QoS-Aware Fair Scheduling in Wireless Ad Hoc Networks with Link Errors*

Muhammad Mahbub Alam, Md. Mamun-or-Rashid, and Choong Seon Hong**

Department of Computer Engineering, Kyung Hee University
1 Seocheon, Giheung, Yongin, Gyeonggi, Korea, 449-701
{mahbub, mamun}@networking.khu.ac.kr,
cshong@khu.ac.kr

Abstract. To provide scheduling in wireless ad hoc networks, that is both highly efficient and fair in resource allocation, is not a trivial task because of the unique problems in wireless networks such as location dependent and bursty errors in wireless link. A packet flow in such a network may be unsuccessful if it experiences errors. This may lead to situations in which a flow receives significantly less service than it is supposed to, while other receives more, making it difficult to provide fairness. In this paper we propose a QoS-aware fair scheduling mechanism in ad hoc networks considering guaranteed and best-effort flows in the presence of link errors. The proposed mechanism provides short-term fairness for error free sessions and long-term fairness for the erroneous sessions and allows a lagging flow to receive extra service and a leading flow to give up its extra service in a graceful way. It also maximizes the resource utilization by allowing spatial reuse of resource. We also propose a CSMA/CA based implementation of our proposed method.

1 Introduction

A wireless ad hoc network consists of a group of mobile nodes without the support of any infrastructure. Such a network is expected to support advanced applications such as communications in emergency disaster management, video conferencing in a workshop or seminar, communications in a battlefield. This class of mission-critical applications demands a certain level of quality of services (QoS) for proper operations. Also due to the distributed nature of these networks providing a fair access of resource to multiple contending nodes is an important design issue .

Fairness is an important criterion of resource sharing in the best effort Internet, especially when there is a competition for the resource among the nodes due to unsatisfied demands. In fair scheduling each flow f is allowed to share a certain percentage of link capacity based on its flow weight indicated as w_f. Let $W_f(t1, t2)$ and $W_g(t1, t2)$ denote the aggregate resources received by flows f and g respectively in time interval $[t1, t2]$ and w_f and w_g are the flow weights of the flows f and g respectively. The allocation is ideally fair if it satisfies (1)

* This work was supported by MIC and ITRC Project.
** Corresponding author.

Y.-T. Kim and M. Takano (Eds.): APNOMS 2006, LNCS 4238, pp. 1 – 10, 2006.

$$\left| \frac{W_f(t_1, t_2)}{w_f} - \frac{W_g(t_1, t_2)}{w_g} \right| = 0 \quad (1)$$

However, most of the research works assumed error free wireless link which is not realistic. In the wireless environment, a packet flow may experience channel error and the transmission may not be successful. Thus the bursty and location dependent error in wireless link may make the existing fair scheduling algorithm inapplicable. Therefore, the goal of ad hoc network fair scheduling is to make short burst of location-dependent channel error transparent to users by a dynamic reassignment of channel allocation over small timescales [14]. Specifically, a backlogged flow *f* that perceives a channel error during a time window $[t_1, t_2]$ is compensated over a later time window $[t_1', t_2']$ when *f* perceives a clean channel. Compensation for *f* involves granting additional channel access to *f* during $[t_1', t_2']$ in order to make up lost channel access during $[t_1, t_2]$, and this additional channel access is granted to *f* at the expense of flows that were granted additional channel access during $[t_1, t_2]$.

Providing QoS in Wireless ad hoc networks is a new area of research. Existing works focuse mainly on QoS routing which finds a path to meet the desired service requirements of a flow. In this paper we consider a mix of guaranteed and best effort flows and investigate fair queueing with QoS support for the network. The goal is to guarantee the minimum bandwidth requirements of guaranteed flows and to ensure a fair share of residual bandwidth to all flows.

In this paper we focus on the fair scheduling issues in a hoc network in the presence of channel errors. We develop a fairness model for wireless ad hoc fair scheduling to deal with channel error. We also implement the model in a distributed manner by localizing the global information required by the nodes.

