Slot Allocation Schemes for Delay Sensitive Traffic Support in Asynchronous Wireless Mesh Networks*

V. Vidhyashankar, B.S. Manoj, and C. Siva Ram Murthy

Department of Computer Science and Engineering
IIT Madras 600036, INDIA
`vidya@dcs.cs.iitm.ernet.in`, `bsmanoj@.cs.iitm.ernet.in`, `murthy@iitm.ernet.in`

Abstract. In this paper, we propose an on-demand QoS routing protocol and heuristics for the slot allocation process in asynchronous single channel wireless mesh networks in the presence of hidden terminals. The heuristics we propose are the Early Fit Reservation (EFR), Minimum Bandwidth Reservation (MBR), and the Position-based Hybrid Reservation (PHR). The heuristic EFR has been found to give the best performance in terms of delay while the PHR and the MBR provide a better throughput. The heuristics proposed above have been adapted to provide an extended battery life for the power constrained mobile nodes and hence reduce the number of battery recharges. The parent and the adapted heuristics are compared in terms of delay and number of battery recharges. Simulation studies have shown that the parent heuristics show better results with respect to delay while the adapted heuristics perform better in terms of the number of *deaths* of mobile nodes.

1 Introduction

One of the major challenges in multihop wireless networks that exists today is that of providing Quality of Service (QoS) guarantee for real-time services. The complexity of the problem arises due to the unpredictability of the network topology caused by the mobility of the nodes. A path between any two nodes in a multihop wireless network can be obtained either by a static table-driven routing approach such as the Destination Sequenced Distance Vector (DSDV) [1] protocol, or by a dynamic on-demand routing approach such as the Dynamic Source Routing (DSR) [2] protocol. In a multihop environment that supports time-bound services, one has to not only set up a path with the minimum length but also reserve bandwidth so as to ensure delivery of data within the prescribed deadline. Hence reservation of bandwidth resources for real-time communications will aid a QoS-guaranteed packet delivery.

A heterogeneous multihop wireless network (wireless mesh network) is a kind of Ad hoc wireless networks that operates with partial infrastructure. Wireless

---

* This work was supported by Infosys Technologies Ltd., Bangalore, India and Department of Science and Technology, New Delhi, India.

T.M. Pinkston and V.K. Prasanna (Eds.): HiPC 2003, LNCS 2913, pp. 333–342, 2003.
© Springer-Verlag Berlin Heidelberg 2003

mesh networks provide an alternative communication infrastructure to existing cellular networks. 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 more resources and are dedicated to forward data and possibly serve as gateways to the Internet.

Our work in this paper, involves development of a QoS routing protocol and implementation of slot allocation schemes for mesh networks 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. A node uses relative time information to convey the exact positions of the reservation slots to a neighbor node, much similar to RT-MAC [3]. To distinguish the mobile nodes from the fixed relay nodes, we assume that the former have a finite battery life and may have to recharge their batteries whenever they get discharged, while the latter are assumed to have enough power to sustain for a long period of time. We have modified the DSR protocol [2] to enable it to support QoS routing. We have proposed three heuristics for the slot allocation algorithm and studied their performance using simulations.

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

2 Related Work

QoS issues in Ad hoc wireless networks have been studied in several works such as [4], [5], [6], and [7]. But many of these works have assumed the existence of either a multi-antenna model as in [8], which counters the effect of interference, or a CDMA-over-TDMA model as in [5], [6], and [7]. In the CDMA-over-TDMA infrastructure, multiple sessions can share the TDMA slot using different spreading codes. In many of these works, a transmitter-based assignment scheme is used to assign a code to each transmitter for data transmission. Different spreading codes are assigned to those nodes within two hops so as to reduce the hidden terminal problem.

C. R. Lin proposed a table-driven QoS routing mechanism in [5] and an on-demand call admission control scheme in [6] for a TDMA environment. The scheme in [5] reserves slots for real-time traffic using a QoS extension of the DSDV protocol. 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 [9]. Hence a heuristic has been proposed in [5] for assigning slots to each node. This protocol suffers from a high blocking probability. The work in [6] proposes an extension of the strategy to an on-demand routing protocol. This yields a better performance than the protocol proposed in [5] as several routes are attempted in parallel. The work in [7] describes a bandwidth allocation algorithm performed at the destination node for every link in the path implemented with an on-demand routing protocol in a CDMA-over-

TDMA based synchronous environment. The algorithm in [7] 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. 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.

