On Improving Service Differentiation under Bursty Data Traffic in Wireless Networks

Hao Zhu and Guohong Cao
Department of Computer Science & Engineering
The Pennsylvania State University
University Park, PA 16802
E-mail: {hazhu, gcao}@cse.psu.edu

Abstract— **The fair queuing model has been widely used to provide QoS for flows sharing a wireless channel. In fluid fair queuing, a flow cannot reclaim its service loss due to absence. As a result, fair queuing models that emulate fluid fair queuing cannot provide good service differentiation under bursty data traffic. On the other hand, strict priority queuing (SPQ) can provide good service differentiation at the cost of QoS provision. To achieve both service differentiation and QoS provision, we propose a new service model called Absence Compensation Fair Queuing. The basic idea is to allow a flow to get compensation of its service loss due to absence. Since the proposed service model is based on fair queuing, QoS provision is guaranteed. We first verify these properties by analysis, then evaluate the performance compared to weighted fair queuing (WFQ) and SPQ. Simulation results show that our service model can provide much better service differentiation than WFQ, and outperforms SPQ in terms of QoS provision.**

Index Terms: Service differentiation, simulations, QoS, scheduling, wireless networks.

I. Introduction

In recent years, there has been an unprecedent growth in the wireless industry. In addition to the traditional voice services, wireless service providers are trying to provide data services over wireless networks. Since the capacity of wireless networks is still less than that of wired networks [8], [16], the wireless part will continue to be the performance bottleneck, and we need to carefully manage the bandwidth of wireless links. One widely used bandwidth management approach is to apply the wireline fair queuing schemes (e.g., weighted fair queuing (WFQ) [3], [18]) to the wireless environment considering the characteristics of wireless channels such as time-varying channel conditions.

Most previous research on fair queuing for wireless networks ([2], [10], [11], [12], [13], [14]) focus on how to provide fair service to the flows and balance the tradeoff between fairness and system throughput. With these models, in the long run, the extent of service differentiation in terms of throughput [2], [13], [14] or time fraction [10], [11], [12] between any two flows should be approximately equal to the ratio of the service weight of the flows. However, this expectation may not be true since the extent of service differentiations also depends on the type of traffics. As studied in [8], under bursty data traffic, when the network utilization is not very high, WFQ only provides little service differentiation to the flows. The reason is as follows. When a flow is *absent*; i.e., the flow does not have any data in its queue, the WFQ scheduler distributes the service that belongs to the absent flow (if it were backlogged) to all backlogged flows in the system. As a result, backlogged low-weight flows could get many extra services from the absent high-weight flows and then their actual data rate could be very high. Since the WFQ model emulates *Fluid Fair Queuing* (FFQ) [3] which does not compensate the service loss of a flow due to absence, after the low-weight flows take extra service from the absent high-weight flows, these high-weight flows cannot reclaim the service loss when they become backlogged again. Thus, under bursty data traffic, WFQ can only provide good service differentiation when the network utilization is very high, in which case almost all flows are backlogged.

This phenomenon reflects the gap between network-level fairness and application-level fairness under bursty data traffic. Since most of the current wireless fair queuing schemes follow the principle of FFQ, they also suffer from the same problem. One simple solution to increase service differentiation under bursty data traffic is to use the *strict priority queuing* (SPQ) model [4]. Under SPQ, high-weight flows are granted the exclusive priority over flows with low weights. Therefore, low-weight flows cannot be served whenever there are some backlogged high-weight flows in the system. It is easy to see that SPQ can achieve the maximum service differentiation between high-weight and low-weight flows. However, low-weight flows do not have any QoS guarantee under SPQ since they may be starved if there are backlogged high-weight flows.

To achieve better service differentiation with QoS provision, we propose a new service model called *Absence Compensation Fair Queuing* (ACFQ). Fundamentally different from other wireless fair queuing models, the ACFQ scheduler accounts for the service loss or gain due to both channel errors and absence. The flow with service gain will relinquish part of its service to another flow with service loss. With absence compensation, the service differentiation between high-weight and low-weight flows is much better than that of WFQ under bursty data traffic. Besides the absence compensation model, we also

This work was supported in part by the National Science Foundation (CAREER CCR-0092770 and ITR-0219711).

design an error compensation model, which opportunistically exploits channel conditions to increase the system throughput with the constraint of fairness. Since ACFQ is on the basis of fair queuing, each flow can still have QoS guarantees. We provide analytical properties of ACFQ, and evaluate its performance by simulations. Simulation results show that ACFQ can provide much better service differentiation and higher system throughput than WFQ, and outperforms SPQ in terms of QoS provision.

