ARTICLE IN PRESS

Available online at www.sciencedirect.com
SCIENCE DIRECT®

Computer Networks xxx (2005) xxx–xxx

Computer Networks

www.elsevier.com/locate/comnet

Slot allocation schemes for delay sensitive traffic support in asynchronous wireless mesh networks

V. Vidhyashankar [a], B.S. Manoj [b], C. Siva Ram Murthy [c,*]

[a] *Department of Computer Science, Cornell University, Ithaca, NY 14853, USA*
[b] *Department of Electrical and Computer Engineering, University of California San Diego, San Diego, CA 92093, USA*
[c] *Department of Computer Science and Engineering, Indian Institute of Technology, Madras, Chennai 600036, India*

Received 17 October 2004; received in revised form 15 July 2005; accepted 26 September 2005

Responsible Editor: E. Gregori

Abstract

Heterogeneous multihop wireless networks such as wireless mesh networks consist of a set of resource-constrained mobile nodes that want to communicate with each other and a set of fixed relay nodes that are equipped with better resources and help in relaying data to the mobile nodes. Supporting real-time traffic in wireless mesh networks is considered as a challenging problem. In a synchronous or asynchronous slotted environment, the position of reservation slots at every link of an end-to-end path influences the end-to-end delay. The major contributions of this paper are the following: (i) an on-demand QoS routing protocol for asynchronous single channel wireless mesh networks, (ii) three heuristics for the slot allocation and positioning process in such networks, and (iii) the adaptations of slot allocation and positioning strategies for taking advantage of the recovery capacity effect of batteries of the mobile nodes. The heuristics we propose are the Early Fit Reservation (EFR), Minimum Bandwidth Reservation (MBR), and the Position-based Hybrid Reservation (PHR). The EFR heuristic reserves bandwidth at the first available slot on every link on a link-by-link basis in the forward path. The MBR heuristic allocates bandwidth on the links in the increasing order of bandwidth and the PHR heuristic assigns bandwidth for every link at a position which is proportional to its location in the path. Recent studies on chemical battery characteristics show that a pulsed current discharge can extend the battery life compared to a continuous discharge. We have also adapted the basic slot allocation strategies, that are aimed at maximizing system throughput, to take benefit of the pulsed discharge model. The adapted heuristics are designed to provide an extended battery life and hence reduce the number of battery recharges. Simulation studies show that EFR performs better in terms of delay characteristics, whereas PHR and MBR provide better system throughput in terms of call acceptance rate. The adapted heuristics are found to be performing better in terms of the number of *deaths* of mobile nodes and the average number of path breaks.

Keywords: Asynchronous ad hoc wireless networks; Time sensitive traffic; End-to-end reservation; Reservation slot positioning; Hybrid wireless mesh networks

* Corresponding author. Tel.: +91 44 2257 4361.
E-mail addresses: vidya@cs.cornell.edu (V. Vidhyashankar), bsmanoj@ucsd.edu (B.S. Manoj), murthy@iitm.ernet.in (C. Siva Ram Murthy).

1389-1286/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2005.09.032

1. Introduction

Hybrid wireless mesh networks are heterogeneous multihop wireless networks that have both fixed and mobile wireless relay nodes. The fixed relay nodes have better resources such as processing capability and battery power and they do not originate or terminate user traffic. The mobile nodes not only originate/terminate user traffic, also forwards data on behalf of other nodes. In addition to the problems added by the mobility, these mobile nodes are highly constrained in terms of computing and energy storage. Each mobile node in a hybrid wireless mesh network can communicate with neighboring nodes and they use multihop routing protocols along with wireless relaying to communicate with the nodes that are beyond their transmission range. Some of the fixed nodes may have connectivity to the external wired network and hence they may act as gateways to the Internet. Hybrid wireless networks have high potential to provide an economical alternative to the existing wide area wireless communication infrastructures. Supporting real-time traffic in an asynchronous hybrid wireless mesh network remains a tough task. Existing work assumes a TDMA or CDMA over TDMA models for handling the hidden terminals. In such cases, synchronization of nodes proved to be very expensive. The major advantages of wireless mesh networks are high data support, quick deployment, high scalability, high extendability and low cost per bit transferred. Fig. 1 shows an example hybrid wireless mesh network topology which also indicating a path between two nodes S and D. The fixed relay nodes merely forward data and can also serve as a gateway to the Internet.

Our work in this paper, involves development of a QoS routing protocol and slot allocation and positioning schemes for asynchronous hybrid wireless mesh networks. Since the mobile nodes are highly energy constrained in comparison to the fixed nodes, we extend the routing and slot allocation schemes to be more energy efficient by utilizing the pulsed battery discharge mechanism.

1.1. Why an asynchronous environment?

While a synchronous TDMA environment simplifies the channel allocation scheme in a multihop wireless network, it faces several challenges in a hybrid wireless mesh networks. We, in this paper, focus our solution on an asynchronous hybrid wireless mesh environment compared to a synchronous TDMA environment because of the following issues.

- *Cost of synchronization:* Environments like the TDMA require synchronization which is expensive in terms of both battery power and band-

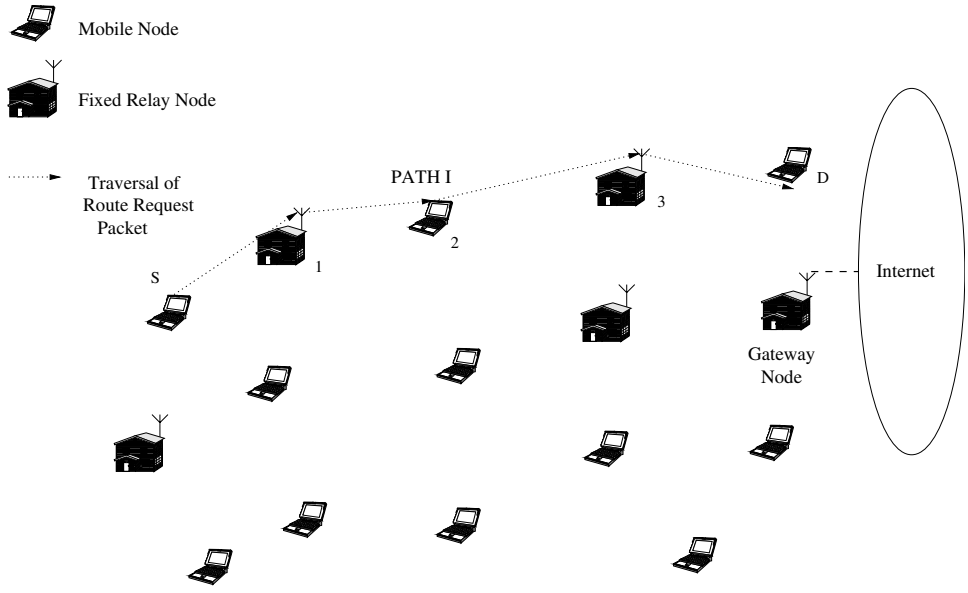

Fig. 1. An example hybrid wireless mesh network topology.

width and hence not attractive for a network with highly mobile nodes. This is because, in many practical situations, a Global Positioning System (GPS) may not always be available. Hence synchronization is achieved through broadcast of control packets which leads to flooding and hence is resource-wise expensive. On the other hand, asynchronous environments do not have to synchronize the entire network explicitly to a single clock.

- *Frequent partitioning and merging:* In an ad hoc network or hybrid wireless mesh environment, the network is subjected to frequent partitioning and merging, especially when the nodes are highly mobile. Consider an example of two network partitions A and B which are synchronized with respect to the time instants T_1 and T_2, respectively. When A and B merge, then the entire network may have to re-synchronize. Apart from the synchronization overhead, some calls are bound to be getting dropped due to collisions or synchronization differences during the synchronization period.

We have modified the DSR protocol [1] to enable it to support QoS routing with bandwidth reservation in asynchronous wireless mesh networks. The QoS path setup process has three major steps, viz., the *Bandwidth Feasibility Test Phase*, the *Bandwidth Allotment Phase*, and the *Bandwidth Reservation Phase*. During the *Bandwidth Feasibility Test Phase*, each node appends the bandwidth reservation information in the *Route Request* packets. During the *Bandwidth Allotment Phase*, the destination node runs a bandwidth allocation algorithm using the reservation information available in the *Route Request* packet. In the *Bandwidth Reservation Phase*, time slots allotted by the algorithm for each link are communicated to the participating nodes through the *Route Reply* packets. Reservation of each link is made using the ResvRTS-ResvCTS-ResvACK three-way handshake as proposed in [5]. This entire mechanism occurs in an asynchronous environment. By asynchronous environment, we mean that the entire network is not synchronized to a global clock as required in a TDMA system, for synchronizing the super-frame. Here a node uses relative time information to convey the exact positions of the reservation slots to a neighbor node. This is similar to the asynchronous MAC protocol proposed in [5]. We have proposed three heuristics for the slot allocation and positioning algorithm and studied their performance using simulations. The presence of heterogeneous wireless nodes within a hybrid wireless mesh network demands new solutions for both routing and bandwidth allocation. This is because, the protocol design can utilize the resource-rich fixed mobile nodes more often than the resource constrained mobile nodes. Therefore, we extended the routing protocol and slot allocation schemes to benefit the highly energy constrained mobile nodes.

This paper is organized as follows. Section 2 describes the related work in this area. Section 3 briefs the existing protocols used in this work. Section 4 describes the slot allocation framework and the heuristics. Section 5 presents results of the simulation of our scheme while Section 6 summarizes the paper.