Protocols Used in Our Work: The algorithms proposed in this work are applicable to both synchronous TDMA-based and asynchronous TDMA-based Ad hoc wireless networks. We have chosen an asynchronous environment owing to two major reasons. One is the fact that synchronous environments like the TDMA are expensive in terms of both battery power and bandwidth unlike asynchronous ones. The other is the problem of frequent partitioning and merging that occurs often in Ad hoc networks. In the synchronous case, apart from the synchronization overhead, there is bound to be some calls getting dropped due to collisions or synchronization differences during the synchronization period.

In order to provide call admission control and reservation of bandwidth in the asynchronous environment, we require a MAC protocol which has the respective capabilities. This is the reason for selecting the Real Time MAC (RTMAC) protocol proposed in [3], a bandwidth efficient, flexible and asynchronous MAC protocol that supports both real-time (RT) and best-effort (BE) traffic. In RT-MAC, bandwidth is provided by dividing the transmission time into successive super-frames. The bandwidth reservations for RT traffic are made by reserving variable-length time slots (*conn-slots*) on the super-frames. The slot allocation differs from the TDMA scheme since no time synchronization has been assumed by the authors of [3] and all reservations are done with respect to the relative time period by which the RT session starts. The reservation is performed through a three-way ResvRTS-ResvCTS-ResvACK handshake. In our work, QoS routing has been implemented as an extension of the DSR protocol [2]. 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.

3 Slot Allocation Framework

The wireless mesh network has been implemented in an asynchronous environment with QoS-DSR as the routing protocol and RTMAC as the MAC protocol. The QoS routing protocol has three phases of operation *viz.,* the *Bandwidth Feasibility Test Phase*, the *Bandwidth Allocation Phase*, and the *Bandwidth Reservation Phase.* These phases will be discussed later in this section.

Power Management

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 transmission. The battery of the node may experience a charge recovery [10] 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 relay nodes, on the other hand are assumed to have infinite power as they have a supportive infrastructure. Recent studies on storage cell characteristics [10] show that a pulsed discharge can extend the battery life more than a continuous discharge. 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 function ([10] and [11]) depending on the current state (the discharge potential and the theoretical capacity) is used to model the recovery. According to the recovery model, the battery may recover when the node is idle. The battery reaches a *dormant state* either when its battery voltage has reached the cut-off voltage (V_{cut}) or when the theoretical capacity has been exhausted.

The QoS-DSR Protocol

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

The Bandwidth Feasibility Test Phase: During 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 reservation information (of the path traversed so far) in the *Route Request* packet that it received. The updated synchronization information is piggybacked on the new *Route Request* packet which is then broadcast. The reservation information is in the form of reservation frames containing time durations. Each of these time durations associated with a state variable indicates whether a slot is reservable during that duration or unreservable because the node and (or) at least one of its neighbors are involved in a call. Of the *Route Request* packets collected within a time interval (called *RouteRequestWindow*), the destination node chooses the one with the maximum number of fixed relay nodes in the path. Other *Route Request* packets that arrive beyond the *RouteRequestWindow* are discarded.

The data structure used for the bandwidth allocation algorithm is called the QoS frame. The reservation information of each link provided in the *Route Request* packet, is used for constructing the QoS frame. It is a linear array of blocks, each block containing a time duration and the state to which it belongs. The state is of the form ASYNC_X_Y where X and Y correspond to the states of the sender and receiver node of the link respectively (refer to Figure 1). X and Y can be one of FREE (the medium is free in its vicinity), UNRESV (when the medium is used by the node either as a sender or as a receiver) or HT (when the node serves as a hidden terminal to a transmission involving one or more of its neighbors).

