Bandwidth Reservation in Multihop Wireless Networks: Complexity and Mechanisms

Leonidas Georgiadis  
Dept. Electrical & Computer Eng.  
Aristotle Univ. Thessaloniki  
Thessaloniki, 54124 Greece  
e-mail: leonid@auth.gr

Philippe Jacquet  
INRIA  
Domaine de Voluceau  
Rocquencourt  
France  
e-mail: philippe.jacquet@inria.fr

Bernard Mans  
Dept. of Computing  
Macquarie University  
Sydney, NSW 2109 Australia  
e-mail: bmans@ics.mq.edu.au

Abstract—We show that link interferences in multihop wireless networks make the problem of selecting a path satisfying bandwidth requirements an NP-complete problem, even under simplified rules for bandwidth reservation. This is in sharp contrast to path selection in wireline networks where efficient polynomial algorithms exist. We also describe a distributed mechanism for the problem of slot allocation according to bandwidth reservation in a wireless slotted environment.

I. INTRODUCTION

Multimedia flows need to satisfy certain performance requirements such as, bandwidth guarantees and upper bounds on packet losses and end-to-end delay in order to function properly. These requirements are generally known as Quality of Service (QoS) guarantees. Satisfaction of QoS guarantees has been an active area of research in wireline networks for several years. The same issue becomes more challenging in radio mobile networks where capacity is a scarce resource and packet transmissions are more prone to errors, noise and interference than wireline networks. This is especially true in wireless ad hoc networks, where packets are relayed by mobile nodes, rather than a base station as in cellular systems.

Two common mechanisms for providing Quality of Service (QoS) guarantees are, admission control and resource reservation. When a new flow is to be set between a source and a destination the source examines whether there are enough resources in the network so that the flow can be admitted and be provided with the required QoS, without affecting the already admitted connections (Admission Control). If these resources can be found, they are reserved for the flow (Resource Reservation) and transmission begins.

In this work we are concerned with bandwidth as a performance requirement of a flow. We assume that a flow needs a certain bit rate in order to satisfy its QoS. We are interested in providing these bandwidth requirements to flows in a wireless ad hoc network. In wireline networks, each link is physically isolated from the other links even if they are attached to the same node. Therefore admission control for a flow consists in finding a path from the source to the destination such that links on the path have enough remaining capacity to satisfy the flow’s bandwidth requirements. By “remaining capacity” is meant the link capacity minus the bandwidth already reserved for existing flows. Each node can advertise the remaining capacity of its links by periodically broadcasting this information in the network. Finding a path that satisfies the bandwidth requirements of a flow in a wireline network can be done using a simple modification of Dijkstra’s shortest path algorithm in $O(n \log n + m)$ time, in a network with $n$ nodes and $m$ links. In a wireless ad hoc network environment, the situation is different and much more complicated. The links are not isolated: traffic carried by neighbor links may interfere, especially if the links operate on the same channel-frequency, as it frequently happens with existing WLANs.

We address the problem of admission control and bandwidth reservation in wireless networks. We consider a proactive routing scheme where nodes obtain through periodic information exchange knowledge of network topology and state. Such a scheme is appropriate in environments with relatively low mobility and high internode traffic. Compared to reactive schemes, it has the advantage that nodes can use knowledge of network topology and state in order to select judiciously paths and thus avoid delays and a large number of control messages that may overwhelm the network. On the other hand, a reactive scheme that does not rely on knowledge of network topology is more appropriate in environments with relatively frequent topology changes and low internode traffic.

Our objective is to devise a scheme by which the node where the new flow arrives will be able to decide on the path that the traffic flow should follow based on available knowledge of topology and on the “available bandwidth” at the network nodes. The latter quantity (available bandwidth) should be a single number in order to avoid inordinate amount of information exchange throughout the network during the periodic updates. Based on this number, we adopt a simplified bandwidth reservation process that reflects the fact that a node’s transmission or reception affects its neighbors. Even with this simplified scheme, we show that the problem of selecting a path satisfying the bandwidth constraints is NP-hard. Moreover, it cannot be approximated in polynomial time within $n^\varepsilon$, $\varepsilon > 0$, provided that $P \neq NP$. It was already known [1] that the global optimization of flows admitted in a network, assuming that all flow requests are available at the beginning of time, is an NP-hard problem as it reduces to the multiknapsack problem. However the complexity of admission of a new request was not known. For a slotted wireless system it was known [10] that finding the slot scheduling along a given path that satisfies a given bandwidth request is NP-
complete.