2. Related work

To the best of our knowledge, no study has been done till date on slot allocation in an asynchronous environment. There exist a few solutions for the synchronous TDMA based environments, some of which are described in this section. Lin proposed a table-driven QoS routing mechanism in [2] and an on-demand call admission control scheme in [3]. The authors of [2] assume a TDMA environment with all nodes synchronized to a global clock. In each slot, different spreading codes are used for adjacent links for improving bandwidth reuse and reducing the effect of hidden terminals. The time frame consists of two distinct phases. The first is the control phase in which the network control functions can be performed in a distributed fashion. Each node takes turns to broadcast its information to its neighbors in a specified time slot. After the end of the control phase comes the data phase. This is the phase in which nodes vie with each other for sending data. The scheme in [2] reserves slots for real-time traffic using a QoS extension of the DSDV protocol. According to the protocol, slot information is stored in the routing table and periodically exchanged. This information is in the form of sets of free slots. The bandwidth reservation problem in a TDMA-based scheme has been shown to be equivalent to the satisfiability (SAT) problem which is known to be NP-complete [6]. Hence a heuristic has been proposed in [2] for assigning slots to each node. The path bandwidth for each of the nodes in the routing table is calculated. During call setup, the slot information is used to make call admission control. Thus, using this slot information, the system

can determine whether a call can be accepted at the beginning of the connection request. The slot assignment is performed at each node by forwarding reservation packets along the path. This is done by observing the common free slots in the adjacent nodes (this involves calculation of the path and the link bandwidths). This protocol, however, suffers from a high blocking probability.

The work in [3] proposes an application of this algorithm in an on-demand routing protocol. According to the protocol in [3], the path bandwidth is calculated at each node in the path when it receives the *Route Request* packet. It performs a feasibility test at every node in the path by comparing the status of its slots with the slot list that is present in the *Route Request* packet. If the node is an intermediate node and it passes the feasibility test, it appends its free slots to the *Route Request* packet and broadcasts it again. However, if it is the destination node and it has passed the feasibility test, then it sends a *Route Reply* packet back through the path. The nodes then reserve the slots when they receive the *Route Reply* packets.

The work in [4] describes a bandwidth allocation algorithm performed at the destination node. The scheme is implemented with an on-demand routing protocol. The source node broadcasts the *Route Request* packets. Each node that receives the *Route Request* packet appends its reservation information in the packet and broadcasts it. This goes on till the *Route Request* packet reaches the destination node. At the destination node, a bandwidth allocation algorithm allocates the slots to the links and sends them back through the *Route Reply* packets. The allocation algorithm selects the link with the least current bandwidth (i.e., the bandwidth reserved so far by the algorithm in each link) at every iteration. The protocol is proposed for a CDMA-over-TDMA based synchronous environment. The algorithm in [4] suffers from the way the links are chosen at every iteration of the algorithm, i.e., when links have the same minimum current bandwidth the tie is broken randomly. This may not produce good results all the time.

All these protocols, as mentioned before, did not have the need to address the issue of the hidden terminal problem as it had already been taken care of by the hardware mechanism. Moreover, the underlying assumption of synchronization exerts pressure on the scarce resources like bandwidth and battery power.

3. Reference protocols and models used in our work

In this section, we discuss the MAC and the routing protocols used in the implementation of the slot allocation strategies. The algorithms proposed in this work are applicable to both synchronous TDMA-based and asynchronous TDMA-based ad hoc wireless networks, wireless mesh networks, and hybrid wireless mesh networks. In order to provide the feasibility check and reservation of the bandwidth in the asynchronous environment discussed above, we require a MAC protocol which has the respective capabilities. This is the reason for selecting the Real Time MAC (RTMAC) protocol proposed in [5], a bandwidth efficient and asynchronous MAC protocol that supports both real-time (RT) and best-effort (BE) traffic.

3.1. The RTMAC protocol

In RTMAC, bandwidth is provided by dividing the transmission time into successive super-frames. The super-frame is implemented as a reservation table maintaining the time intervals during which nodes within its transmission range (including itself) are busy. The bandwidth reservations for RT traffic are made by reserving variable-length time slots (*conn-slots*) on the super-frames. The essential concept in this approach is the flexibility of slot placement in the super-frame. A set of consecutive reservation slots (each of which has a duration of twice the maximum propagation delay) is reserved by the pair of nodes involved in the RT session. The slot allocation differs from the TDMA scheme since no time synchronization is necessary for RTMAC [5] and all reservations are done with respect to the relative time period by which the RT session starts. This is illustrated in Fig. 2.

The reservation process shown in Fig. 2 is a three-way handshake. The ResvRTS is initiated by the sender, the ResvCTS is a means of acceptance of the suggested *conn-slot* by the receiver and an indication that the destination node has reserved the *conn-slot* while the ResvACK is needed to indicate that the bandwidth has been reserved at the sender's side. The hidden terminal effect is alleviated using the three-way handshake protocol as neighbors of the two nodes reserve the time interval of the RT session in their super-frame when they obtain the ResvACK or ResvCTS (as the case may be). Whenever the *conn-slot* suggested by the ResvRTS cannot be accepted by the MAC level des-

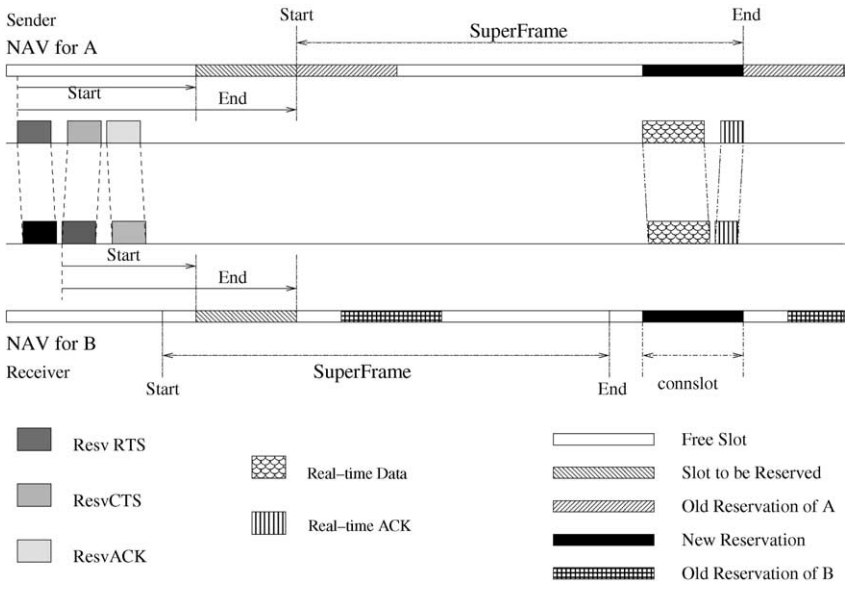

Fig. 2. Reservation mechanism used in RTMAC.

tination node, it searches for a feasible *conn-slot* to the sender through a ResvNCTS packet. The sender checks whether the *conn-slot* suggested is feasible to it. If not, the reservation fails at the MAC layer.

3.2. The DSR protocol

In our work, QoS routing has been implemented as an extension of the DSR protocol [1]. We have modified the protocol by piggybacking on the *Route Request* packets, the reservation information along with the relative time information for calculating the path bandwidth. According to the DSR protocol, the *Route Request* packets are initiated from the source node and each of those nodes which receives the *Route Request* packet checks if it is the destination node that the request has been addressed for. Otherwise, the *Route Request* packet is forwarded. The node's address is appended to the field indicating the traversed path in the *Route Request* packet. When a *Route Request* packet reaches the destination node, it responds back with a *Route Reply* packet which travels along the traversed path as indicated by the *Route Request* packet. We assume the existence of bidirectional links in the network. When the *Route Reply* packet reaches the source node, it then starts sending data through the route that was suggested. During path breaks, *Route Error* packets are sent on both the sides of the broken link till they reach the sender and the destination.

3.3. Power management model

Different battery models have been used to characterize the batteries of mobile nodes. Typical battery characteristics are specified in [8]. There are chiefly two models of battery discharge, viz., constant current discharge and pulsed current discharge [11]. The constant current discharge behavior of storage cells shows that the discharge curve is flat over a large time period and a gradual slope is developed as the voltage reaches the cut-off voltage (V_{cut}) specified by its manufacturer. In a pulsed current discharge, the battery is made to switch between short discharge periods and idle periods (rest periods). Fig. 3 shows the performance of the bipolar lead-acid battery subjected to pulsed discharge current of six current pulses obtained by the authors of [11]. While Fig. 3(a) shows the characteristics of the current density of the cell, Fig. 3(b) depicts the voltage characteristics for the pulsed discharge. After each discharge, which lasts for 3 ms, the cell is kept idle for 22 ms during which no discharging is allowed to take place. The cell is able to recover and revert back to its initial open circuit voltage during the first three rest periods. After the fourth current pulse the rest period of 22 ms turns out to be inadequate for the full recovery of the cell.

