((p_i, p_{i+1}))$ is the network layer delay over the link (p_i, p_{i+1}), represents end-to-end delay over path π,

- the bottleneck capacity,

$$\beta(\pi) = \min_{i=1,\ldots,n-1} \{\beta((p_i, p_{i+1}))\}, \tag{11}$$

where $\beta((p_i, p_{i+1}))$ is the installed capacity of the link (p_i, p_{i+1}) along π, and

- the path reliability,

$$\rho(\pi) = \prod_{i=1,\ldots,n-1} \rho((p_i, p_{i+1})), \tag{12}$$

where $\rho((p_i, p_{i+1}))$ is the reliability of the link (p_i, p_{i+1}) along π.

Note, we have to consider different types of cumulative QoS constraints, viz. additive (end-to-end delay), bottleneck (bandwidth) and multiplicative (end-to-end reliability).

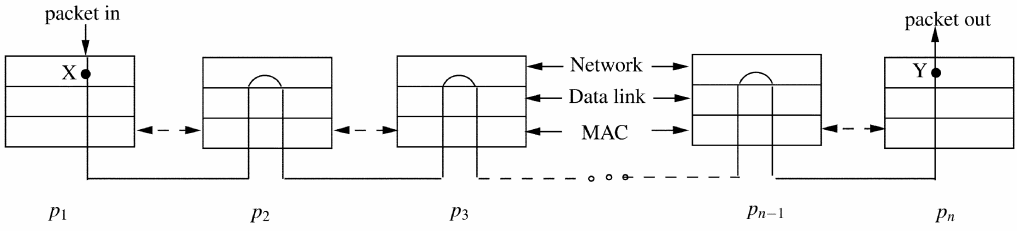

FIG. 21. End-to-end delay is the total delay occurred between X and Y.

The delay switching over from the primary path π_0 to any of the segment-backup path π_k, $\Delta(\pi_0, \sigma_k)$, is the delay in detecting, locating a fault in corresponding segment σ_k and in switching traffic from π_0 to π_k.

In what follows, we formally describe the problem.

Given:

- the source S, and the destination D,
- a list of all possible loop-free paths $\mathbb{P}$ from S to D (most of the ad hoc routing protocols essentially discover end-to-end loop-free paths by propagating control packets from source S to destination D),
- $\delta(p_i, p_j)$, $\beta(p_i, p_j)$, and $\rho(p_i, p_j)$ on the paths from S to D,
- the periodicity of liveness massages τ_p,
- the maximum permissible one way end-to-end delay d,
- the minimum required bandwidth b,
- the minimum required reliability r, and
- the maximum permissible switch-over delay T_{so},
  discover a primary path π_0 from S to D and a corresponding complete set of segment-backup paths $\{\pi_i\}$ that satisfy the following constraints:

$$\delta(\pi_0) \le d, \tag{13a}$$

$$\beta(\pi_0) \ge b, \tag{13b}$$

$$\rho(\pi_0) \ge r, \tag{13c}$$

$$\delta(\pi_i) \le d, \quad \forall i = 1, 2, \dots, k, \tag{14a}$$

$$\beta(\pi_i) \ge b, \quad \forall i = 1, 2, \dots, k, \tag{14b}$$

$$\rho(\pi_i) \ge r, \ \forall i = 1, 2, \dots, k, \tag{14c}$$

$$\Delta(\pi_0, \sigma_i) \le T_{so}, \quad \forall i = 1, 2, \dots, k \tag{15}$$

where

d = maximum permissible one way end-to-end delay, b = minimum required bandwidth, r = minimum required reliability, and T_{so} = maximum permissible switch-over delay.