In this paper, we also describe a distributed mechanism for admission control and bandwidth reservation along a given path in a slotted system, that ensure that each flow receives its requested bandwidth once it is admitted by the network. Finally, we present implementation issues related to mobile ad hoc routing protocols, such as OLSR. (Also, implementation issues related to CSMA/CA are available in the full paper [2]).

II. NP-Completeness

A. Node Remaining Capacity

Consider an undirected graph \(G(V, E)\) modelling a wireless network. Link \((i, j)\) means that nodes \(i\) and \(j\) can communicate. Let \(N_j\) be the neighbors of node \(j\). Denote by \(N_j(h)\) the set of nodes that are at most \(h\) hops away from node \(j\) (including node \(j\)), e.g., \(N_j(1) = \{j\} \cup N(j)\}.\) Node \(i\) has bandwidth \(C_i\), i.e., it can send a message to its neighbors at rate \(C_i\).

A transmission interferes with nodes that are within a number of hops, \(H_i\), from the transmitter and receiver, depending on the signal to noise ratio required for a correct reception. In the general case, for a successful transmission from node \(i\) to node \(j\), it must be ensured that no node within \(H_i\) hops from receiver \(j\) are transmitting at the same time. Moreover, in order to ensure that \(i\) ’s transmission does not interfere with other ongoing transmissions, it must be ensured that no nodes within \(H_i\) hops from \(i\) are receiving at the same time. Therefore reserving a unit of bandwidth for transmission from \(i\) to \(j\) requires the reservation of a unit of bandwidth from all receiving nodes within \(H_i\) hops from \(i\) and all transmitting nodes within \(H_i\) hops from \(j\).

To simplify the bandwidth reservation process, one can impose the rule that whenever node \(i\) transmits to node \(j\), the nodes in \(N_i(H_i)\cup N_j(H_j)\) cannot transmit nor receive packets - this is the basic mode of operation in a CSMA-CD system. Therefore, transmitting \(b\) units of bandwidth from \(i\) to \(j\), requires the reservation of \(b\) units of bandwidth on the nodes in \(N_i(H_i)\cup N_j(H_j)\). Determining the minimal units that need to be reserved is complicated and depends on the selected end-to-end flow path. Instead, we first consider a simplified bandwidth reservation process determined by the following rule:

**Rule A:** Whenever \(b\) units of bandwidth need to be transmitted between nodes \(i\) and \(j\), reserve \(b\) units of bandwidth on all nodes in \(N_i(H_i)\cup N_j(H_j)\).

We use Rule A in our presentation of NP-completeness results in order to simplify the discussion. In Section III, we discuss difficulties that arise when Rule A is used for bandwidth reservation and we introduce an extension of this rule that guarantees the existence of appropriate scheduling mechanisms. Using this extension, relatively simple bandwidth reservation mechanisms can be implemented without the need to carry inordinate amount of information throughout the network.

B. Incremental check

Assume that a path \(\Pi = \{v_1, \cdots, v_k\}\) from \(s(= v_1)\) to \(t(= v_k)\) is established in order to transmit a unit of bandwidth, and \(H_i = 1\). It is easy to see that for this established path, the bandwidth \(b_i\) required to be reserved according to Rule A on node \(i\) is equal to the number of edges in the established path \(\Pi\) that are reachable within one hop from \(i\). More specifically, for any node \(i\), \(b_i = |E(\Pi(i))|\) where \(E(\Pi(i)) = \{(v_j,v_{j+1})\}, 1 \leq j < k, \) such that \(\{v_j,v_{j+1}\} \subseteq \Pi\) and \((v_j \in N_i(1) \lor v_{j+1} \in N_i(1))\).