The rest of the paper is organized as follows. Section 2 describes related works. In Section 3 we explain the network model and problem specifications. Section 4 describes the proposed mechanism and is followed by the details of the implementation of the proposed mechanism in section 5. Section 6 presents the simulation and results. We conclude in section 7 by conclusion and future works.

2 Related Works

Fair queueing has been a popular paradigm for providing fairness, minimum throughput assurance and guaranteed delay in wired network [1], and in packet cellular networks [2] – [4]. Recently some techniques have been proposed to incorporated fair queueing in shared channel, multihop wireless networks [5] – [7]. Also, providing QoS in wireless ad hoc networks is a new area of research. Some of the research works also incorporated both QoS and fair queueing in ad hoc networks. Both QoS guarantee and fair queueing in ad hoc networks have been proposed in [8] and [9]. Also some of the works provide fairness with error compensation, e.g., [13], [14], [15] and most of these are proposed based on the support of base stations.

In [13], channel-condition independent packet fair queueing (CIF-Q) is proposed. Each session is associated with a parameter called *lag* to indicate whether the session should be compensated. If a session is not leading and the channel is error free at its scheduled time, its head-of-line packet is transmitted; otherwise, the time slot is

released to other sessions. The problem with CIF-Q is that the leading sessions are not allowed to terminate unless all their leads have been paid back, regardless of whether such terminations are caused by broken routes. This property makes CIF-Q an inadequate solution for ad hoc networks, because a connection may be broken.

The Idealized Wireless Fair-Queueing (IWFQ), and the Wireless Packet Scheduling protocol (WPS) are proposed in [14]. In this paper the base station calculates the number of slots per frame, each flow can use, based on the weight of the flows. If a flow experiences channel error at its allocated time, the base station tries to find a flow that can exchange its slot with the flow within the frame; and the error session of the flow is compensated at a later frame. The mechanism is not suitable for ad hoc networks, as it requires the presence of a base station.

In [15], QoS-aware fair scheduling is proposed for mobile ad hoc networks in the presence of error; and based on the channel condition estimation, a flow may either transmit or give up its allocation to others. When this flow perceives an error free channel it will have packets with smaller service tag than that of its neighbors' and its packets will be transmitted first. The problem with this protocol is that after recovery from error a flow exclusively accesses the channel until its virtual time catches up with other flows. Also the service release of a leading flow is not graceful.

3 Network Model and Problem Specifications

In this paper we consider shared channel, packet-switched, multihop wireless ad hoc networks where the hosts are not mobile. There is contention for channel among multiple nodes and errors in wireless link are location dependent and bursty in nature. Therefore, some hosts may not be able to transmit data due to channel errors even when there is backlogged flows on those hosts while others may have error-free channels and can transmit data in that time.

We define the error-free service of a flow as the service that had been error-free. A flow is said to be leading if it has received channel allocation in excess of its error-free service. A flow is said to be lagging if it has received channel allocation less than its error-free service. If a flow is neither leading nor lagging, it is said to be in sync.

During a period of channel error, error-free flows will receive more service than their fair share, while a flow with errors will receive no service. Since the virtual time of a session increases when it receives service, this may result in a large difference between the virtual time of an erroneous flow and that of an error-free session. We address the following issues of ad hoc network fair scheduling with channel errors:

i) If flow f_i exits from errors, and is allowed to retain its virtual time, then it will have the smallest virtual time among all flows. Then f_i will exclusively access the channel until its virtual time catches up with those of other flows and all the other flows will receive no service.

ii) If flow f_i exits from error, and its virtual time is updated to the system virtual time $V(\bullet)$, then error-free flows will not be penalized and f_i will never be able to regain its lost service, resulting in unfair behaviors.

iii) The compensation of all lagging flows should not take same amount of time regardless of their weights, and violate the main idea that larger weight implies better service.

iv) The extra service releases by a leading flow should be graceful to guarantee short-term fairness.