1. $\Pi \leftarrow \phi$; $\Omega \leftarrow \phi$;
2. Select a suitable $\pi_0 \in \mathbb{P}$ that satisfies QoS constraints (13). If $\pi_0 = \langle S, D \rangle$ then return {SUCCESS, Π, Ω} else compute $n = |\pi_0| + 1$ and re-label all nodes in G such that nodes on π_0 are numbered in increasing order 1 to n, S = 1, D = n.
3. Corresponding to each $P_i \in \mathbb{P} - \{\pi_0\}$ compute a set $\mathbb{B}$ of all possible bridges.
4. Compute $\mathbb{P}$ wherein each $\mathcal{P}_i \in \mathbb{P}$ is a potential segment bypass from S to D corresponding to each $B_i \in \mathbb{B}$ capable of protecting nodes in the segment $\sigma_i = \mathcal{N}(\pi_0) - \mathcal{N}(\pi_0) \cap \mathcal{N}(\mathcal{P}_i)$.
5. For each $\mathcal{P}_i$, compute $\delta(\mathcal{P}_i)$, $\beta(\mathcal{P}_i)$ and $\rho(P_i)$ (see eqs (10)–(12)). Delete $\mathcal{P}_i$ from $\mathbb{P}$ if it does not satisfy any one of QoS-constraints in (14).
6. Compute a complete set of segments Ω and the corresponding complete set of segment-backup paths Π using Algorithm 3. If unsuccessful select another primary path π_0 and repeat steps 3 through 7.

Note: $|\pi|$ is the number of links over the path π.

FIG. 22. An algorithm to compute a complete set of segments Ω and a corresponding complete set of segment-backup paths Π for a given network.

For a given list of all possible paths $\mathbb{P} = \{P_1, P_2, \ldots, P_r\}$ and the pair of source-destination nodes, S and D, steps 1 through 6 of the procedure given in Fig. 22 help compute a complete set of segments Ω and a corresponding complete set of segment-backup paths Π. Together, the primary path and the corresponding segment-backup paths satisfy QoS constraints. The notation $\mathcal{N}(p) = \{x_1, x_2, \ldots, x_u\}$ denote the ordered set of nodes corresponding to a path $p = \langle x_1, x_2, \ldots, x_u \rangle$. This algorithm guarantees a solution if one exists.

Consider, for example, the directed network given in Fig. 23.

(1) Initialize $\Pi = \phi$, and $\Omega = \phi$.

(2) The possible paths from S to D are $\mathbb{P}$ = {⟨1, 2, 6, 8, 12⟩, ⟨1, 2, 6, 9, 12⟩, ⟨1, 2, 7, 9, 12⟩, ⟨1, 3, 7, 9, 12⟩, ⟨1, 3, 6, 8, 12⟩, ⟨1, 3, 6, 9, 12⟩, ⟨1, 4, 5, 8, 12⟩, ⟨1, 4, 6, 8, 12⟩, ⟨1, 4, 6, 9,12⟩, ⟨1, 2, 5, 8, 12⟩, ⟨1, 2, 5, 10, 11, 12⟩, ⟨1, 4, 5, 10, 11, 12⟩}.

(3) Let the selected primary path π_0 = ⟨1, 2, 6, 8, 12⟩. This is acceptable if π_0 satisfies constraints (13).

(4) The set of all bridges $\mathbb{B}$ over π_0 is $\mathbb{B}$ = {⟨6, 9, 12⟩, ⟨2, 7, 9, 12⟩, ⟨1, 3, 7, 9, 12⟩, ⟨1, 3, 6⟩, ⟨1, 4, 5, 8⟩, ⟨1, 4, 6⟩, ⟨2, 5, 8⟩, ⟨2, 5, 10, 11, 12⟩, ⟨1, 4, 5, 10, 11, 12⟩}.

(5) Correspondingly, the set of segment bypasses, $\mathbb{P}$ = {⟨1, 2, 6, 9, 12⟩, ⟨1, 2, 7, 9, 12⟩, ⟨1, 3, 7, 9, 12⟩, ⟨1, 3, 6, 8, 12⟩, ⟨1, 4, 5, 8, 12⟩, ⟨1, 4, 6, 8, 12⟩, ⟨1, 2, 5, 8, 12⟩, ⟨1, 2, 5, 10, 11, 12⟩, ⟨1, 4, 5, 10, 11, 12⟩}.

(6) Let each segment bypass $\mathcal{P}$ in $\mathbb{P}$ also satisfy constraints (14).