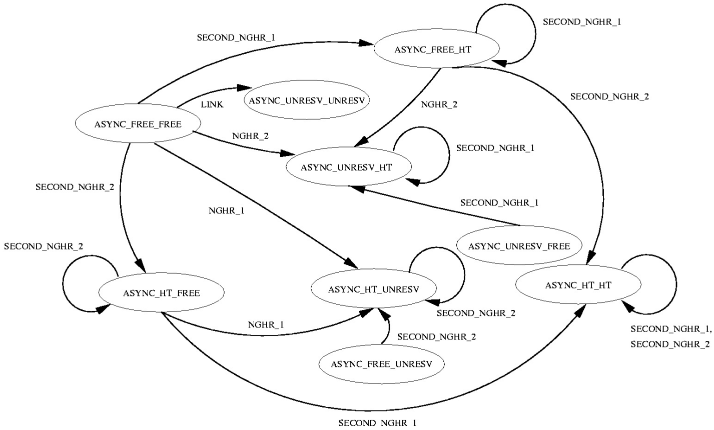

Fig. 1. State diagram describing the UpdateState function. The transitions are labeled, each label referring to the nature of the link *viz.,* LINK (the link for which the reservation is being made), NGHR_1 and NGHR_2 (the first neighboring links), and SECOND_NGHR_1 and SECOND_NGHR_2 (the second neighboring links).

The Bandwidth Allocation Phase: During this phase, the destination node performs a link-wise *conn-slot* allocation algorithm to assign the *conn-slots* at every link. 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 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. 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. A guard band interval is included on both the sides of the *conn-slot* to cushion the error incurred in synchronization. The algorithm returns the time durations allocated for each link if it is successful, otherwise returns NULL. The function *Resv_Bandwidth* presented in Figure 2 contains the generic bandwidth allocation algorithm. A number of (equal to the number of nodes in the path) invocations of the *GetAvailable_Bandwidth_Info* function which is presented in Figure 2 finally produces the frame *ResultantFrame* which contains the free time durations during which a *conn-slot* may be reserved. The UpdateState function used in the algorithm shown in Figure 2 can be understood from the state diagram in Figure 1. 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). 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. The heuristics proposed are described as follows.

(i) **Early Fit Reservation (EFR):** The EFR heuristic 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 *conn-slot* reserved for 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 allocates bandwidth to the links in the increasing order of free *conn-slots i.e.,* 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.

(iii) **Position-based Hybrid Reservation (PHR):** The PHR heuristic assigns bandwidth for every link with the slot placement proportional to its position in the path. The operation of this heuristic is similar to the previous one. The difference lies in the selection of the *conn-slot* for reservation at each link. The link with the least free bandwidth is selected for each iteration. We observe that this heuristic tries to reap the benefits of both the EFR and the MBR heuristics. Thus, the earlier links in the path are reserved (at a greater probability) in the beginning of the frame while the later links in the path are reserved towards the end. Hence this type of reservation yields a lesser delay than the MBR heuristic.

To adapt these heuristics to mesh networks, we have modified them to EFRMesh, MBRMesh, and PHRMesh. The idea of the adapted heuristics is to allocate a free time duration (hole) in a link just before a *conn-slot* during the bandwidth allocation algorithm, whenever the sender node of the link is a mobile 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 node, a *conn-slot* is assigned such that there is a hole (*recovery-hole*) just before it. However, if the sender node of the link is a fixed relay node, the *recovery-hole* is not allocated. The *recovery-hole* can possibly be used by the mobile node for the 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.

The Bandwidth Reservation Phase: If the bandwidth allocation algorithm is successful in its reservation, information about the bandwidth allocation will

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 and 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} ) $\wedge$
      (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}) $\wedge$
      (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) $\wedge$ (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.
```

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 1.
            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 $\phi$. //Unsuccessful Reservation.
   End of while loop.
3) Return Resv.
4) End.
```

Fig. 2. Algorithm to obtain information on free bandwidth and the generic algorithm. Steps 1, 2.*i*, 2.*j.(iii)* in the generic algorithm depend on the heuristic (EFR, MBR, and PHR) employed.

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 performs call setup using the RTMAC. If it is successful, the *Route Reply* packet is then sent to the next node. This goes on till the entire path reservation is completed.