We aim to reap the benefits of the pulsed discharge model by strategically placing reservations so as to (possibly) enable recovery during the resulting idle periods. An exponentially decreasing func-

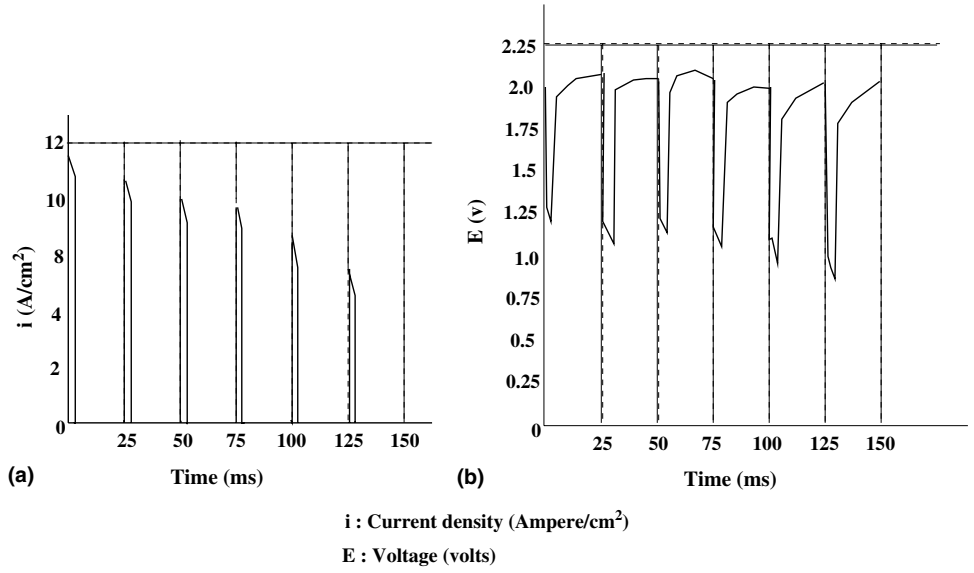

Fig. 3. Pulsed discharge characteristics of Lithium-Ion cell.

tion [11,12] depending on the current state (the discharge potential and the theoretical capacity) is used to model the recovery. Thus, we have adopted a recovery model for the mobile nodes by which the battery may recover when the node is idle. The battery is said to be in a *dormant state* either when its battery voltage has reached V_{cut} or when the theoretical capacity has been exhausted.

4. Our slot allocation framework

The wireless mesh network has been implemented in an asynchronous environment with QoS-DSR as the routing protocol. The RTMAC protocol [5] is used for effecting the reservations in the asynchronous environment. The QoS routing protocol has three phases of operation, viz., the *Bandwidth Feasibility Test Phase*, the *Bandwidth Allocation Phase*, and the *Bandwidth Reservation Phase*.

- *The Bandwidth Feasibility Test Phase:* In this phase, the selection of paths with the required bandwidth is performed. This is achieved during the *Route Request* propagation. Whenever a particular node receives a *Route Request* packet, it checks for bandwidth availability in the link through which it is received. If sufficient bandwidth is available, then the *Route Request* is forwarded, hence avoiding the paths which cannot support the bandwidth requirement. Before forwarding, the node synchronizes its reservation information with the already present information in the *Route Request* packet that it received. This is piggybacked on the new *Route Request* packet which is then broadcasted.
- *The Bandwidth Allocation Phase:* During this phase, the destination node performs a link-wise *conn-slot* (the time interval needed for a single RT session) allocation algorithm to assign the *conn-slots* at every link. The allotted *conn-slot* information is included in the *Route Reply* packet and sent to the source node using the path traversed by the *Route Request* packet.
- *The Bandwidth Reservation Phase:* In this phase, the reservation of bandwidth in the form of *conn-slots* at every intermediate link is carried out. If the allotted *conn-slot* is not currently free then any other free *conn-slot* can be reserved by the MAC layer. The details of the reservation failure is discussed in detail later in this section. This unavailability of the allotted *conn-slot* can be caused by the simultaneous allotment of the same *conn-slot* for a different flow during the period between the feasibility check and the actual reservation.

These phases will be discussed in detail later in this section. The mobile nodes in the mesh network have finite battery power. Since transmission involves maximum power consumption, we assume that the battery discharges only during transmis-

sion. The battery of the node may experience a charge recovery [11] when the node is idle and recharges whenever it is discharged fully. Once it is fully discharged, we assume that there is a short time period during which the node is unable to operate. The node is then said to have attained a *dormant state* or a *death* is said to have occurred. The fixed relay nodes, on the other hand are assumed to have infinite power as they have a supportive infrastructure.

4.1. The QoS-DSR Protocol

In this section, we present a detailed discussion on the routing protocol and the slot allocation schemes.

1. *The Bandwidth Feasibility Test Phase:* During this phase, the *Route Request* packets containing the reservation information are forwarded till the destination node is reached. Each node sends its reservation table along with the reservation frames of the links of the path covered so far. The reservation table of a node consists of the information about the reservations of its neighbors and itself. The *Route Request* carries the information stored in the reservation table which is in the form of reservation frames containing time durations. Each of the reservation frames belongs to one of the following states.

- *ASYNC_FREE:* This state is acquired when there is no transmission in the medium within the range of the node during that particular time duration.
- *ASYNC_UNRESV:* This corresponds to the unreservable state of the medium within the range of the node when it is involved in transmission (either as a sender or a receiver) during that particular time duration.
- *ASYNC_HT:* This corresponds to the unreservable state of the medium within the range of the node when one or more of its neighbors are involved in transmission during that particular time duration (HT in *ASYNC_HT* refers to Hidden Terminal).

The *Route Request* packet transmitted by an intermediate node contains the reservation frames of the nodes of the path traversed so far, in addition to the reservation table of the intermediate node. Whenever a node receives a *Route Request* packet, it uses the reservation information of the transmitter (of the *Route Request*) and its own to obtain a reservation frame for the link. Then, using the new reservation frame for the current link and the reservation frames of the previous two links (obtained by using the reservation frames of the last three nodes in the path traversed so far), the node checks whether a call can be admitted or not. If the call can be admitted, the node appends the reservation frame of the current link to the existing array and also inserts its reservation table to the new *Route Request* packet. A *Route Request* packet is forwarded by a node only if it satisfies the feasibility test. Along with the reservation information of the traversed nodes, the *Route Request* will additionally have information on whether the traversed nodes are mobile or fixed relay nodes represented by a boolean value. An intermediate node, before forwarding the *Route Request* packet, updates the reservation information contained in the *Route Request*.

The destination node waits for a certain amount of time decided by a parameter called *Route Request Window* and one of the *Route Request* packets collected during this time interval, is chosen for the next phase, viz., the Bandwidth Allocation Phase. The choice is made by the destination node by choosing the *Route Request* packet with the maximum number of fixed relay nodes in the path. A tie is broken by choosing the path with a lesser number of hops. If there is a tie again, the one with the greater bandwidth is chosen. Other *Route Request* packets that arrive beyond the *Route Request Window* are discarded. This path selection process is depicted in the example topology shown in Fig. 4. The figure shows three paths traversed by *Route Request* packets for establishment of a connection between nodes S and D. The three paths have the same number of fixed relay nodes, but Path I (S-1-2-3-D) is chosen as it has the least hop count [10].

2. *The QoS Frame:* The data structure used for the bandwidth allocation algorithm is called the QoS frame. This data structure is constructed just before the destination node runs the algorithm presented in Fig. 9. The reservation information of each link, provided in the *Route Request* packet, is used for this purpose. The definition of the data structure is useful in calculating the time complexities of the algorithms. It is a linear array of blocks, each block containing a time duration and the state to which it belongs. The state can be one of those tabulated in Fig. 5. The table has been constructed with respect to the directed link A–B as shown in the figure. The state of the link depends on whether

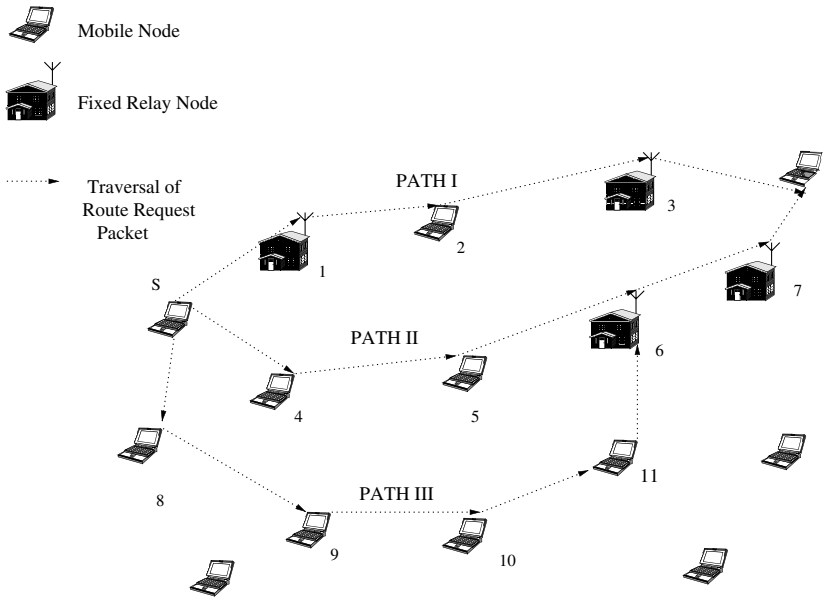

Fig. 4. A scheme for choosing the path.

Node A	Node B	State of the link
Free	Free	ASYNC_FREE_FREE
Free	Unreservable	ASYNC_FREE_UNRESV
Free	Hidden Terminal	ASYNC_FREE_HT
Unreservable	Free	ASYNC_UNRESV_FREE
Unreservable	Unreservable	ASYNC_UNRESV_UNRESV
Unreservable	Hidden Terminal	ASYNC_UNRESV_HT
Hidden Terminal	Free	ASYNC_HT_FREE
Hidden Terminal	Unreservable	ASYNC_HT_UNRESV
Hidden Terminal	Hidden Terminal	ASYNC_HT_HT

Fig. 5. States with respect to the directed link AB.

each of the concerned nodes is Free (the medium is free in its vicinity), Unreservable (when the medium is used by the node either as a sender or a receiver) or is a Hidden Terminal (when the node serves as a hidden terminal to a transmission involving one or more of its neighbors). It is to be noted that the states ASYNC_UNRESV_FREE and ASYNC_FREE_UNRESV are not possible in an ideal case. But they can arise either due to the unreliability of the medium or due to the transient nature of the network caused by the mobility of the nodes. Consider, for example, two nodes A and B trying to reserve the channel using the three-way handshake (at the MAC layer). Suppose, the ResvCTS sent by node B does not reach its neighbor node C. Before the data transmission between node A and node B starts, if node C tries to broadcast a *Route Request* packet for establishing some other connection, the reservation information stored in the *Route Request* packet by node C will not contain the reservation made by node B. When the *Route Request* packet reaches node B, it will store its reservation information that includes the recent most reservation with node A. This is one possible scenario in which the state ASYNC_FREE_UNRESV may arise.