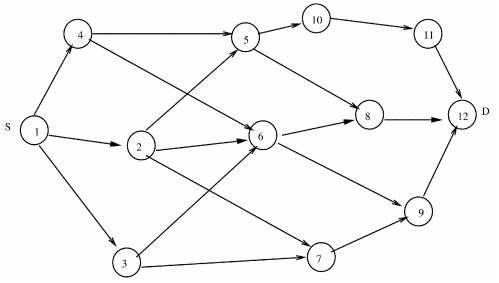

FIG. 23. An example network.

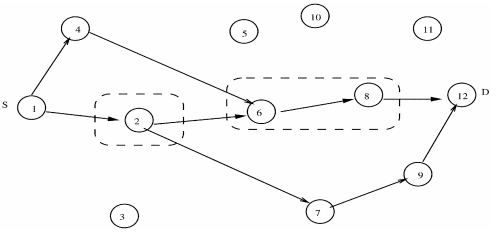

FIG. 24. A complete set of segments $\Omega = \{\{2\}, \{6, 8\}\}$ and a complete set of segment bypasses $\Pi = \{\langle 1, 4, 6, 8, 12\rangle, \langle 1, 2, 7, 9, 12\rangle\}$ for a network given in Fig. 23.

Algorithm 3: Computation of a complete set of segments Ω, and the corresponding complete set of segment-backup paths Π, from a given set of potential QoS-constrained segment bypasses, $\mathbb{P}$, corresponding to a given primary path $\pi_0 = \langle S, 2, \ldots, n-1, D\rangle$ and $S = 1$, $D = n = |\pi_0| + 1$.

1: $tbp = n - 1$; {$tbp \equiv$ node **t**o **b**e **p**rotected}
2: **while** (tbp > S) **do**
3: **Select** $\mathcal{P} \in \mathbb{P}$ **such that**
$\mathrm{SSR}(\mathcal{P}) < tbp < \mathrm{SMR}(\mathcal{P})$, $\mathrm{SSR}(\mathcal{P})$ is the smallest, and
$\Delta(\pi_0, \sigma) \le T_{so}$ where $\sigma = \{SSR(P) + 1, \ldots, tbp\}$
4: **if** (unsuccessful) **then exit**; **endif**
5: $\Omega \leftarrow \Omega \cup \{\sigma\}$; $\Pi \leftarrow \Pi \cup \{\mathcal{P}\}$;
6: **for all** $\mathcal{P}_i$ such that $SSR(\mathcal{P}_i) \ge SSR(P)$, $\mathbb{P} \leftarrow \mathbb{P} - \{\mathcal{P}_i\}$; {clean up}
7: $tbp \leftarrow SSR(\mathcal{P})$;
8: **end while**
9: return $\{\pi_0, \Pi, \Omega, tbp\}$; {tbp = 1 $\Rightarrow$ "SUCCESS"}

(7) If the delay on each link is 3 ms, periodicity of liveness messages is 1 ms, and specified $T_{so} = 8$ ms then a complete set of segment-backup paths Π and corresponding set of segments Ω using Algorithm 3 are shown in Fig. 24.

6. Conclusion

In this paper, we have considered the problem of together identifying (a) an optimal primary path which satisfies the required QoS constraints, and (b) a set of alternate paths that may be used in case a link or a node on the primary path fails. The alternate paths are also required to satisfy the same set of QoS constraints as is the case with primary path. Our approach is distinct from the ones that others have suggested. Others have suggested that traffic be re-routed along a subpath that bypasses a failed link or node, or that it be re-routed along a path that is completely link-disjoint (or node-disjoint) from the primary path. We have, on the other hand, proposed that the traffic be re-routed along a subpath that bypasses a segment (or a portion) of the primary path that contains the failed link or node. The identification of the segments (and their size) is not fixed a priori but will be determined based on (a) availability of alternate paths, and (b) so that QoS constraints are met. This has several implications, the most significant of which has to do with availability of alternate paths. It has been proved that:

Result: For a given (a) source-destination pair of nodes in a network, and (b) any primary path between them, the nodes on the primary path can be divided into a collection of segments such that for each segment there exists an alternate path which completely bypasses the segment **if and only if** *there exist two or more node-disjoint paths between the source and destination nodes.*