Assume that node \(i\) has (remaining) capacity \(C_i\). We must have

\[
C_i \geq b_i, \ i \in V \tag{1}
\]

We can now re-define formally our Bandwidth Admission Control into a “Path with Remaining Capacity problem” as follows.

**Definition 1:** The Path with Remaining Capacity problem (RC) is defined as:

**Instance:** A Graph \(G = (V, E)\), two vertices \(s\) and \(t\) from \(V\), and a Capacity \(C_i \in N\) for each vertex \(i\) from \(V\).

**Question:** Is there a simple path from \(s\) to \(t\) in \(G\) that satisfies the constraint \(C_i \geq b_i, \ i \in V\)?

In the following section, we prove that, because of the basic constraint (1), this problem is NP-complete.

C. NP-completeness proof

We first recall the definition of the Path with Forbidden Pairs problem (PFP).

**Definition 2:** The Path with Forbidden Pairs problem (PFP) (see GT54 in [4], [3]) is defined as:

**Instance:** A Graph \(G = (V, E)\), two vertices \(s\) and \(t\) from \(V\), and a collection \(C = \{(x_1,y_1), \ldots,(x_m,y_m)\}\) of pairs of vertices from \(V\).

**Question:** Is there a simple path from \(s\) to \(t\) in \(G\) that contains at most one vertex from each pair in \(C\)?

The PFP problem is known to be NP-complete (see GT54 in [4]). Variants of this problem exist where a measure is the length of the path (i.e., the number of edges in the path): the shortest feasible PFP and the longest feasible PFP problems. Not surprisingly both variants are NP-complete and also NPO PB-complete [5], i.e., in polynomial time, they cannot be approximated within \(n^\epsilon\) for some \(\epsilon > 0\), where \(n\) is the size of the input, provided that \(P \neq NP\).

**Theorem 3:** With \(H_i = 1\), the Path with Remaining Capacity problem (RC) is NP-complete.

**Proof:** It is easy to see that \(RC \in NP\), because a nondeterministic algorithm needs only to guess a path and check in polynomial time that the constraint \(C_i \geq b_i, \ i \in V\) is valid. (Of course, checking the constraint for the vertices of the path and for their neighbors is sufficient.)

We now give a polynomial reduction from this problem to the Path with Forbidden Pairs problem (PFP).

We first transform an instance \((G = (V, E), s, t, C)\) of the PFP problem to an instance \((G' = (V', E'), s, t, C')\) of the Remaining Capacity problem by formally defining:
\[ V' = V \cup \{v_{xy}\} \]
\[ E' = E \cup \{(x, v_{xy}), (y, v_{xy})\} \]
\[ s \text{ and } t \text{ are unchanged} \]
\[ C' \text{ is the capacity defined as: } C'_i = 2 \text{ for } i \in \{v_{xy}\} \]
\[ s_{xy} \text{ is smaller than 3, it is easy to check the two remaining possibilities.} \]

Informally, \( G' \) contains all the vertices of \( G \) as well as \( m \) vertices representing each forbidden pair. Let us define \( F \) as the set of \( m \) vertices in \( G' \) representing each forbidden pair of \( C \). Each vertex of \( F \) is only connected to its two respective "forbidden" vertices and is assigned a capacity equal to 2. All other vertices are assigned a capacity equal to \(|V|\) that is larger than any possible vertex bandwidth induced by a given established path. Of course, without loss of generality, we assume that \((s, t)\) is not a forbidden pair.

We now prove that a solution of this instance of the \( RC \) problem is a solution if and only if it is a solution for the original instance of the \( PFP \) problem. Let us first assume that the length of this solution path is at least 3.

It is easy to see that a solution path \( \Pi \) from \( s \) to \( t \) for the \( RC \) problem in \((G' = (V', E'), s, t, C')\) does not include any of the vertices of \( F \), as each vertex \( v \) of the path (except, possibly, \( s \) and \( t \)) requires \( b_v \geq 3 \). Hence this path \( \Pi \) is also a path \( \Pi \) in \( G \).