The rest of the paper is organized as follows. Section II develops the necessary background. In Section III, we present the ACFQ service model. Analytical properties of ACFQ will be given in Section IV. In Section V, we evaluate the performance of ACFQ. Section VI summarizes some related work. Section VII concludes the paper.

II. Preliminaries

A. The System Model

We only consider the wireless part of the communication system, where mobile users communicate with wireless base stations directly. The wireless link is accessed in TDMA and is managed by the base station. The packet scheduling algorithms discussed in this paper allocate time slots to packets of users within the coverage area of the base station. We assume the majority of the traffic is from the base stations to the mobile users, and the sole congestion point of the system is the downlink. Similar to many existing works [10], [13], we assume that the base station has a way to obtain the channel condition. Based on the information of channel conditions, the system (e.g., EDGE [16]) selects the most suitable modulation and coding scheme to transmit the impending packet. For each time slot, there is a set of possible transmission rates $\{0, C^1, C^2, ..., C^M\}$. We assume the service provider uses the scheduler to implement the *Olympic service model* [4], which assigns different service weights to different QoS (i.e., the 'Gold', 'Silver', and 'Bronze'.)

B. Fair Queuing Model

Fair queuing was originally developed as an attempt to maintain fairness among competing flows. It serves flows in proportion to their pre-specified weights, and isolates the misbehaving flows. Most recent studies in fair queuing are motivated by FFQ. FFQ guarantees that for any time period $[t_1, t_2]$, any two *backlogged* flows f_i and f_j have the same amount of normalized service, i.e., $W_i(t_1, t_2)/r_i = W_j(t_1, t_2)/r_j$, where $W_i(t_1, t_2)$ is the service (in bits) received by f_i and r_i is the weight of f_i. However, since network schedulers serve flows at the granularity of packet, and the service is non-preemptive, the service of each flow cannot be counted in bits. Therefore, all the existing packetized fair queuing models [1], [3], [5], [6], [15] try to emulate FFQ in the unit of packet and have bounded $|W_i(t_1, t_2)/r_i - W_j(t_1, t_2)/r_j|$. In the long run, these service models behave similarly in term of providing fair service to flows with different weights.

III. The Absence Compensation Fair Queuing (ACFQ) Service Model

A. Overview of ACFQ

In order to improve service differentiation and provide QoS, the ACFQ service model is designed with the following objectives:

1) The model should be backward compatible to the existing fair service models in terms of QoS provision. Therefore, we design the new service model based on the wireline WFQ considering the characteristics of wireless networks.
2) The model should consider service losses due to absence and channel errors. The absence compensation model targets at improving service differentiation under bursty data traffic, whereas the error compensation model is used to exploit channel conditions to increase the system throughput with fairness constraints.
3) The model should be simple, elegant, and easy to implement. The extent of the action of compensations should be flexibly controlled by the network administrator.

The ACFQ model consists of three parts: the base model that is based on WFQ to provides QoS to each flow, the accounting mechanism that tracks the service gain/loss due to absence or channel errors, and the compensation model that improves the service differentiation and provides fair service by letting flows with service gain relinquish part of their services to flows with service loss.

B. The Base Model

The base model is used for providing QoS for each flow and acting as the reference system to account for the service loss or gain. By comparing the amount of received service with that of the reference system, a flow can be in three status: *losing, gaining, normal*. A flow is *losing* if it has received less service than it would have received in the reference system, *gaining* if it has received more, and *normal* if it has received the same amount. We choose the wireline WFQ model as the base model to assign a rate weight r_i to each flow f_i. We assume that all weights are normalized based on the smallest weight so that $r_i \geq 1$. The j^{th} packet of f_i, denoted by p_i^j, is assigned a start tag $S(p_i^j)$, and a finish tag $F(p_i^j)$ according to:

$$S(p_i^j) = \begin{cases} V(a_i^j), & f_i \text{ is absent} \\ F(p_i^{j-1}), & f_i \text{ is backlogged} \end{cases} \quad (1)$$

$$F(p_i^j) = S(p_i^j) + \frac{l_i^j}{r_i} \quad (2)$$

where a_i^j is the arrival time of p_i^j and $V(t)$ is the virtual time of the system. The relationship between the virtual time and real time conforms to:

$$\frac{dV(t)}{dt} = \frac{C(t)}{\sum_{j \in B(t)} r_j}$$

where $C(t)$ is the channel capacity at time t and $B(t)$ is the set of flows that are backlogged at time t. The scheduler picks

up the packet for service in increasing order of the associated finish tag.