3. *The Bandwidth Allocation Phase:* During this phase, the bandwidth allocation for the path is performed at the destination node. The input to the algorithm is the set of reservation frames constructed from the reservation information in the *Route Request* packet. These frames correspond to those of the links of the path. The basic underlying principle in the algorithm is that the bandwidth of a link is reserved by considering its frame along with the two levels of neighboring links on both the sides of this link. This is done for every link. The algorithm tries to avoid overlapping of reservation in neighboring frames at the first and at the second level. By doing this, we try to alleviate the hidden terminal problem and allot the necessary bandwidth at the same time. The heuristics, in general, vary with respect to the selection of the link at every iteration of the algorithm. We note at this point, that

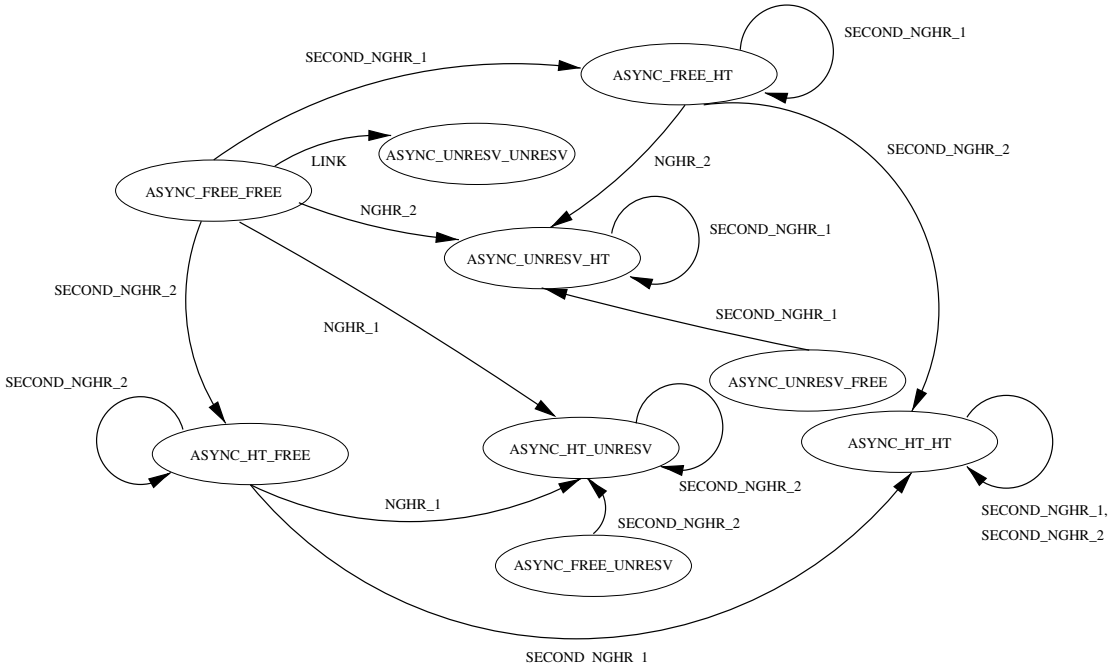

Fig. 6. State diagram describing the UpdateState function.

there is a maximum error equal to the propagation delay whenever the reservation information of a node is synchronized with its neighbor. The maximum value of propagation delay is 1 μs when the transmission range is 300 m. This is much smaller than the guard band interval that is included in the bandwidth that is allocated. This guard band on both the sides of the *conn-slot* helps in cushioning the error incurred in synchronization. The algorithm returns the time durations allocated for each link if it is successful. If the algorithm is unsuccessful, it returns NULL.

The function *Resv_Bandwidth* presented in Fig. 9 contains the generic bandwidth allocation algo-

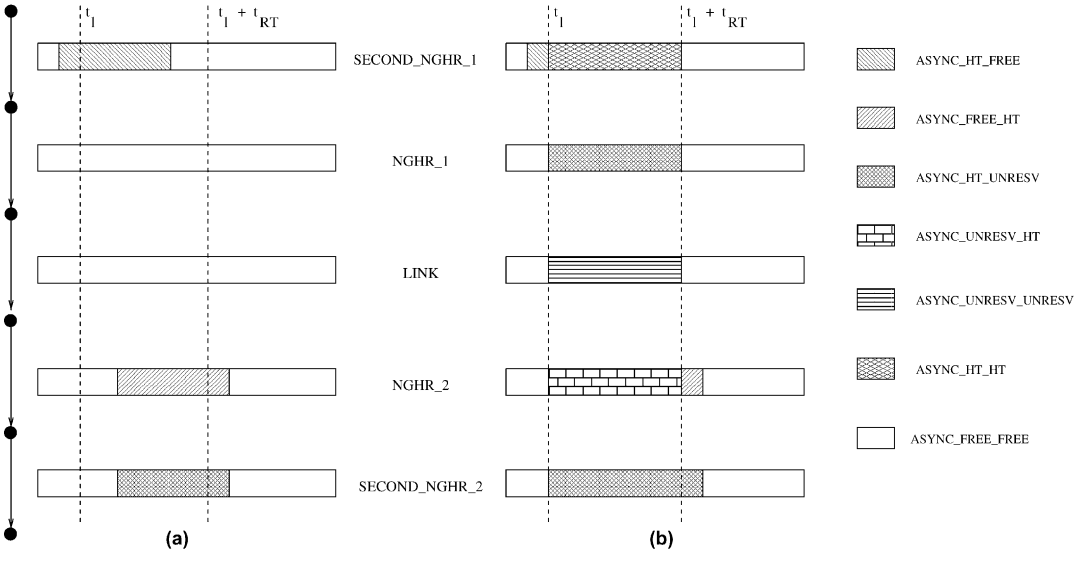

Fig. 7. A sample execution of the UpdateState function.

GetAvailable_Bandwidth_Info(QoSFrame P, QoSFrame Q, States typeOfIntersection)

1) *Let* $t_k \leftarrow 0$, $t_{prev} \leftarrow 0$.
   //Let State(Q,(t_k, t_l)*) return the state of the frame Q in the time interval* (t_k, t_l).
   //Let SF denote the total super frame time.
   //Let R denote the output frame.
2) *Find the minimum interval* (t_k, t_l) *such that in none of the frames P and Q is there a state transition.*
3) *switch(typeOfIntersection)*
   a) *case FIRST_TWO :*
      If ((State (P,(t_k, t_l)*)) IN*
      {ASYNC_FREE_FREE, ASYNC_FREE_HT, ASYNC_HT_FREE, ASYNC_HT_HT, ASYNC_UNRESV_FREE, ASYNC_UNRESV_HT}) ∧
      (State (Q,(t_k, t_l)*)) IN*
      {ASYNC_FREE_FREE, ASYNC_HT_FREE})) then
      SetState(R, (t_k, t_l)*, ASYNC_FREE_FREE)*
      else SetState(R, (t_k, t_l)*, ASYNC_UNRESV_UNRESV).*
   b) *case LAST_TWO :*
      If ((State(Q,(t_k, t_l)*)) IN*
      {ASYNC_FREE_FREE, ASYNC_FREE_HT, ASYNC_HT_FREE, ASYNC_HT_HT, ASYNC_FREE_UNRESV, ASYNC_HT_UNRESV}) ∧
      (State(P,(t_k, t_l)*)) IN*
      {ASYNC_FREE_FREE, ASYNC_FREE_HT}) then
      SetState(R, (t_k, t_l)*, ASYNC_FREE_FREE)*
      else SetState(R, (t_k, t_l)*, ASYNC_UNRESV_UNRESV).*
   c) *case MIDDLE_TWO :*
      If ((State(Q,(t_k, t_l)*)) = ASYNC_FREE_FREE)*
      ∧ (State(P,(t_k, t_l)*)) = ASYNC_FREE_FREE)*
      then SetState(R, (t_k, t_l)*, ASYNC_FREE_FREE)*
      else SetState(R, (t_k, t_l)*, ASYNC_UNRESV_UNRESV).*
4) *If State(R,*(t_k, t_l)*)=State(R,*(t_{prev}, t_k)*) then Merge(R,*(t_{prev}, t_k)*,*(t_k, t_l)*)*
   else $t_{prev} \leftarrow t_k$.
   //The Merge function merges the previous duration (t_{prev}, t_k) *and the current*
   //duration (t_k, t_l) *into one* (t_{prev}, t_k) *with the same state.*
5) $t_k \leftarrow t_l$.
6) *If* $t_l <$ *SF then goto Step 2.*
7) *Return R.*

Fig. 8. Algorithm to obtain free bandwidth.

rithm. Four invocations of the *GetAvailable_Bandwidth_Info* function which is presented in Fig. 8 finally produces the frame *ResultantFrame* which contains the free time durations during which a *conn-slot* may be reserved.

The algorithm tries to avoid overlapping of reservation in neighboring frames at the first and at the second level. Steps 2 $(e - h)$ determine the time durations in which the *conn-slot* can be reserved. The Steps 1, 2(i), and 2(j).(iii) depend on the heuristic used.