Furthermore, none of the forbidden pairs are included in the solution path. Otherwise there would exist a pair of vertices \( x \) and \( y \) (with \((x, y) \in C\)) that both belong to the solution path. This would imply that the vertex \( v_{xy} \) is reaching more than 2 edges of the path within 1 hop and that the basic capacity constraint for the vertex \( v_{xy} \) is not valid as the bandwidth \( b_{xy} \geq 3 > C_{xy} = 2 \), thus leading to a contradiction. Hence a solution for the \( RC \) problem in \((G' = (V', E'), s, t, C')\) is a solution for the instance \((G = (V, E), s, t, C)\) of the \( PFP \) problem.

Conversely, given a solution path \( \Pi' \) for the instance \((G = (V, E), s, t, C)\) of the \( PFP \) problem, we can verify that the path \( \Pi' \) is a feasible solution path for the \( RC \) problem in \((G' = (V', E'), s, t, C')\). Obviously, \( \Pi' \) is a simple path of \( G' \) as \( G \) is a subgraph of \( G' \). Hence, we just need to verify that the bandwidth induced by the path respects the capacity constraint. It is clear from the reduction that only the nodes of \( F \) may jeopardize the feasibility of the solution, as they may not have sufficient capacity. Again, none of them belong to the path. A vertex of \( F \) can be a neighbor of a vertex of \( \Pi' \), incurring a bandwidth of at most 2. However, a vertex of \( F \) cannot be a neighbor of two nodes in the paths as it can only be neighbor of its forbidden pair which would contradict the feasibility of \( \Pi' \) for the instance \((G = (V, E), s, t, C)\) of the \( PFP \) problem.

Finally, in the case that the length of the solution path is smaller than 3, it is easy to check the two remaining possibilities. When the path is direct from \( s \) to \( t \), the only forbidden pair of interest is \((s, t)\) which is trivial. When the path is of length 2, say \( \Pi = \{s, v, t\} \), we must also check that \((s, v)\) and \((v, t)\) are not forbidden pairs. All cases take a constant time to check.

As the proof follows a polynomial reduction from this problem to the Path with Forbidden Pairs problem (\( PFP \)), an immediate corollary follows.

**Corollary 4:** With \( H_1 \geq 1 \), the Path with Remaining Capacity problem (\( RC \)) is NP-complete.

For other considerations, we may also request that each path reserved is a shortest path. Using the result of Kann [5] on the shortest feasible PFP, it is easy to deduce the following corollary.

**Corollary 5:** With \( H_1 \geq 1 \), Shortest Path with Remaining Capacity problem (\( SPRC \)) remains NP-complete and is also hard to approximate (i.e., NPO PB-complete).

In the full paper [2], we introduce some variants of these optimization problems (e.g., Least Remaining Capacity, Total Consumed Capacity) and introduce some heuristics. We also show that a more accurate evaluation of the remaining bandwidth may be done on a per link basis. Nevertheless, the NP-completeness of the respective problem still holds.

### III. BANDWIDTH RESERVATION AND SLOT ALLOCATION

#### A. Slotted Model

In this section we assume as in [10] that the channel is slotted, i.e., time is divided in slots of equal length, equal to the length of information packets. Moreover, time slots are divided into frames of length \( L \). This channel may operate in parallel with a separate collision resolution channel. Transmissions that do not require reservation, as well as bandwidth reservation requests, take place on the collision resolution channel, while flows that require bandwidth reservation are transmitted in the slotted channel without collisions. There are several implementation issues concerning the previously described model, one of them being the manner in which slot synchronization can be maintained throughout the network. This can be accomplished by having a system like GPS issue synchronization signals. However, we do not dwell further into these issues, as our main objective at this point is to show how a proposed method of bandwidth reservation, coupled with a relatively simple distributed algorithm for reserving time slots, can ensure that all admitted flows receive their requested bandwidth (slots). Of course, the method does not guarantee that if an appropriate path exists it will always be found since this problem is NP-complete. The point is that using a simplified rule for bandwidth reservation, if a path is selected by the source node based only on partial knowledge of slot allocation at each node, then a slot schedule satisfying the bandwidth requirements of the flow can also be found in a distributed manner.