4 Proposed Fair Scheduling Mechanism

We now describe a new approach to QoS-aware distributed fair scheduling model in ad hoc networks. This model is fully distributed, localized and local scheduler at each node has a certain level of coordination with its neighboring nodes. And this does not require any global information propagation and global computation. The mechanism is as follows:

1) Maintaining Flow Information within Two-hop Neighborhood: Each node maintains two tables for the proper operation of fair scheduling. One table is to keep the flow information of two-hop neighbors, say *flow_table* and is kept sorted according to service tag of the flows. The fields of the table are *node_id*, *flow_id*, *service_tag,* s_i, and *lag*. Another table, *compensation_table*, contains the compensation tag of the lagging flows. The fields of the table are *node_id*, *flow_id* and *compensation_tag* and kept sorted according to the *compensation_tag*.

2) Assignment of flow weight: We assume that both guaranteed and best-effort flows exist simultaneously in the network. Flow weights are assigned based on [16]:

Weight of QoS flows, w_q	Weight of best-effort flows, w_b

For i = 1 to n {
w =Min_f/C +(C-$\sum Min_f$)/(n + m)
if w*C > Req_f
W_q = C/Req_f
else
W_q = w
}
	For i = 1 to m {
	w_b =(C – ($\sum w_g$)*C)/m
	}

m = number of QoS flows C = Link Bandwidth	m = number of best effort flows

We consider a multi-hop flow consists of number of single hop flows and the flow weight is not fixed for the path; every forwarding node assigns a different weight. Over time, based on the current flows within two-hop neighbors, the weight may change, otherwise either the network utilization or the fairness will be poor.

3) Tagging Operations: For each flow *f* we use the SFQ [10] algorithm to assign tags for the arriving packets: a *start tag* and a *finish tag*.

4) Path Registration: To provide guaranteed service a routing protocol should find a feasible path. AODV [11] is one of the most widely used table-based and reactive routing protocols. But AODV is designed to find a feasible route only. Therefore, to support QoS we need to modify AODV. To ensure QoS, AODV is modified to support two types of schemes: *admission scheme* and *adaptive scheme*. In admission scheme a feasible path should provide the required minimum bandwidth, while in adaptive feedback scheme the source is informed about the minimum available bandwidth so that the source can adapt its transmission speed.

To initiate QoS-aware routing discovery, the source host sends a RREQ packet whose header is changed to <*model-flag*, *required bandwidth*, *min-bandwidth*, AODV RREQ header>. The model-flag indicates whether the source is using the admission scheme or the adaptive feedback scheme. When an intermediate host receives the RREQ packet, it first calculates its residual bandwidth. If the model-flag is set to admission scheme, the host compares its residual bandwidth with the minimum requested bandwidth. If its residual bandwidth is greater than the minimum bandwidth, it forwards this RREQ. Otherwise, it discards this RREQ. If the model-flag is adaptive, the host compares its residual bandwidth with the min-bandwidth field in the RREQ. If its residual bandwidth is greater than the min-bandwidth, it forwards the RREQ. Otherwise, it updates the min-bandwidth value using its residual bandwidth. Finally the forwarding node temporarily stores the flow. When a node forwards a RREP message it assigns a flow-weight to the flow and stores the flow information.

5) Channel Error Prediction: Perfect channel-dependent scheduling is only possible if a node has accurate information about the channel state. The location-dependent nature of channel error requires each node to monitor its channel state continuously, based on which the node may predict its future channel state. To do this, each node periodically measures the signal-to-noise ratio (SNR) of the channel. If the SNR value falls below a predefined threshold, the channel is considered as error-prone.

6) Lead and Lag Model: The purpose of the lead and lag model is to determine how much additional service a lagging flow is entitled to receive in the future in order to compensate service lost in the past and how much service a leading flow should relinquish in the future in order to give up additional services received in the past. Also, the lag of a lagging flow is incremented on a lost time slot only if another flow received its service and is ready to release this service.