The UpdateState function used in the algorithm shown in Fig. 9 can be understood from the state diagram in Fig. 6. The states correspond to those of each time instant in the QoS frame of each link. Hence, all states shown in the diagram qualify to be initial states (however, the function cannot be called when the state for the time instant is ASYNC_UNRESV_UNRESV). The transitions are labeled, each label referring to the nature of the link, viz., LINK (the link for which the reservation is being made), NGHR_1 and NGHR_2 (the first neighboring links), and SECOND_NGHR_1 and SECOND_NGHR_2 (the second neighboring links). A state transition occurs only for those time instants for which the UpdateState function has been called, also depending on the current nature of the link specified as a parameter of the function. A sample execution of the UpdateState function is shown in Fig. 7. Fig. 7(a) shows the frames before the state transition while Fig. 7(b) shows them after the state transition.

(i) *Early Fit Reservation (EFR):* The EFR heuristic shown in Fig. 10 assigns bandwidth link-by-link starting from the sender to the receiver. At every link, the EFR tries to allocate the first available free *conn-slot* after the reserved *conn-slot* on the previous link. The iterations are performed starting from the first link (that involves the source node), then proceeding with the successive links of the path. This heuristic tries to reduce the end-to-end delay of data delivery.

(ii) *Minimum Bandwidth-based Reservation (MBR):* The MBR heuristic shown in Fig. 11 allocates bandwidth to the links in the increasing order of free *conn-slots*, i.e.,

The Generic Algorithm
Resv_Bandwidth(QoSFrames, pathLength, t_{RT})

1) *$Q_i \leftarrow$ SelectQoSFrame(QoSFrames, pathLength).*
   //the i^{th} frame in the path. This selection varies depending on the
   //heuristic employed.
2) *While ($Q_i \neq \phi \wedge Q_i$ not reserved) do*
   a) *$Q_{i-2} \leftarrow$ SelectSecondPrevFrame(i, QoSFrames).*
      //If there does not exist such a frame, then the function returns
      //a QoS frame with the entire time duration having the state
      //ASYNC_FREE_FREE.
   b) *$Q_{i-1} \leftarrow$ SelectPrevFrame (i, QoSFrames).*
   c) *$Q_{i+1} \leftarrow$ SelectNextFrame (i, QoSFrames).*
   d) *$Q_{i+2} \leftarrow$ SelectSecondNextFrame (i, QoSFrames).*
   e) *FirstTwo $\leftarrow$ GetAvailable_Bandwidth_Info (Q_{i-2}, Q_{i-1}, FIRST_TWO).*
   f) *LastTwo $\leftarrow$ GetAvailable_Bandwidth_Info (Q_{i+1}, Q_{i+2}, LAST_TWO).*
   g) *FirstAndLastTwo $\leftarrow$ GetAvailable_Bandwidth_Info (FirstTwo, LastTwo, MIDDLE_TWO).*
   h) *ResultantFrame $\leftarrow$ GetAvailable_Bandwidth_Info (Q_i, FirstAndLastTwo, MIDDLE_TWO).*
   i) *Select a time interval (t_l,$t_l + t_{RT}$) having the state ASYNC_FREE_FREE in the frame ResultantFrame. Again this selection varies based on the heuristic.*
   j) *If such an interval exists,*
      (i) *Resv[i] $\leftarrow t_l$.*
      (ii) *UpdateState(Q_i, t_l, $t_l + t_{RT}$, SECOND_NGHR_1).*
      //The UpdateState function updates the state for the given
      //duration as specified in the state diagram shown in Figure 5.
      UpdateState(Q_{i+1}, t_l, $t_l + t_{RT}$, NGHR_1).
      UpdateState(Q_{i-1}, t_l, $t_l + t_{RT}$, LINK).
      UpdateState(Q_{i+2}, t_l, $t_l + t_{RT}$, NGHR_2).
      UpdateState(Q_{i-2}, t_l, $t_l + t_{RT}$, SECOND_NGHR_2).
      (iii) *Select the next link depending on the heuristic.*
      else
      (i) *Break.*
      (ii) *Return ϕ. $//Unsuccessful\ Reservation$.*
   End of while loop.
3) *Return Resv.*
4) *End.*

Fig. 9. The generic algorithm.

the link with the least free bandwidth is considered first. At every link, MBR allocates the first free *conn-slot* available. At each iteration, the frame that has the minimum reservable bandwidth will be taken into consideration. Thus we are trying to improve the probability with which the algorithm is successful. The reservation for each link is done by traversing from the beginning of the frame.

The EFR Heuristic
EFR_Resv_Bandwidth(QoSFrames, pathLength, t_{RT})

1) *$i \leftarrow 1$; prevTime $\leftarrow 0$.*
2) *While ($i \leq$ pathLength) do*
   a) *$Q_{i-2} \leftarrow$ SelectSecondPrevFrame(i, QoSFrames).*
   b) *$Q_{i-1} \leftarrow$ SelectPrevFrame(i, QoSFrames).*
   c) *$Q_{i+1} \leftarrow$ SelectNextFrame(i, QoSFrames).*
   d) *$Q_{i+2} \leftarrow$ SelectSecondNextFrame(i, QoSFrames).*
   e) *Let R be the final resultant frame after calling GetAvailable_Bandwidth_Info as specified in the generic algorithm.*
   f) *Select a conn-slot (t_l, $t_l + t_{RT}$) in the frame R, having the state ASYNC_FREE_FREE. This selection is made by starting the search from prevTime in the frame.*
   g) *If such an interval exists,*
      i) *Resv[i] $\leftarrow t_l$.*
      ii) *Update the states of the frames as mentioned in the generic algorithm.*
      iii) *i $\leftarrow$ i+1.*
      iv) *prevState $\leftarrow t_l + t_{RT}$.*
      else
      i) *Break.*
      ii) *Return ϕ. $//Unsuccessful\ Reservation$*
   End of while loop.
3) *Return Resv.*
4) *End.*

Fig. 10. The EFR heuristic.

The MBR Heuristic
MBR_Resv_Bandwidth(QoSFrames, pathLength, t_{RT})

1) *Let Q_i be the frame which has the minimum reservable bandwidth.*
2) *While ($\exists$ Q_i yet to be reserved) do*
   a) Q_{i-2} $\leftarrow$ *SelectSecondPrevFrame(i, QoSFrames).*
   b) Q_{i-1} $\leftarrow$ *SelectPrevFrame(i, QoSFrames).*
   c) Q_{i+1} $\leftarrow$ *SelectNextFrame(i, QoSFrames).*
   d) Q_{i+2} $\leftarrow$ *SelectSecondNextFrame(i, QoSFrames).*
   e) *Let R be the final resultant frame after calling GetAvailable_Bandwidth_Info as specified in the generic algorithm.*
   f) *Select a conn-slot (t_l, $t_l + t_{RT}$) in the frame R, having the state ASYNC_FREE_FREE. This selection is made by starting the search from the beginning of the frame.*
   g) *If such an interval exists*
      (i) *Resv[i]* $\leftarrow$ t_l.
      (ii) *Update the states of the frames as mentioned in the generic algorithm.*
      (iii) *Select the link Q_i from the rest of the unreserved frames having the least reservable bandwidth. In case of a tie, take the frame that has the lesser index.*

      else
      i) *Break.*
      ii) *Return* ϕ. $//Unsuccessful\ Reservation$

   End of while loop.
3) *Return Resv.*
4) *End.*

Fig. 11. The MBR heuristic.

(iii) *Position-based Hybrid Reservation (PHR):* The PHR heuristic shown in Fig. 12 assigns bandwidth slot for every link the position of which is proportional to the position of the link in the path. The operation of this heuristic is similar to the previous one. The difference lies in the selection of the *conn-slot* at a particular position for reservation at each link. The link with the least free bandwidth is selected for each iteration. The reservation slot is positioned for each link depending on its position in the path.

We observe that this heuristic tries to reap the benefits of both the EFR and the MBR heuristics. Thus positioning the *conn-slot* depending on the value of *posn* (see Fig. 12) will result in the earlier links in the path reserve (with a greater probability) in the beginning of the frame and the later links in the path, reserve towards the end. Hence this type

The PHR Heuristic
PHR_Resv_Bandwidth(QoSFrames, pathLength, t_{RT})

1) *Let Q_i be the frame with the minimum reservable bandwidth.*
2) *While ($\exists$ Q_i yet to be reserved) do*
   a) Q_{i-2} $\leftarrow$ *SelectSecondPrevFrame(i, QoSFrames).*
   b) Q_{i-1} $\leftarrow$ *SelectPrevFrame(i, QoSFrames).*
   c) Q_{i+1} $\leftarrow$ *SelectNextFrame(i, QoSFrames).*
   d) Q_{i+2} $\leftarrow$ *SelectSecondNextFrame(i, QoSFrames).*
   e) *Let R be the final resultant frame after calling GetAvailable_Bandwidth_Info as specified in the generic algorithm.*
   f) *Select a time interval (t_l, $t_l + t_{RT}$) in the frame R, having the state ASYNC_FREE_FREE. This selection is made according to the conditions given below.*
      i) *posn* $\leftarrow$ $\frac{i}{pathLength}$.
      ii) *Let SF denote the total super-frame time.*
      iii) *Search the frame from the time instant (posn . SF) till you reach a time interval (t_l,$t_l + t_{RT}$), say, in which an RT session can be fitted.*
   g) *If such an interval exists*
      i) *Resv[i]* $\leftarrow$ t_l.
      ii) *Update the states of the frames as mentioned in the generic algorithm.*
      iii) *Select the link Q_i from the rest of the unreserved frames having the least reservable bandwidth. In case of a tie, take the frame that has the lesser index.*

      else
      i) *Break.*
      ii) *Return* ϕ. $//Unsuccessful\ Reservation$

   End of while loop.
3) *Return Resv.*
4) *End.*

Fig. 12. The PHR heuristic.

of reservation yields a lesser delay than the MBR heuristic.