Assume that a new flow \( f \) arrives at network node \( s \), and it is decided to use path \( \Pi_f \) to transmit its packets to its destination \( t \). When we say that a flow is given a bandwidth of \( b_f \leq L \) units, we mean that some of the network nodes reserve slots so that all nodes along the path \( \Pi_f \) are able to transmit or receive \( b_f \) of flow \( f \)'s packets within each frame and that these transmissions can take place without interference, i.e., they can be safely delivered to the next hop (in the absence of errors due to channel noise).
In order to motivate our particular method of bandwidth reservation we consider first some examples. In Figure 1 (a) assume that the frame length is \( L = 2 \), \( H_I = 1 \) and that Rule A is used for bandwidth reservation. Let flow \( f_1 \) requiring 2 slots per frame be established between nodes \( h \) and \( g \). Thus nodes \( h \) and \( g \) fill-up all the available slots in a frame and announce zero available bandwidth. The same is true for node \( e \), since this node cannot receive packets during a frame, due to node \( h \)’s transmission. Assume next that a new flow request \( f_2 \) arrives at node \( a \) requiring again bandwidth 2, and with destination node \( e \). Since node \( e \) has available bandwidth 0 according to Rule A, flow \( f_2 \) cannot be established, as is indeed the case. However, rule A may overestimate the bandwidth requirements needed to establish a flow along a path, i.e., it may not take advantage of all possibilities of spatial reuse. Assume for example in Figure 1 (a), that flow \( f_1 \) has origin node \( g \) and destination node \( h \). Then node \( a \) will decide again that flow \( f_2 \) cannot be established. However, this time \( f_2 \) can be established since both nodes \( h \) and \( g \) will be receiving and therefore there will be no interference (recall that we assumed that \( H_I = 1 \)).

As another example consider the situation depicted in Figure 1 (b). Assume that \( L = 4 \), \( H_I = 1 \) and that flow \( f_1 \) arrives first with \( b_{f_1} = 1 \). Hence, one slot must be reserved on nodes \( i \) and \( k \), two slots on node \( j \) (one for reception and another one for transmission) and two slots on node \( c \) (to avoid interference during transmissions \( i \) to \( j \) and \( j \) to \( k \)). Assume that the first two slots in a frame are reserved on \( c \) to accommodate flow \( f_1 \). When flow \( f_2 \) arrives next, requesting bandwidth \( b_{f_2} = 1 \), node \( c \) must reserve two slots according to rule A i.e., it must reserve the last two slots in the frame. However, a schedule requiring node \( c \) to reserve only one slot is easy to construct: node \( a \) uses one of the first two slots for transmission from \( a \) to \( b \) and node \( b \) uses one of the last two slots, say slot 3, for transmission from \( b \) to \( c \). The possibility of reducing the reserved bandwidth on a node in the previous examples comes at the expense of requiring knowledge of exact allocations of frame slots at each node in the network. The amount of information exchange required to obtain this knowledge can be unacceptable in wireless networks.

Unfortunately, the use of Rule A to determine a path \( p_f \) for a new flow \( f \) with bandwidth requirements \( b_f \) may also be too optimistic. That is, Rule A does not always guarantee that there are \( b_f \) available slots in a frame at each node on the path \( p_f \) that can be dedicated for the correct transmission of flows \( f \) packets. To see this, consider the example in Figure 2. Assume that \( H_I = 1 \), and \( L = 2 \). Assume that flow \( f_1 \) arrives first with \( b_{f_1} = 1 \). Nodes \( a \) and \( b \) reserve the first slot in the frame for the transmission of flow \( f_1 \) packets and hence according to Rule I, nodes \( g \), \( a \), \( b \), \( c \), reserve one unit of bandwidth. Next flow \( f_2 \) with \( b_{f_2} = 1 \) arrives and nodes \( c \) and \( d \) reserve the second slot in the frame for transmission of flow \( f_2 \) packets. Hence nodes \( b \), \( c \), \( d \), \( e \), have one unit of available bandwidth. Assume now that flow \( f_3 \) with \( b_{f_3} = 1 \) arrives. According to Rule A, there is available bandwidth on path \((e,g)\) for flow \( f_3 \). However, no slot assignment can be found in a frame for this transmission: the first slot is prohibited because of transmission of flow \( f_1 \) packets and the second slot because of transmission of flow \( f_2 \) packets.