To represent the amount of lead and lag service of a flow we use a parameter called *lag*. The value of *lag* of a lagging flow is positive, it is negative for a leading flow and zero otherwise.

When a node f_i perceives an erroneous wireless link and it is the turn for one of its flow to transmit, the node does not transmit. Instead one of its neighbors, f_j, which perceives a good channel and has the immediate higher service tag (lagging flows get higher preference than leading flows) will transmit. Both of the flows will either initialize (if they just become lagging and leading) or update their *lag* value.

7) Compensation Model: The compensation model is the key component of ad hoc network fair scheduling algorithm in the presence of errors. It determines how lagging flows receive extra service to make up their lag and leading flows give up their lead to relinquish extra service. The compensation model should take into account the following two conditions to ensure short-term fairness:

i) The service releases by a leading flow should not be more than a certain fraction of its received service.

ii) The lagging flows should receive the extra service in proportion to their flow weights. That is, flow with largest *lag* should not receive more service irrespective to its guarantee rate.

To achieve graceful degradation of service of a leading flow, it relinquishes a fraction of services allocated to the flow. We define a system parameter α ($0 \leq \alpha \leq 1$) to control the minimal fraction of service retained by a leading session. That is, a leading flow can give up at most $(1 - \alpha)$ amount of its service to compensate for lagging

flows. Each leading flow, f_i, is associated with another parameter, called normalized service received by the leading flow, s_i. When a flow becomes leading, s_i is initialized to αv_i (v_i is the virtual time of f_i) and s_i is updated whenever f_i is served.

On the other hand, lagging flows should have higher priority to receive extra service to compensate their loss. And such compensation is possible because leading flows give up their leads. To provide short-term fairness we distribute these additional services among lagging flows in proportion to the lagging flows weights. To accomplish this we use the compensation virtual time, c_i, which keeps the track of normalized amount of compensation services received by a flow while it is lagging. When flow f_i becomes lagging c_i is initialized according to (2)

$$c_i = \left\lceil \max(c_i, \min_{k \in A} \{ f_k \mid \mathtt{lag}_k > 0 \} \right\rceil \tag{2}$$

8) Scheduling Mechanism: For scheduling the flows we use a table driven, back-off based approach which uses local information and local computation only. With the tagging operation and a method of exchanging tags, each node has the knowledge of its local neighborhood. These tags are stored in tables and are ordered so that each node can learn whether that node itself has the minimum tag and therefore has to transmit next. The basic scheduling mechanism is as follows:

i) When a node finds one of its flows f_i has the minimum service tag and the flow is not leading or leading but did not receive a minimum fraction of service, then the packet at its queue is transmitted. However, if f_i is leading and already receives minimum service, it does not transmit its packet to give up its lead to a lagging flow. If there is any lagging flow within two-hop neighbors of the flow, then the flow f_j, which has the minimum compensation tag among the lagging flows can transmit its packet, otherwise f_i transmits its packet.

ii) After the packet to transmit has been determined, the virtual time of f_i is updated. If f_i is leading but receives services due to graceful degradation, s_i is updated to $s_i + L_p$. If f_j is served and the overhead is charged to f_i where ($i \neq j$), then the lag values are update by $lag_j = lag_j - L_p$ and $lag_i = lag_i + L_p$.

iii) When a flow just becomes lagging its *compensation_tag*, c_i is initialized by (2) and updated every time a lagging flow gets compensation service by $c_i = c_i + L_p / w_i$.

When flow f_i becomes leading its s_i is initialized by $s_i = \alpha v_i$ and updated every time it gets service due to graceful degradation by $s_i = s_i + L_p / w_i$.

9) Table Update: Whenever a node hears a new service tag for any flow on its table or a new flow, it updates the table entry for that flow or adds this flow information on its table. Whenever any node transmits a head-of-line packet for a flow, it updates that flow's service tag and compensation tag in the table entry.