C. The Accounting Mechanisms

In ACFQ, we need to establish the service account for each flow to keep track of the service loss or gain due to absence or channel errors. To decouple the service gain/loss due to absence and channel error, we establish the accounting mechanism for them separately.

1) Error Accounting: When the channel condition of the serving flow is poor, continually serving the flow will decrease the system throughput. In order to alleviate the impact of the poor channel condition, it is better to swap the time slot from the flow suffering channel errors to another flow with clean channel and compensate the former flow later. With ACFQ, each flow f_i is associated with an error credit (denoted by EC_i), which is bounded by $[-EC_{max}, EC_{max}]$. Since we assume that the channel has multi-rate capability, the amount of data transmitted in different time slot may be different. Suppose a time slot is swapped from f_i to f_j and f_j transmits the packet of b bits, EC_i is decreased by b and EC_j is increased by b.

2) Absence Accounting: One simple way to accumulate the service loss because of absence is to calculate the finish tag for each packet as:

$$F(p_i^j) = \frac{L_i^j}{r_i} + F(p_i^{j-1})$$

with $F(p_i^0) = 0$. With this scheme, whenever a packet p_i^j is served, its finish tag is updated to the total normalized service provided to f_i. Therefore, f_i's service loss due to absence can be expressed by the normalized service difference between f_i and the flow being served. As discussed in [5], [15], [18], if the scheduler always serves the packet with the minimum finish tag, a backlogged flow with large finish tag can be starved for a long time when some new flows join the system. One modification mentioned in [19] is to replace the aforementioned $F(p_i^j)$ by $max\{F(p_i^j), a_i^j\}$, where a_i^j is the arrival time of p_i^j. However, as discussed in [5], since the real time a_i^j is not a true representation of the work progress in the system upon arrival of p_i^j, this solution still cannot avoid blocking backlogged flows with large normalized service.

Our approach: We present a new approach to calculate the service gain/loss due to absence. In our approach, each flow in the system is assumed to be virtually backlogged, and the scheduler needs to insert a *virtual packet* to a flow *whenever* it actually becomes absent. To support this mechanism, each flow is required to *join* or *leave* the system explicitly so that the scheduler can stop tracking service gain/loss due to absence of the flow. Like a real packet, the virtual packet is assigned a finish tag as follows:

$$F(p_i^j) = \frac{L_{vp}}{r_i} + F(p_i^{j-1}) \quad (3)$$

where L_{vp} is the virtual packet length that is pre-specified by the system. Under ACFQ, each flow has an absence credit AC_i bounded by $[-AC_{max}, AC_{max}]$. We can keep track of the service loss due to absence as follows. When the scheduler picks up a virtual packet of f_i, and swaps the time slot to serve the real packet of another flow f_j, AC_i is decreased by 1 and AC_j is increased by 1.

Three important things need to be mentioned here: First, at any time, there is at most one virtual packet in the queue of each flow, and it must stay at the head of the queue. Second, there is no need to actually allocate a space for the virtual packet in the queue, and the queue only needs to keep the finish tag of the virtual packet. Third, the use of virtual packets does not have any side effect on the service tags of real packets since Eq (1) implies that the virtual packet staying in the queue will be discarded if a real packet arrives at the queue.

D. The Compensation Model

After knowing which flow has how many service loss or gain, the gaining flow should decide whether to use the allocated time slot or relinquish it to other losing flows. There are several options to relinquish service gains. One simple way is to relinquish all service gains, which means that the gaining flow f_i cannot transmit any data until its credits become zero. However, if the credits are large, f_i may be starved. While bounding credits provides a partial solution, we present a more elegant solution. Our compensation model is based on the parameter which is called *relinquish probability*. When the scheduler selects a flow, it decides whether to let the flow relinquish the time slot or not according to the relinquish probability. We decouple the absence compensation and the error compensation by defining different ways to calculate the related relinquish probability.