In our approach we would like to ensure that once a path \( p_f \) is chosen, the required slots for successful transmission can be found. This way, the likelihood that the connection setup request will fail is minimized. Hence, in order to avoid the last problem, (too optimistic rule) we change Rule A to the following.

**Rule B:** Whenever \( b_f \) packets per frame need to be transmitted between nodes \( i \) and \( j \), reserve \( b_f \) slots in a frame on all nodes in \( N_i(H_I + 1) \cup N_j(H_I + 1) \).

Next we show that Rule B, combined with local information that each node has about the use of frame slots by its neighbors in \( N_i(H_I + 1) \), guarantees the correct selection of paths for flow requests that arrive in the system.

Assume that node \( i \) knows the slots in a frame that are used for transmission or reception of packets by any of the nodes in \( N_i(H_I + 1) \). The available bandwidth \( A_i \) at node \( i \) is then
defined as the unused slots in the frame. If a request requiring
slots arrives and \( b_f \leq A_i \), then node \( i \) can reserve any \( b_f \) of
the \( A_i \) slots for transmission of flow \( f \) packets. This process
ensures the correct reception of these packets by the next hop
node, say node \( j \). To see this, note that \( N_j(H_T) \subseteq N_i(H_T+1) \).
Hence an unused frame slot at node \( i \) implies that no nodes
in \( N_i(H_T) \cup N_j(H_T) \) is using this slot for transmission or
reception. This implies that the transmission of a flow \( f \) packet
from \( i \) to \( j \) will take place without interference.

Nodes \( i \) and \( j \) must also inform their neighbors in \( N_i(H_T+1) \cup N_j(H_T+1) \) that they are reserving the particular \( b_f \) slots in
a frame for transmission and reception respectively. Note that
the actual number of slots reserved at a node with this process
may be smaller than \( b_f \). To see this, consider the example of
Figure 1(b) again. According to Rule B, after connection \( f_1 \) is
established, the remaining bandwidth of nodes \( b \) and \( c \) is 2, and
the used slots are slot 1 and 2 in the frame. When connection
\( f_2 \) is established, it will be calculated by the source node \( a \) that
the remaining bandwidth on nodes \( b \) and \( c \) is zero. However,
if node \( a \) picks slot 1 in the frame for transmission to node
\( b \), and slot 3 for transmission to node \( c \), then slot 4 will be
unused on both nodes \( b \) and \( c \). Hence the remaining bandwidth
on nodes \( b \) and \( c \) will be 1. Since we adopt a proactive scheme,
each node advertises periodically its remaining bandwidth to
the other nodes. Hence the nodes will be informed about the
extra remaining capacity of nodes \( b \) and \( c \) which implies that
spatial reuse gain may be obtain.

B. Connection Establishment Mechanism

We now describe a distributed mechanism for slot reserva-
tion using Rule B, that results in collision-free transmissions.
Node \( i \) in the network maintains a list \( R_i[k] \), \( k = 1, ..., L \). When \( R_i[k] = 0 \), then slot \( k \) in a frame can be used for a new
transmission or reception by node \( i \), or to avoid interference
with its neighbors in \( N_i(H_T) \). When \( R_i[k] > 0 \), then slot \( k \) is
unusable because this slot has been reserved for transmission
or reception of already established flows, or in order to avoid
interference with its neighbors in \( N_i(H_T) \). We call such a slot,
an “occupied” slot. Any flow that causes a slot to be occupied
is said to “occupy” the slot. Note that \( R_i[k] \) can be larger than
one in case the same slot is occupied to avoid interference with
more than one neighbors. How to keep track of this is explained
below. The “available bandwidth” \( A_i \) on node \( i \) is defined as number of unoccupied slots in a frame, i.e., the
number of slots \( k \) for which \( R_i[k] = 0 \).