5 Implementation of the Proposed Mechanism

In this section, we describe a distributed implementation of the proposed mechanism within the framework of CSMA/CA MAC architecture. Our implementation is based on the method used in [8] and we incorporated the channel errors with this addressing the following practical issues:

1) Message Exchange Sequence: In this mechanism, each data transmission follows a basic sequence of RTS-CTS-DS-DATA-ACK handshake, and this message exchange is preceded by a backoff of certain number of minislot times. At the begging of each transmission slot, each node checks whether it has a flow with minimum service tag. The flow with the minimum service tag is transmitted based on the rules explained next. To allow multiple flows to transmit simultaneously and to reuse the channel we assign a backoff value to flows like [8]. But we assign backoff value first to lagging flows and then to leading flows otherwise the leading flows will further increase their lead. The value of the backoff timer of flow f_i is the number of flows with tag smaller than the tag of the flow f_i.

If the node does not hear any transmission then it decreases backoff value by one in each minislot. If the backoff timer of f_i expires without overhearing any ongoing transmission, it starts RTS to initiate the handshake. If the node overhears some ongoing transmission, it cancels its backoff timer and defers until the ongoing transmission completes. In the meantime, it updates its local tables for the tags of the ongoing neighboring transmitting flow. When other nodes hear a RTS, they defer for one CTS transmission time to permit the sender to receive a CTS reply. Once a sender receives the CTS, it cancels all remaining backoff timers (for other flows) and transmits DS (other motivations for DS have been explained in [12]). When hosts hear either a CTS or a DS message, they will defer until the DATA-ACK transmission completes.

2) Transmission of Packet Based on Sender and Receiver's Table: To schedule a flow, a node should know the flow information of the neighbors of sender and receiver. This information needs to be combined and known by the sender to make the scheduling decision. A straight forward solution would be to broadcast the receiver's table to the sender periodically. However, significant overhead will be induced if the table is large and updated frequently. In our design we assign the backoff time of each flow based on table of both sender and receiver as in [8].

3) Exchanging Slot of an Erroneous Flow with another Flow that Perceives a Good Channel: In the absence of centralized scheduler, it is very difficult to exchange a transmission slot because the node, that is experiencing the errors, cannot convey this message to its two-hop neighbors. We consider a different approach for solving this problem. If the packet of flow f_i, which has the minimum flow tag, is not transmitted within the first minislot, then we consider that the flow is experiencing channel error. So the flow f_j with immediate higher service tag (as mentioned earlier we consider first the lagging flows and then the leading flows in ordering) will transmit the packet. As a result, f_i and f_j will become the lagging and leading flows respectively. Flow f_i will increase its *service_tag* and *lag* value and flow f_j will decrease its *lag* value. But this approach will create another problem, because now the neighbors could not identify whether f_i is experiencing channel errors or it has another neighbor (which is not a neighbor of f_j) with flow that has smaller flow tag than f_i. This problem arises because of the distributed nature of the ad hoc network fair scheduling. Here we generalize both of these two cases. Whatever the reason is as flow f_j is getting chance to transmit before f_i we consider f_i as lagging flow and f_j as leading flow. This assumption will further increase the fairness of the scheduling mechanism.

4) Compensation of Service: A leading flow can releases its extra service, when the flow gets chance to transmit (has the minimum service tag) but its $lag_i < 0$ and it already receives it minimum fraction of service. We assume that all of the two-hop

neighbors have the information regarding flow f_i, it is known to all that f_i should relinquish its extra service in this slot. Therefore, the next flow will be selected from one of the lagging flows (if there is any) and determined based on the *compensation_tag* instead of *service_tag*. Therefore, once a leading flow has the minimum service tag which has to release its extra services, backoff values are assigned only to lagging flows and as a result one of the lagging flows is scheduled to transmit its packet.