Error Compensation: The goal of the error compensation model is to exploit the channel utilization to achieve high system throughput without loss of the long-term fairness of each flow. In order to efficiently exploit the channel utilization, the speed of the compensation should not be too fast. In other words, reducing the compensation speed may give the scheduler more opportunities to exploit the system throughput by serving each lagging flow at the time when it has good channel conditions. We use $\alpha_i = X_i/C$ to represent the channel condition of flow i , where X_i is the available transmission rate of f_i[1]. Under ACFQ, the channel condition and the error credit of each flow are the parameters of the error compensation model. The *error relinquish probability* of flow f_i, denoted by $rel_{err}(i)$ is defined as follows:

$$rel_{err}(i) = \begin{cases} max\{0, 0.5\frac{EC_i}{EC_{max}} + 0.625(1-\alpha_i)\}, & \alpha_i > 0 \\ 1.0, & \alpha_i = 0 \end{cases} \quad (4)$$

When the channel is clean (e.g. $\alpha = 1.0$), only the gaining flow has positive rel_{err}. On the other hand, when the channel

[1]Without loss of generality, we assume that α_i is no less than 0.2 when $\alpha_i > 0$ in this paper.

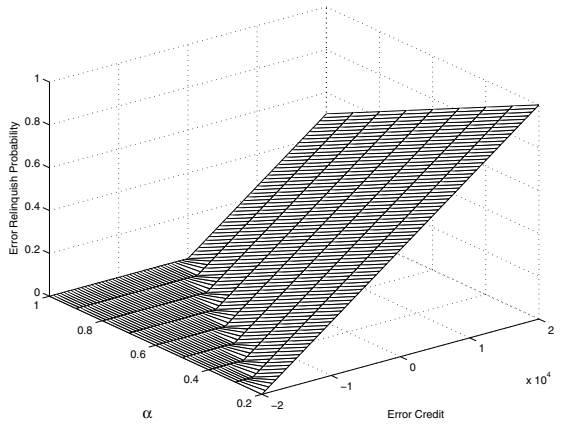


Fig. 1. The error relinquish probability as a function of the error credit and α

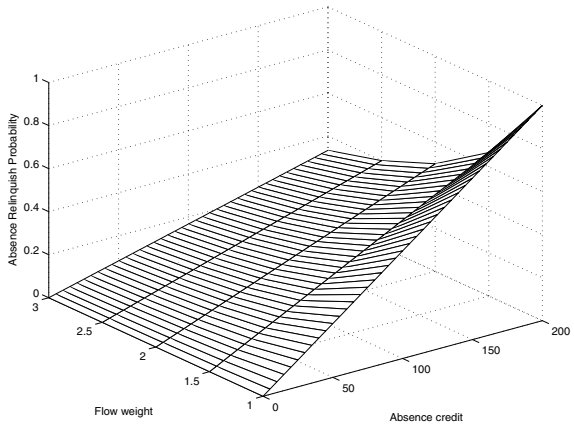


Fig. 2. The absence relinquish probability as a function of the absence credit and the flow weight

quality is poor (e.g., $\alpha = 0.2$), the lagging flow may still relinquish the time slot, since serving the flow will significantly degrade the system throughput. However, the service delay is bounded since the error credit of the flow will eventually be sufficiently small to reduce the relinquish probability. Figure 1 gives the numerical results of rel_{err} as a function of the error credit and α, where EC_{max} is assumed to be 2×10^4 bits.

When a flow relinquishes the time slot, the time slot is used to serve another flow. Again, we need to consider the channel quality and the error credit of each flow to balance the tradeoff between system throughput and fairness. Under ACFQ, each flow f_i is assigned an *error compensation value* val_i, which is defined by:

$$val_i = \begin{cases} -sign(EC_i)(\frac{EC_i}{EC_{max}})^2 + \alpha_i, & \alpha_i > 0 \\ -\infty, & \alpha_i = 0 \end{cases} \quad (5)$$

where $sign(x) = 1$ for $x \geq 0$, and $= -1$ for $x < 0$. Suppose flow f_i decides to relinquish a time slot, the candidate flow f_j to be compensated is selected as following:

$$f_j = \underset{k \in \mathcal{B} \wedge val_k > val_i}{\arg\max} \{val_k\}; \quad (6)$$

where $\mathcal{B}$ is the set of backlogged flows.