4 Simulation Results

The proposed QoS-DSR and the heuristics for the asynchronous environment were simulated using GloMoSim. 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 milliseconds, for both RT and BE connections. Simulation time was taken as 600 seconds and CBR sessions of 200 seconds were generated randomly between 0 and 400 seconds of the simulation. CBR sessions were classified into two classes, namely RT and BE sessions. Terrain range used for the simulations was 1000m × 1000m. Transmission range of a node was taken to be 300m. The simulated network has 30 nodes which were uniformly distributed in the terrain area. Random way-point mobility model is used. The average end-to-end delay in RT communications is a crucial parameter in deciding the performance of the protocol. In these simulations, we find that the average end-to-end RT delay is the least for the EFR heuristic. The reason can be attributed to the manner in which *conn-slots* are assigned. In the EFR, we find that successive links are assigned their slots in order. The session intervals are assigned in an early-fit manner. Hence it is bound to give the least delay when compared to the other heuristics. The PHR gives a delay lesser than its parent heuristic. This can be explained by the slot selection method followed in the heuristic. The first experiment is a comparison of the three parent heuristics on increasing network load ([12]). Figure 3 compares the average end-to-end delay of the three heuristics. The EFR gives the least delay while the MBR yields the maximum delay and the PHR provides a delay in between the two. Figure 4, comparing the throughput of the three heuristics, shows that the MBR and the PHR give a greater throughput than the EFR.

The next set of simulation results compare a pair of heuristics X and XMesh (X = EFR or PHR). 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*. Figure 5, comparing the average RT delay between EFR and EFRMesh under increasing network load with no BE calls, shows that the EFRMesh suffers due to the allocation of *recovery-holes*. Figure 6, that compares the average number of *deaths* between PHR and PHRMesh under increasing network load with a heavy BE load, shows that, on an average, PHRMesh causes a lower number of *deaths* than PHR. Figure 7 shows that the call dropping ratios in PHR and PHRMesh increase with increasing mobility, with the PHRMesh heuristic exhibiting better results on an average. Figure 8 shows that the EFRMesh heuristic displays a lower number of *deaths* of mobile nodes than the EFR heuristic, on increasing mobility.

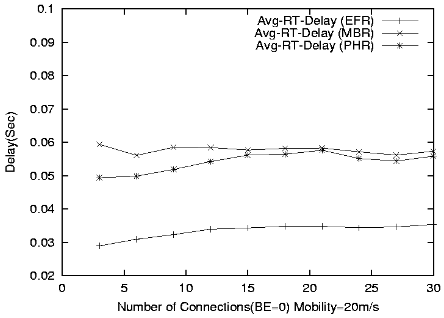

Fig. 3. Average end-to-end delay vs. network load (mobility = 20m/s).

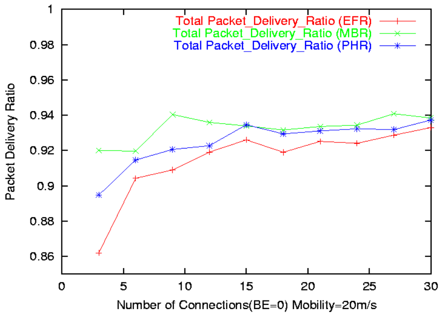

Fig. 4. Packet delivery ratio vs. network load (mobility = 20m/s).

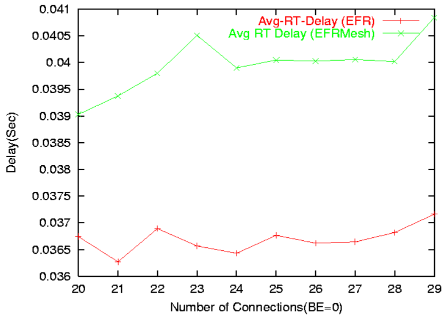

Fig. 5. Average end-to-end delay vs. network load (mobility = 20m/s).

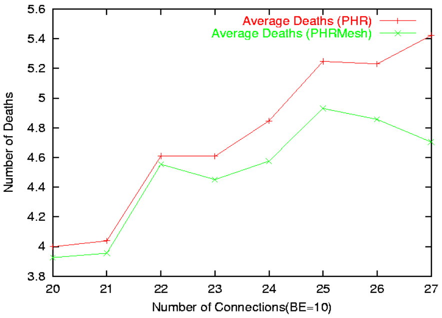

Fig. 6. Average number of *deaths* vs. network load (mobility = 20m/s).