5) Propagation of Updated Service Tag: In order to propagate a flow's service tag to all its one-hop neighbors in the node graph and reduce the chance of information loss due to collisions during the propagation, we attach the *service_tag, compensation_tag* and s_i for flow f_i in all four packets RTS, CTS, DS and ACK. However, we do not use the updated tags for flow f_i in RTS and CTS packets, since RTS and CTS do not ensure a successful transmission. When the handshake of RTS and CTS is completed, we attach the updated flow tag in DS and ACK, to inform neighboring nodes about the new updated information of the current transmitting flow f_i.

6 Simulation and Results

In this section, we evaluate our proposed algorithm by simulation and we measure the fairness property of our algorithm. The flow graph used in the simulation is shown in figure 1. There are four flows where each of flow f_0 and f_2 makes contention with all remaining three flows respectively. Reusing of wireless resource allows flow f_1 and f_3 to be transmitted simultaneously and thus increases the throughput of the networks. Each of the flows starts at time 0. Flow f_1, f_2 and f_3 experience error-free wireless link and f_0 experiences a wireless link with error.

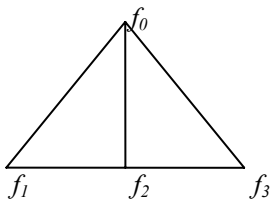

Fig. 1. Example flow graph used in simulation

Figure 2 and Figure 3 show the simulation result of our proposed mechanism. In Figure 2, we consider all the flows are experiencing error-free wireless channel.

In Figure 2(a), all the flows are best-effort flows and have the same weight. So, all flows have got the equal share of the bandwidth. In Figure 2(b), flow f_2 is guaranteed flow and all other flows are best-effort flows. Flow f_2 has the minimum bandwidth requirement of 0.2C, where C is the link bandwidth. So the flow weights are $w_2 = 2$ and $w_0 = w_1 = w_3 = 1$. Accordingly, flow f_2 got double share than the other flows.

In Figure 3, flow f_0 is experiencing an erroneous channel during time interval 4 to 9 seconds and so in this period flow f_0 does not receive any service. Figure 3(a) shows that, at that time f_1 receives the service (lost service of f_0) and as f_1 and f_3 can transmit

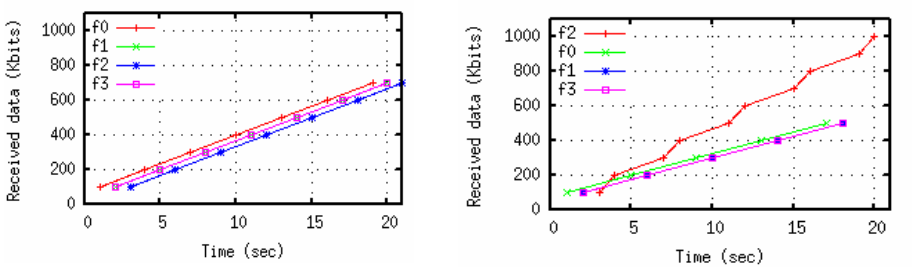

Fig. 2. Simulation results with error-free channel (a) All flows are best-effort (b) Flow f_2 is guaranteed, all others are best-effort

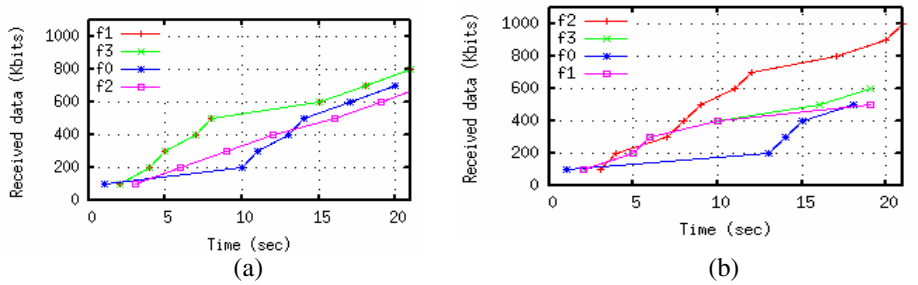

Fig. 3. Simulation results with error in wireless channel (a) all flows are best-effort and (b) flow f_2 is guaranteed flow and all the other flows are best-effort

simultaneously, so f_3 also receives extra service. As the graph shows the lost service of f_0 is compensated at time 11 and 13 seconds, which justify our proposed algorithm.