This ensures that if connectivity between a given pair of nodes is rich enough then for any primary path between them one can always find alternate paths so as to address the problem of link or node failure. This flexibility in identifying the segments (and thereby choosing their size) can also be used to ensure that the switch-over time, and the resulting packet loss, is within the prescribed bounds. This should be evident since the switch-over time is directly related to the one-way delay over a segment. We have described an algorithm to identify (a) a primary path, (b) the collections of segments, and (c) the corresponding set of alternate paths, one for each segment, each of which satisfies specified QoS constraints, and so that the delay in switching traffic over to an alternate path is bounded.

References

1. Gwalani, E. M. Royer, and C. E. Perkins, AODV-PA: AODV with path accumulation, *Next Generation Internet Symp., held in conjunction with ICC,* Anchorage, Alaska, May 2003.
2. D. B. Johnson, and D. A. Maltz, *Ad hoc networking*, Ch. 5, pp. 139–172, Addison Wesley (2001).
3. C. E. Perkins, and P. Bhagwat, Highly dynamic destination–sequenced distance-vector routing (DSDV) for mobile computers, *Proc. ACM SIGCOMM*, Aug. 1994, pp. 234–244.
4. C. E. Perkins, E. M. Royer, and S. R. Das, Ad hoc on demand distance vector routing (AODV), *Internet RFC 3561* (2002).
5. S. Guo, and O. W. Yang, Performance of backup source routing in mobile ad hoc networks, *Proc. IEEE Wireless Networking Conf.*, pp. 440–444 (2002).
6. S.-J. Lee, and M. Gerla, AODV-BR: Backup routing in ad hoc networks, *IEEE Wireless Communications and Networking Conf. (WCNC)*, Sept. 2000, Vol. 3, No. 1, pp. 1311–1316.
7. S.-J. Lee, and M. Gerla, Split multipath routing with maximally disjoint paths in ad hoc networks, *Proc. IEEE ICC 2001*, Helsinki, Finland, June 2001, pp. 3201–3205.
8. M. K. Marina, and S. R. Das, On demand multipath distance vector routing in ad hoc networks, *Proc. Int. Conf. for Network Protocols* (*ICNP*), Riverside, Nov. 2001.
9. A. Nasipuri, R. Castaneda, and S. R. Das, Performance of multipath routing for on-demand protocols in mobile ad hoc networks, *Mobile Networks Applic.*, **6**, 339–349 (2001).
10. A. Nasipuri, and S. R. Das, On-demand multipath routing for mobile ad hoc networks, *Proc. IEEE ICCCN'99*, Oct. 1999, pp. 64–70.
11. V. D. Park, and M. S. Corson, A highly adaptive distributed routing algorithm for mobile wireless networks, *Proc. IEEE INFOCOM'97*, Apr. 1997.
12. Y.-H. Wang, and C.-C. Chuang, Ad hoc on-demand backup node setup routing protocol, *J. Inf. Sci. Engng*, **20**, 821–843 (2004).
13. S. Chen, and K. Nahrstedt, Distributed quality-of-service routing in ad hoc networks, *IEEE J. Selected Areas Commun.*, **17**, 1488–1505 (1999).
14. R. Leung, J. Liu, E. Poon, A. Charles, and B. Li, MP-DSR: A QoS aware multipath dynamic source routing protocol for wireless ad hoc networks, *Proc. 26th IEEE Anual Conf. on Local Computer Networks (LCN 2001)*, Tampa, Florida, Nov. 2001, pp. 102–111.
15. A. Gupta, and B. N. Jain, QoS-aware path protection schemes for MPLS networks, *Proc. Int. Conf. on Computer and Communication (ICCC)*, Aug. 2002, pp. 103–118.
16. K. P. Gummadi, M. J. Pradeep, and C. S. R. Murthy, An efficient primary-segmented backup scheme for dependable real-time communication in multihop networks, *IEEE/ACM Trans. Networking*, **11**, 81–94 (2003).
17. A. Agarwal, Thesis under preparation, Department of Computer Science and Engineering, Indian Institute of Technology, Delhi 110 016, India.