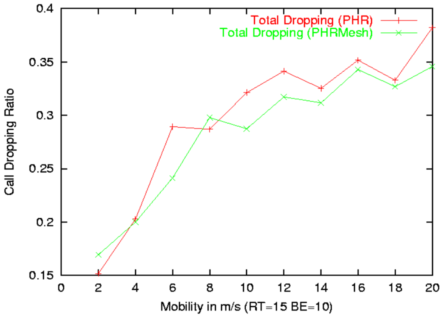

Fig. 7. Average call dropping ratio vs. mobility.

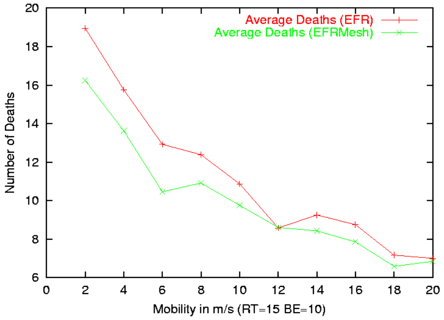

Fig. 8. Average number of *deaths* of mobile nodes vs. mobility.

5 Summary

In this paper, we have proposed a QoS extension of the DSR protocol with slot allocation heuristics for the asynchronous multihop wireless networks. The heuristic EFR has been found to give the best performance in terms of delay while the MBR provides a better throughput. The PHR provides an intermediary performance. The heuristics have been adapted to the mesh environment to provide an increased lifetime for the power-constrained mobile nodes. We have adopted a pulsed discharge scheme for the mobile nodes according to which the battery of a mobile node recovers during idle time and reaches a *dormant state* whenever it gets discharged completely. The fixed infrastructure-based nodes, on the other hand, are assumed to have infinite power. Simulation studies have shown that the adapted heuristics perform better in terms of the number of *deaths* of mobile nodes.

References

1. C. E. Perkins and P. Bhagwat, “Highly Dynamic Destination-Sequenced Distance Vector Routing (DSDV) for Mobile Computers,” *Proc. of ACM SIGCOMM 1994,* pp. 234-244, September 1994.
2. D. B. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad hoc Wireless Networks”, *Mobile Computing*, Edited by T. Imielinski and H. Korth, Kluwer, pp. 153-181, 1996.
3. B. S. Manoj and C. Siva Ram Murthy, “Real-time Traffic Support for Ad Hoc Wireless Networks”, *Proc. of IEEE ICON 2002*, August 2002.
4. C. R. Lin and M. Gerla, “MACA/PR: An Asynchronous Multimedia Multihop Wireless Network”, *Proc. of IEEE INFOCOM ’97*, March 1997.
5. C. R. Lin and J. S. Liu, “QoS Routing in Ad hoc Wireless Networks”, *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pp. 1426-1438, August 1999.
6. C. R. Lin, “Admission Control in Time-Slotted Multihop Mobile Networks”, *IEEE Journal on Selected Areas in Communications*, Vol. 19, No. 10, pp. 1974-1983, October 2001.
7. H. C. Lin and P. C. Fung, “Finding Available Bandwidth in Multihop Mobile Wireless Networks”, *Proc. of IEEE VTC*, Tokyo, Japan, May 2000.
8. S. Chen and K. Nahrstedt, “Distributed Quality-of-Service Routing in Ad Hoc Networks”, *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pp. 1594-1603, August 1999.
9. M. R. Garry and D. S. Johnson, *Computers and Intractability*, San Francisco, CA Freeman 1979.
10. C. F. Chiasserini and R. R. Rao, “Importance of a Pulsed Battery Discharge in Portable Radio Devices”, *Proc. of ACM MOBICOM 1999,* August 1999.
11. D. Panigrahi, C. F. Chiasserini, S. Dey, R. R. Rao, A. Raghunathan, and K. Lahiri, “Battery Life Estimation for Mobile Embedded Systems”, *Proc. Int. Conf. VLSI Design,* pp. 55-63, January 2001.
12. V. Vidhyashankar, B. S. Manoj, and C. Siva Ram Murthy, “Slot Allocation Strategies for Delay Sensitive Traffic in Multihop Wireless Networks”, *Technical Report, HPCN Lab, Department of Computer Science and Engineering, IIT Madras,* April 2003.