To adapt these heuristics to hybrid wireless mesh networks, we have modified them to the EFR-Mesh, MBR-Mesh, and the PHR-Mesh. These adapted heuristics take into consideration the power-constrained nature of the mobile relay nodes. The main objective behind the adapted heuristics is to allocate a free time slot (hole) in a link just before a *conn-slot* during the bandwidth allocation algorithm, whenever the sender node of the link is a mobile relay node. In other words, when a *conn-slot* has to be allocated to a link, the nature of the sender node is verified. If it is a mobile relay node, a *conn-slot* is assigned such that there is a hole just before it. This, artificially created bandwidth hole is termed *recovery-hole*. On the other hand, if the sender node of the link is a fixed relay node, the *recovery-hole* is not allocated. The *recovery-hole* can possibly useful to the mobile node for its battery to recover. Only the sender node is checked since the power management has been addressed only with respect to transmission in our model. Hence, the adaptive heuristics try to reduce the possibility of the node reaching a *dormant state* at the cost of fragmenting the reservation frame. An illustration of the inclusion of a *recovery-hole* in the bandwidth allocation strategy when EFR-Mesh is used in comparison with the EFR strategy is shown in Fig. 13. It compares the NAVs of the node B after call establishment when EFR and EFR-Mesh are executed.

4. *The Bandwidth Reservation Phase:* If the bandwidth allocation algorithm is successful in its reservation, the bandwidth allocation information will be attached to the *Route Reply* packet and sent to the previous node in the path. When a node receives a *Route Reply* packet, it then checks whether it is in the path mentioned in the packet. If so, it reads the time allotted for the link between itself and the node which had sent the *Route Reply*. The node then checks from its super-frame whether it is free during the time interval suggested. If so, the call setup takes place at the MAC layer using the RTMAC. Accordingly there is a three-way handshake taking place between the two nodes. If this three-way handshake is successful, then the call gets established. The *Route Reply* packet is then sent to the next node. This goes on till the entire path reservation is completed.

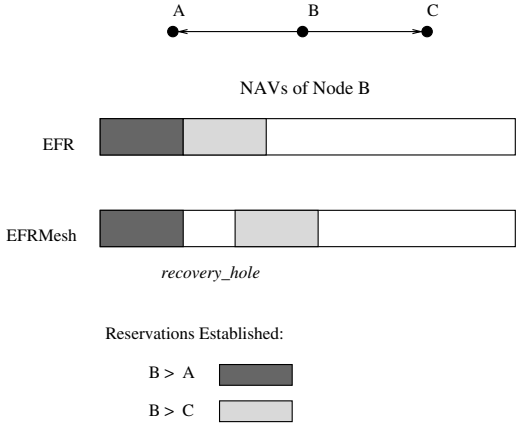

Fig. 13. Recovery slot allocation in EFR and EFR-Mesh.

5. *Handling Bandwidth Reservation Failure:* Due to dynamic nature of the network and the temporal overlap of multiple connection setup phases, it is likely that, at a link, a bandwidth slot allotted during the *Bandwidth Allocation Phase* may be unavailable during the *Bandwidth Reservation Phase*. This is referred to as the *Reservation Failure* and it can be handled in the following two ways: (i) the end-to-end reservation failure handling and (ii) the local reservation failure handling. According to the end-to-end reservation failure handling mechanism, the intermediate node that detects the *Reservation Failure* stops proceeding with the *Bandwidth Reservation Phase* and initiate *ResvFailure* packets to the source and the destination of the connection. Upon reception of the *ResvFailure* packet, nodes in the downstream (nodes on the path from the node that originate the *ResvFailure* packet to the destination of the connection) release their reservations and cache entries made for that connection. The upstream nodes (nodes on the path from the node that originate the *ResvFailure* packet to the source of the path) prepare for releasing any cache entries made for the connection. The source of the path will then initiate a new reservation process. In the local reservation failure handling scheme, the node that detects a *Reservation Failure* can locally allot another free slot instead of the originally reserved one. We, in this work focused more on the performance of the slot allocation and positioning scheme and therefore used the end-to-end reservation failure handling mechanism.

5. Simulation results

The proposed QoS-DSR routing protocol and the heuristics for reservation slot positioning and allocation for the asynchronous hybrid wireless net-

work environment were simulated using GloMoSim [7]. The heuristics were compared under identical environments and loads. We have used Constant Bit Rate (CBR) sessions which generate datagram packets of size 216 bytes, every 60 ms, for both RT (Real Time) and BE (Best Effort) connections. We have used the radio capture model and the free-space propagation model throughout the simulation. Simulation time was taken as 600 s and CBR sessions of 200 s were generated randomly between 0 and 500 s of the simulation. The physical layer bandwidth for the wireless network interface is taken as 1 Mbps. This value for the physical layer bandwidth is taken to focus the study at high traffic load situations. The CBR sessions were classified into two classes, namely RT and BE sessions. Terrain range used for the simulations was 1000 m × 1000 m. Transmission range of a node was taken to be 300 m. The simulated network has 30 nodes which were uniformly distributed in the terrain area. Random way-point mobility model is used. As mentioned before, an exponential distribution is used to model the recovery and the value of V_{cut} follows that of [9]. The discharge and the recharge rates are assumed to be constant and the latter is assumed to be ten times the former. The first experiment is a performance comparison of the EFR, MBR, and PHR heuristics with increasing network load. Fig. 14 compares the average end-to-end delay performance for each of the three heuristics. Mobility is fixed at 20 m/s during this experiment. The EFR gives the least delay while the MBR yields the maximum delay and the PHR scheme provides an intermediate level of performance. The reason for the performance of EFR can be attributed to the manner in which *conn-slots* are assigned. In the EFR, we choose the first available *conn-slots* on successive links from source to destination. Hence the *conn-slots* are assigned in an early-fit manner. Hence it is bound to give the least delay when compared to other heuristics. We estimated the end-to-end delay jitter as the standard deviation of the end-to-end delay. The average jitter compared in Fig. 15 also shows a similar trend as that of the end-to-end delay. The jitter performance showed an increasing trend with increase in the number of connections in the network. This is attributed to the increased congestion in the network and the high mobility of nodes. Mobility, which is set at 20 m/s for this experiment, is also found to be contributing to the end-to-end jitter. Fig. 16 compares the throughput achieved by the three heuristics, measured in terms of packet delivery ratio. Packet delivery ratio performance shows that the MBR and the PHR schemes provide a much greater throughput than the EFR scheme.

In our experiments, we also conducted simulation experiments to compare the performance of EFR, PHR, and MBR to their variations, adapted for hybrid wireless mesh networks called EFR-Mesh, PHR-Mesh, and MBR-Mesh, respectively. An important parameter that has been introduced in this set of results is the number of *deaths* of a mobile node. This value is equal to the number of times a node reaches the *dormant state*.

The first set of simulations for comparing EFR and EFR-Mesh has been carried out with increasing

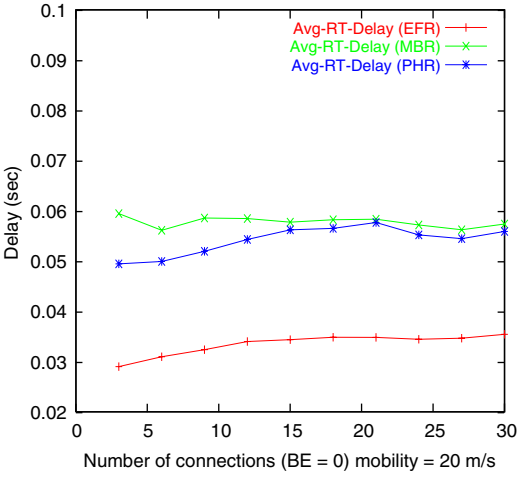

Fig. 14. Average end-to-end delay vs. network load (mobility = 20 m/s).

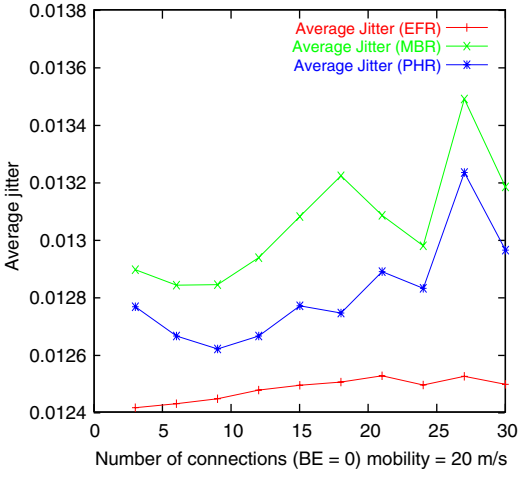

Fig. 15. Average jitter vs. network load (mobility = 20 m/s).

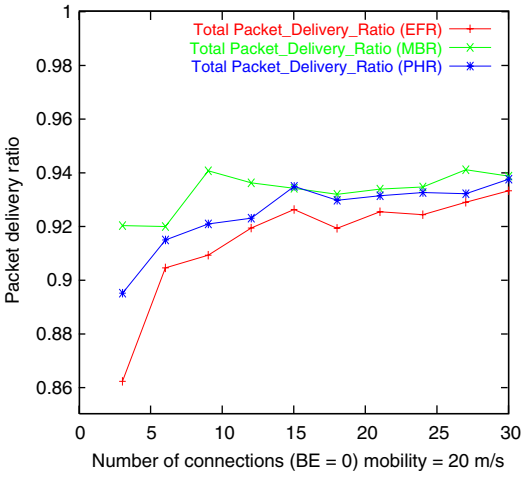

Fig. 16. Packet delivery ratio vs. network load (mobility = 20 m/s).