In addition to \( R_i[k] \), node \( i \) keeps track of information for
each flow \( f \) that occupies some of the slots in a frame, in the
form of an array \( F_i[f,j,k] \), where \( F_i[f,j,k] = 1 \) if node
\( j \) requested slot \( k \) on \( i \) ’s frame to be occupied in order
to ensure the collision-free transmission of flow \( f \)’s packets.
Otherwise, \( F_i[f,j,k] = 0 \). Note that during the process of
bandwidth reservation more than one nodes may request slots
to be occupied by the same flow \( f \). For example, in Figure 1
(b), node \( e \) needs to reserve two slots for flow \( f_1 \), one for the
transmission of node \( i \) and one for the transmission of node \( j \). In this example, \( F_e(f_1,i,1) = 1 \), \( F_e(f_1,j,2) = 1 \).

We assume that each node knows the available bandwidth
of the nodes in the network. During the process of bandwidth
reservation, the nodes exchange the lists \( F_i[f,j,k] \) with their
\( H_T + 1 \)-hop neighbors.

1) Arrival of a New Flow:

a) New Flow Arrival at the Source Node: Suppose that
a new flow with bandwidth requirements \( b_f \) arrives at node \( s \)
from the outside world. Then node \( i \) does the following.

1) Attempts to find a path \( P_f \) that joins \( s \) to the destination
of the flow and satisfies the requirements of Rule B.
Specifically, Path Constraints: Let \( (i_0 = s, i_1, i_2, ..., i_M = d) \) be
the path nodes. The available bandwidth on any node \( j \in \bigcup_{m=0}^{M} N_{i_m}(H_T + 1) \) must be at least as large as the
quantity \( r_j \) determined by the following algorithm

\[
\begin{align*}
&\text{a) Initialize } r_j = 0 \text{ for all nodes in } \bigcup_{m=0}^{M} N_{i_m}(H_T + 1) \\
&\text{b) For each node } i_m, \quad m = 0, ..., M - 1, \text{ do} \\
&\quad i) \quad r_j \leftarrow r_j + b_f; \quad j \in N_{i_m}(H_T + 1) \cup N_{i_{m+1}}(H_T + 1); \\
&\quad \text{c) endo}
\end{align*}
\]

The quantities \( r_j \) are in effect the bandwidth that needs
to be reserved on the nodes in \( \bigcup_{m=0}^{M} N_{i_m}(H_T + 1) \) to
support a new flow that joins \( s \) to \( d \) according to Rule B, in order to guarantee interference-free
transmission.

2) If a path satisfying the Path Constraints cannot be found
then the flow is rejected. Else,

3) Node \( s \) picks \( b_f \) of slots \( k \) such that \( R_{s,k} = 0 \) and
makes them one. Moreover, it sets \( F_{s,f,s,k} = 1 \) if slot
\( k \) is one of the slots picked in this manner.

4) Node \( s \) sends a reservation request that includes
\( F_s[f,s,k] \) to all its neighbors in \( N_s(H_T + 1) \) and
forwards the path \( P_f \) to the next-hop node \( i_1 \).

b) Processing a New Reservation Request: Suppose that
a reservation request \( F_i[f,l,k] \) arrives to node \( j \) from a
neighbor \( i \). Then

1) If the elements \( F_i[f,l,k] = 1, ..., L \) are not already
set (the discussion that follows explains how this can happen), then

2) Node \( j \) sets \( F_j[f,l,k] = F_i[f,l,k], R_j[k] \leftarrow R_j[k] +
F_j[f,l,k], k = 1, ..., L \).