In Figure 3(b), f_2 is guaranteed flow and all other flows are best-effort flows. And flow f_0 experiences erroneous channel during time interval 4 to 9 seconds and later its service is paid back by flow f_1, f_2 and f_3 those who received the extra service earlier.

7 Conclusions

In this paper, we proposed a distributed fair scheduling algorithm for providing scheduling service in wireless ad hoc networks with errors in wireless link. Our proposed mechanism provides a compensation model for flows that experience an erroneous channel, and allows the flows to get their lost service at a later time. It also ensures that a leading flow can give up its extra service in a graceful way. Also we considered the presence of both QoS and best-effort flows simultaneously in the networks. Our proposed mechanism satisfies the minimum bandwidth requirement of guaranteed flows and provides a fair share of the residual bandwidth to all flows. As a future work, we like to apply our proposed algorithm to mobile ad hoc networks.

References

1. A. Demers, S. Keshav, and S. Shenker: Analysis and simulation of a fair queueing algorithm. ACM SIGCOMM, (1989) 1–12
2. S. Lu, V. Bharghavan, and R. Srikant: Fair scheduling in wireless packet networks. IEEE/ACM Transaction On Networking Vol. 7 (1999) 473–489
3. T. S. Ng, I. Stoica, and H. Zhang: Packet fair queueing algorithms for wireless networks with location-dependent errors. IEEE INFOCOM (1998) 1103–1111
4. P. Ramanathan and P. Agrawal: Adapting packet fair queueing algorithms to wireless networks. ACM MOBICOM (1998) 1–9
5. H. Luo and S. Lu: A self-coordinating approach to distributed fair queueing in ad hoc wireless networks. IEEE INFOCOM (2001) 1370–1379
6. H. Luo and S. Lu: A topology-independent fair queueing model in ad hoc wireless networks. IEEE Int. Conf. Network Protocols (2000) 325–335
7. H. L. Chao and W. Liao: Credit-based fair scheduling in wireless ad hoc networks. IEEE Vehicular Technology Conf. (2002)
8. Jerry Cheng and Songwu Lu: Achieving Delay and Throughput Decoupling in Distributed Fair Queueing Over Ad Hoc Networks. IEEE ICCCN (2003)
9. H. L. Chao and W. Liao: Fair Scheduling With QoS Support in Wireless Ad Hoc Networks. IEEE Trans. on Wireless Comm., vol. 3 (2004)
10. P. Goyal, H.M. Vin and H. Chen: Start-time fair queueing: A scheduling algorithm for integrated service access. ACM SIGCOMM (1996)
11. C. Perkins, E. Belding-Royer and S. Das: Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561 (2003)
12. V. Bharghavan, A. Demers, S. Shenker, and L. Zhang: MACAW: A Medium Access Protocol for Wireless LANs. ACM Ann. Conf. Special Interest Group on Data Comm. (SIGCOMM) (1994)
13. T.S.E. Ng, I. Stoica, H. Zhang: Packet Fair Queueing Algorithms for Wireless Networks with location-Dependent Errors. IEEE INFOCOM (19998)
14. S. Lu, V. Bharghavan and R Srikant: Fair Scheduling in Wireless Packet Networks. IEEE/ACM Transaction. On Networking vol. 7 (1999)
15. H. L. Chao and W. Liao: Fair Scheduling in Mobile Ad Hoc Networks with Channel Errors. IEEE Transaction of Wireless Communications, Vol 4 (2005)
16. M. M. Alam, M. M. Rashid and C. S. Hong: Distributed Coordination and Fair Queueing in Wireless Ad Hoc Networks. Lecture Notes in Computer Science, Vol. 3981 (2006)