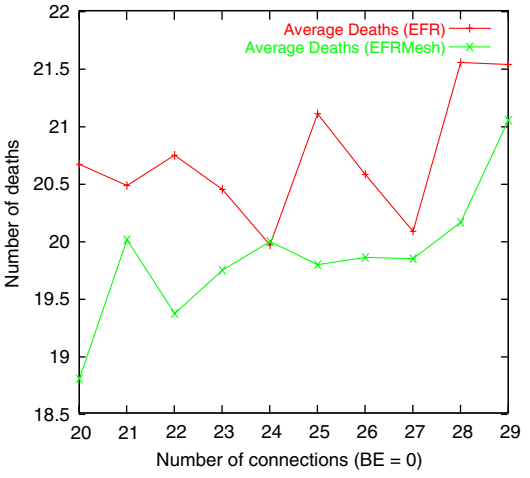

Fig. 18. Average number of *deaths* vs. network load (mobility = 20 m/s).

RT traffic load without any BE traffic. Fig. 17 compares the average end-to-end RT delay between EFR and EFR-Mesh. The end-to-end RT delay performance graph clearly shows that the EFR-Mesh suffers due to the allocation of *recovery-holes*. This is mainly because, the allocation of *recovery-holes* resulting in an increased delay, that is proportional to the length of the *recovery-hole*, at every intermediate mobile relay node. Fig. 18 compares the average number of *deaths* per node between EFR and EFR-Mesh schemes. The EFR-Mesh heuristic performs better than EFR heuristic owing to the *recovery-hole* allocated for every mobile relay node in order to enable their batteries to recover the charge.

Until now we discussed experiments with no background BE traffic and in order to study the performance behavior of the slot allocation and positioning heuristics with the presence of BE traffic, we carried out more experiments as described further in this section.

We have simulated the heuristics under a background BE traffic load of 10 BE calls. Fig. 19 compares the average RT end-to-end delay experienced by PHR and PHR-Mesh when background BE traffic is present. We noticed a moderate increase of about 10% in the delay of both PHR and PHR-Mesh compared to the case where no BE traffic is present. The delay difference between PHR and

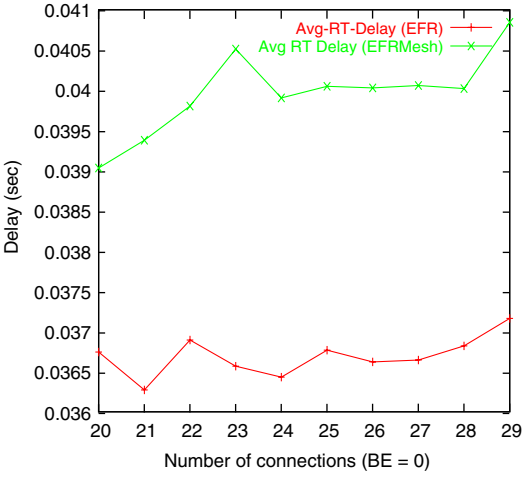

Fig. 17. Average end-to-end delay vs. network load (mobility = 20 m/s).

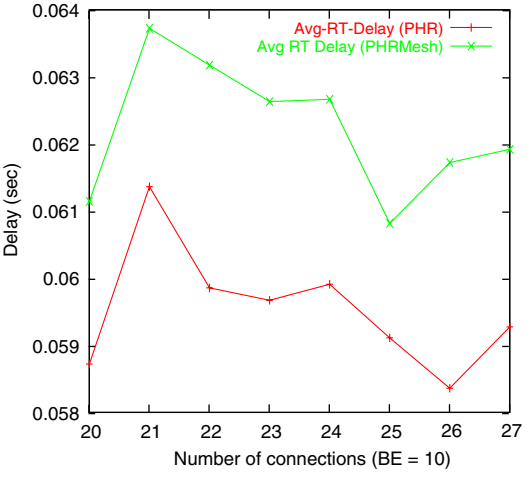

Fig. 19. Average RT end-to-end delay vs. network load (mobility = 20 m/s).

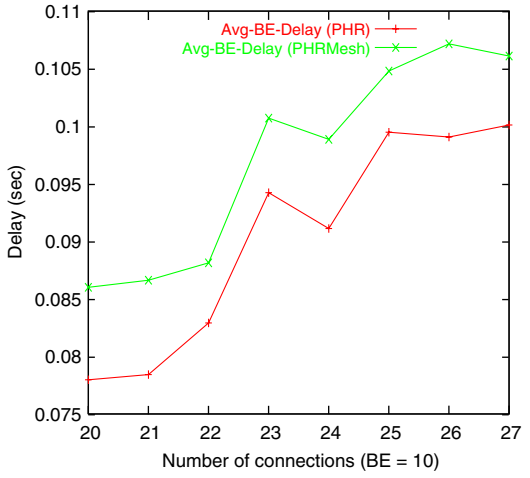

Fig. 20. Average BE end-to-end delay vs. network load (mobility = 20 m/s).

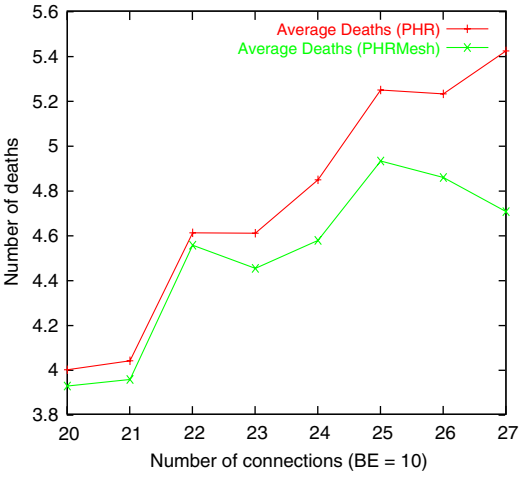

Fig. 21. Average number of *deaths* vs. network load (mobility = 20 m/s).

PHR-Mesh remains almost constant. We also studied the variation of the end-to-end delay experienced by BE traffic flows when PHR or PHR-Mesh heuristics are used for the RT traffic. It was interesting, in Fig. 20, to note that while the end-to-end delay for RT traffic is almost remaining constant with load, the end-to-end delay for BE traffic is found to be increasing rapidly with number of RT connections. This also shows that the bandwidth reservation for high priority RT traffic can indeed provide a much higher quality of service in comparison to the BE traffic. In addition to the performance improvement for the RT traffic, our scheme can support both BE traffic and RT traffic simultaneously. The call blocking ratios are compared between EFR and EFR-Mesh with background BE traffic load in Fig. 22. The graph shows that, on an average, the overall call blocking ratio increasing rapidly when background traffic is present. This is because, the control packets for path finding and bandwidth reservation experiences high contention from the background BE traffic. Also, we noted a slight increase of call blocking ratio for the PHR-Mesh heuristics. This is mainly due to the presence of fractured bandwidth in terms of *recovery-holes* that reduces the probability of call acceptance (see Fig. 21).

We now present the performance of the heuristics with varying mobility. For this experiment, we kept the number of RT and BE connections at 15 and 10, respectively. The mobility of nodes is varied from 2 to 20 m/s. Each sample point is averaged over 10 runs. We noticed that the trends for mobility are

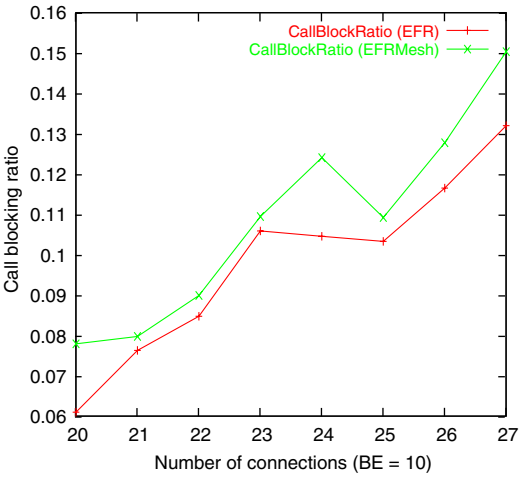

Fig. 22. Average call blocking ratio vs. network load (mobility = 20 m/s).

the same for all the three heuristics. Fig. 23 shows that the number of path breaks experienced per connection increases with mobility. This not only increases the control packets but also influences the call dropping rate. The PHR-Mesh heuristics performs slightly better than the PHR heuristics. This is attributed to the fact that due to the fact that the call acceptance rate for PHR-Mesh is lower than the PHR heuristics. Fig. 24 shows that the call dropping ratios in PHR and PHR-Mesh increase with increasing mobility, with the PHR-Mesh heuristic exhibiting slightly results. This increase in call dropping ratio is mainly attributed to the rapid change in the topology where paths get broken quickly and

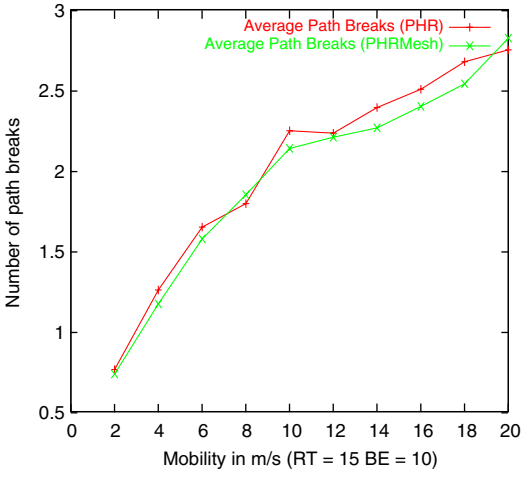

Fig. 23. Average number of path breaks vs. mobility.