3) If node \( j \) is the next hop on the path \( P_f \), it does the
following

4) Node \( j \) sends a reservation request that includes
\( F_j[f,l,k] \) to all its neighbors in \( N_j(H_T + 1) \)

5) If node \( j \) is not the last node on the path, then it acts
as if the flow arrived from the outside world with the
exception that the flow path \( P_f \) is the one received by
node \( i \). That is

a) Node \( j \) picks \( b_f \) of slots \( k \) such that \( R_j[k] = 0 \) and
makes them one. Moreover, it sets \( F_j[f,j,k] = 1 \) if slot
\( k \) is one of the slots picked in this manner.

b) Node \( j \) sends a reservation request that includes
\( F_j[f,j,k] \) to all its neighbors in \( N_j(H_T + 1) \) and
forwards the path \( P_f \) to the next-hop.
The reservation in Step 4 ensures that all nodes in $H_I + 1$ neighborhood of the next-hop node reserve slots for the transmission of node $i$. The reservation in Step 5b ensures that all nodes in the $H_I$ neighborhood of node $j$ reserve slots for the transmission on node $j$ (if $j$ is not the last hop on the path). The combination of steps 4 and 5b may result in a node being charged twice for the transmission of the same node. For example, in Figure 1 (b), node $c$ must be charged only once for node $i$’s transmission. However, it will receive the same request twice: first when node $i$ sends the request in step 5b and second when node $j$ sends the reservation request at step 4. The check in Step 1 eliminates this possibility and makes sure that all nodes that belong to the intersection of the $H_I$ neighborhood of nodes $i$ and $j$ are charged only once.

2) Completion of Flow: When a flow $f$ completes, the established connection must be torn down and the reserved bandwidth must be released. This can be done either by having the sender initiate the process of connection tear-down, or by maintaining the connection in "soft state". In the latter case, which seems more appropriate for a wireless environment, refresh messages are sent while the connection is active, and timers are maintained at each of the intermediate nodes. After each refresh message the timer at a node is set to zero. If the value of the timer exceeds certain threshold, then the node considers that the connection completed and releases the reserved resources.

IV. APPLICATION TO MANET

The IETF standardization forum is addressing the problem of mobile ad hoc routing in the working group MANET. Several protocols have been proposed in experimental standards. There are two classes of routing protocols: the proactive protocols and the reactive protocols. The reactive protocols discover routes on demand and they do not seem suitable for the admission control we describe in this paper, since we need an a priori knowledge on the remaining bandwidth of each node in the network before the admission control process.

The OLSR protocol [7], [8] is well suited for the admission control because it contains an embedded broadcast mechanism that optimizes the number of retransmissions. The broadcast mechanism uses specific relay nodes called MultiPoint Relay nodes (MPR) [9]. There exist discussions in order to extend OLSR to QoS management (QOLSR) [6], but these previous works did not address the fundamental problem of link interferences. In the optimized version of OLSR protocol, nodes only advertise a subset of their adjacent links (i.e. their neighbor nodes). For the admission control to work properly, every node must know all the links between the other nodes in order to properly evaluate the radius of interference areas. We shall use F-OLSR, the version of OLSR where nodes advertise their full neighborhood, relayed by MPR. Nodes will advertise their remaining bandwidth in a specific control message, relayed by the MPR. We can call this kind of message, Remaining Bandwidth Advertisement (RBA). This message will be generated periodically. In order to compute their remaining bandwidth, nodes must know the status of the flows at nodes that are within $H_I + 1$ hops away. To this end every node can broadcast within $H_I + 1$ hops the identification of the incoming and outgoing flows as well as their bandwidth consumption. This information will be contained in a message called Flow Schedule Advertisement message (FSA). This message will be sent with TTL equal to $H_I + 1$ in order to avoid broadcasting beyond the $H_I + 1$ neighborhood of a node. This message can also be used for connection termination advertisement. With the knowledge of the remaining bandwidth of every node and the knowledge of the links of the network, every source node can perform the admission control described so far. OLSR protocol naturally maintains a routing table for best effort traffic. This routing table should not be affected by the QoS management. Instead admitted connections should be routed on the basis of source routing (packets are forced to be routed on the route mentioned in their headers). That way two different connections with the same source and destination may take different routes in order to avoid bandwidth depletion.

Acknowledgments. Part of this work was done while Bernard Mans was the INRIA-Hitachi 2003 Chair financed by Hitachi and hosted at INRIA-Rocquencourt, Domaine de Voluceau, 78153 Le Chesnay Cédex, France.

REFERENCES