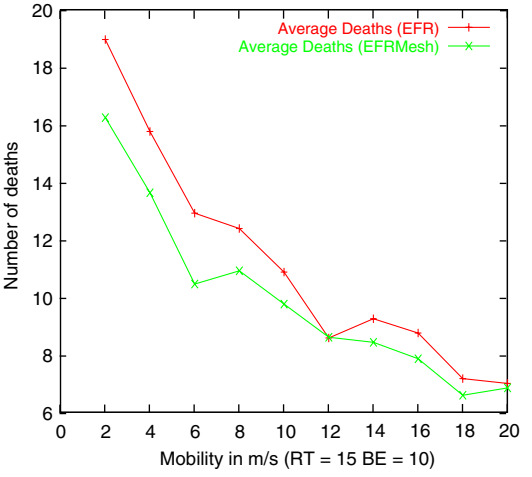

Fig. 25. Average number of *deaths* of mobile nodes vs. mobility.

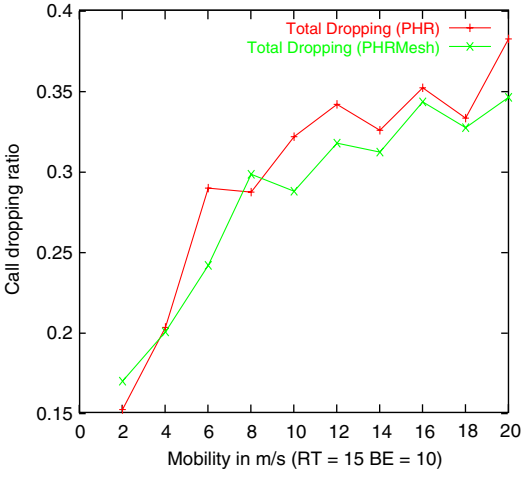

Fig. 24. Average call dropping ratio vs. mobility.

hence the system may not find enough bandwidth resources while attempting a path reconfiguration. Another reason for this increase in call dropping rate is the delay in releasing bandwidth upon detecting a broken path. The nodes in a path may take some time to detect that the path is broken and hence the bandwidth reserved for the path remains occupied and unused for a short while. This is one of the important contributors to the increase in dropping rate. Again, due to the slight difference in the call acceptance rate in PHR-Mesh, it shows a slightly lower call dropping rate. We noted that the node mobility inversely affects the number of *deaths* as presented in Fig. 25 for EFR heuristics. This is found to be in line with the performance in path breaks and call dropping rate. Since, the call dropping rate and path breaks are increasing with mobility, the resultant number of active calls in the system is less at high mobility. Therefore, the number of node *deaths* is found to be decreasing with mobility. We also noted that the node mobility almost equally affects the three heuristics.

6. Conclusions

In this paper, we have proposed the following: (i) a QoS extension of the DSR protocol for hybrid wireless mesh networks, (ii) three new slot allocation and positioning heuristics for supporting real-time traffic in asynchronous multihop wireless networks, and (iii) adaptation of the proposed heuristics for the asynchronous hybrid wireless mesh networks. The hybrid wireless mesh network is a multihop wireless network that consists of both fixed and mobile relay nodes. The QoS-DSR routing protocol, proposed in this paper, is designed to provide a stable route by preferring the fixed relay nodes over the mobile relay nodes. The routing decision, which is made at the destination node for selecting a path, is made based on the number of fixed relay nodes among the choice of available paths. The proposed new slot allocation and positioning heuristics are Early Fit Reservation (EFR), Minimum Bandwidth-based Reservation (MBR), and Position-based Hybrid Reservation (PHR). The EFR heuristic attempts to allocate bandwidth, on links starting from the source to destination, choosing earliest available slot. MBR attempts to provide reservation on links starting from the minimum bandwidth link, in the order of increasing bandwidth, choosing the

first available slot on each link. The PHR scheme provides a novel mechanism in which the position of the reservation slot, on a link, is placed in proportion to the link's position on the path. We also have adapted the slot allocation and positioning heuristics to support the highly resource constrained mobile relay nodes by providing *recovery-holes* along with bandwidth reservation, in order to recover the battery charge, in a hybrid wireless mesh network. We have carried out extensive simulation experiments to evaluate the performance of the proposed heuristics. The EFR heuristic has been found to give the best performance in terms of end-to-end delay while the MBR provides better throughput. The PHR provides an intermediate performance in comparison with EFR and MBR. The heuristics adapted to the hybrid wireless mesh environment provided an increased lifetime and fewer number of node *deaths* for the power-constrained mobile nodes, with slight increase in the end-to-end delay.

References

[1] D.B. Johnson, D.A. Maltz, Dynamic source routing in ad hoc wireless networks, in: T. Imielinski, H. Korth (Eds.), Mobile Computing, Kluwer, Boston, 1996, pp. 153–181.

[2] C.R. Lin, J.S. Liu, QoS routing in ad hoc wireless networks, IEEE Journal on Selected Areas in Communications 17 (8) (1999) 1426–1438.

[3] C.R. Lin, Admission control in time-slotted multihop mobile networks, IEEE Journal on Selected Areas in Communications 19 (10) (2001) 1974–1983.

[4] H.C. Lin, P.C. Fung, Finding available bandwidth in multihop mobile wireless networks. in: Proc. of IEEE VTC'00, vol. 2, May 2000, pp. 912–916.

[5] B.S. Manoj, C. Siva Ram Murthy, Real-time traffic support for ad hoc wireless networks, in: Proc. of IEEE ICON'02, August 2002, pp. 335–340.

[6] M.R. Garry, D.S. Johnson, Computers and Intractability, Freeman, San Francisco, CA, 1979.

[7] X. Zeng, R. Bagrodia,, M. Gerla, Glomosim: a library for parallel simulation of large-scale wireless networks, in: Proc. of IEEE PADS'98, May 1998.

[8] K. Lahiri, A. Raghunathan, S. Dey, D. Panigrahi, Battery-driven system design: a new frontier in low power design. in: Proc. of ASP-DAC/VLSI Design'02, January 2002, pp. 261–267.

[9] Panasonic Lithium Ion Batteries: Individual Data Sheet CGR 17500. Available from: <http://www.panasonic.com/industrial/battery/oem/chem/lithion/>, April 2002.

[10] V. Vidhyashankar, B.S. Manoj, C. Siva Ram Murthy, Slot allocation strategies for delay sensitive traffic in multihop wireless networks, in: Proc. of HiPC'03, December 2003, pp. 333–342.

[11] C.F. Chiasserini, R.R. Rao, Importance of a pulsed battery discharge in portable radio devices, in: Proc. of ACM MOBICOM'99, August 1999, pp. 88–95.

[12] D. Panigrahi, C.F. Chiasserini, S. Dey, R.R. Rao, A. Raghunathan, K. Lahiri, Battery life estimation for mobile embedded systems, in: Proc. Int. Conf. VLSI Design'01, January 2001, pp. 55–63.

V. Vidhyashankar obtained his B.Tech. degree in Computer Science and Engineering in 2003 from the Indian Institute of Technology (IIT), Madras, India. He is currently working towards the Ph.D. degree in the department of Computer Science at the Cornell University, Ithaca, USA.

B.S. Manoj completed his graduation in 1995 and post graduation in 1998 both in Electronics and Communication Engineering from Institution of Engineers (India) and Pondicherry Central University, Pondicherry, India, respectively. He has worked as a Senior Engineer with Banyan Networks Pvt. Ltd., Chennai, India from 1998 to 2000 where his primary responsibility included design and development of protocols for real-time traffic support in data networks. He was an Infosys doctoral student during 2000–2003 in the Department of Computer Science and Engineering at the Indian Institute of Technology (IIT) Madras, India, where he focused on the development of architectures and protocols for Ad hoc wireless networks and next generation hybrid wireless network architectures. During January 2004–January 2005, he was a Project Officer at IIT, Madras, India. He is currently a post-doctoral researcher at the University of California, San Diego, USA. Indian Science Congress Association has awarded him the Young Scientist Award for the Year 2003. His current research interests include Ad hoc wireless networks, next generation wireless architectures, and wireless sensor networks.

C. Siva Ram Murthy received the B.Tech. degree in Electronics and Communications Engineering from Regional Engineering College (now National Institute of Technology), Warangal, India, in 1982, the M.Tech. degree in Computer Engineering from the Indian Institute of Technology (IIT), Kharagpur, India, in 1984, and the Ph.D. degree in Computer Science from the Indian Institute of Science, Bangalore, India, in 1988.

He joined the Department of Computer Science and Engineering, IIT, Madras, as a Lecturer in September 1988, and became an Assistant Professor in August 1989 and an Associate Professor in May 1995. He has been a Professor with the same department since September 2000. He has held visiting positions at the German National Research Centre for Information Technology (GMD), Bonn, Germany, the University of Stuttgart, Germany, the University of Freiburg, Germany, the Swiss Federal Institute of Technology (EPFL), Switzerland, and the University of Washington, Seattle, USA.

He is the co-author of the textbooks Parallel Computers: Architecture and Programming, (Prentice-Hall of India, New Delhi, India), New Parallel Algorithms for Direct Solution of Linear Equations, (John Wiley & Sons, Inc., New York, USA), Resource Management in Real-time Systems and Networks, (MIT Press, Cambridge, Massachusetts, USA), WDM Optical Networks: Concepts, Design, and Algorithms, (Prentice Hall, Upper Saddle River, New Jersey, USA), and Ad Hoc Wireless Networks: Architectures and Protocols, (Prentice Hall, Upper Saddle River, New Jersey, USA). His research interests include parallel and distributed computing, real-time systems, lightwave networks, and wireless networks. He has published more than 100 international journal and 100 international conference papers in these areas.

Dr. Murthy is a recipient of Best Ph.D. Thesis Award from the Indian Institute of Science, Indian National Science Academy (INSA) Medal for Young Scientists, and Dr. Vikram Sarabhai Research Award. He is a co-recipient of Best Paper Awards from the 5th IEEE International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS), and the 6th and 11th International Conferences on High Performance Computing (HiPC).

He is a Fellow of the Indian National Academy of Engineering and an Associate Editor of IEEE Transactions on Computers.