Absence Compensation: Suppose flow f_i is allocated a time slot, the scheduler first calculates the *absence relinquish probability* of f_i, denoted by $rel_{abs}(i)$, according to:

$$rel_{abs}(i) \begin{cases} \frac{AC_i}{AC_{max} * r_i}, & AC_i \geq 0 \\ 0, & AC_i < 0 \end{cases} \quad (7)$$

Unlike the calculation of error relinquish probability, we consider the absence credit and the weight of f_i to determine $rel_{abs}(i)$, since the goal of absence compensation is to improve the service differentiation. With the same amount of absence credit, high-weight gaining flows have smaller relinquish probability than low-weight gaining flows. In Figure 2, we give numerical results of the absence relinquish probability as a function of the absence credit and flow weight. According to the obtained $rel_{abs}(i)$, the scheduler makes a decision as follows:

- **Case 1:** The head-of-line packet of f_i is a real packet. Depending on $rel_{abs}(i)$, it decides to allocate the time slot to serve f_i or let f_i relinquish the time slot to the other backlogged losing flows. If it decides to let f_i relinquish the time slot but no backlogged losing flow exists, the scheduler further examines weather to serve f_i or not according to $rel_{err}(i)$.
- **Case 2:** The head-of-line packet of f_i is a virtual packet, i.e., the queue of f_i is empty. Depending on $rel_{abs}(i)$, the time slot is allocated to a backlogged losing flow or the backlogged flow whose head-of-line packet has the minimum finish tag among all the real packets. If there is no backlogged flow, the time slot is wasted.

To select a backlogged flow for absence compensation, the scheduler picks up the candidate losing f_j with the minimum weighted absence credit as follows:

$$f_j = \underset{k \in \mathcal{B} \wedge AC_k < 0 \wedge \alpha_k > 0}{\arg\min} \{AC_k * r_k\} \quad (8)$$

With the same amount of credit, high-weight losing flows are compensated much faster than the low-weight losing flows, which is helpful to improve the service differentiation.

E. The ACFQ Scheduler

In the ACFQ service model, the scheduler serves the packets in the increasing order of their finish tags. After selecting flow f_i, the scheduler first decides if f_i should relinquish the time slot for absence compensation according to the rules of the absence compensation model. If f_i does not relinquish the time slot, the scheduler further checks the efficiency of serving f_i according to $rel_{err}(i)$ to balance the tradeoff between fairness and system throughput. There exists a tradeoff between the absence compensation and the error compensation. Suppose the time slot is swapped from f_i to f_j for absence compensation, and there is another backlogged flow f_k which has better channel condition than f_j. At this time, serving f_j favors the

Notations:

V**:** system virtual clock
F_i**:** the finish tag of the head-of-line packet of f_i
$\mathcal{K}$**:** the set of all flows
$\mathcal{B}$**:** the set of backlogged flows
j: the flow to be served
ac: the absence compensation indicator
ec: the error compensation indicator

schedule()
begin:
1. ac = **false**; ec = **false**; j = **NULL**; m = **NULL**;
   i = $\underset{k\in\mathcal{K}}{argmin}\{F_k\}$; /* initializations */
select:
2. **if** (f_i is backlogged)
3. **if** $(random(0,1) < rel_{abs}(i))$
4. select f_j according to Eq (8);
5. **if** (f_j not exists) $j = i$;
6. **else** ac= **true**;
/* check whether to relinquish the slot due to error */
7. **if** $(ac == false \wedge$
$(random(0,1) < rel_{err}(i))$
8. select f_j according to Eq (6);
9. **if** (f_j not exists) j=i;
10. **else** ec=**true**;
11. **else** /* f_i is absent */
12. **if** $(random(0,1) < rel_{abs}(i))$
13 select f_j according to Eq (8);
14. **if** (f_j exists) ac=**true**;
15. **else** $m = \underset{k\in\mathcal{B}}{\arg\min}\{F_k\}$;
16. **if** $(m \neq NULL)$ { i=m; **goto select**; }
17. **if** (f_j not exists) {idle in the time slot; **goto begin**;}
18. $p = Q_j.deque()$;
19. **send**(p);
20. **if** $(m \neq NULL)$ { i=m; ac = **true**; }
21. **if** $(ac == true)$
22. **if** (f_i is backlogged) $F_i = F_i + p.length/r_i$;
23. **else** $F_i = F_i + L_{vp}/r_i$;
24. $AC_i = AC_i - 1$; $AC_j = AC_j + 1$;
25. **else if** $(ec == true)$
26. { $F_i = F_i + p.length/r_i$;
27. $EC_i = EC_i - p.length$;
28. $EC_j = EC_j + p.length$;
29. } **else if** $(Q_j.legnth > 0)$
30. update F_j according to Eq (2);
31. **else** $F_j = F_j + L_{vp}/r_j$;
32. **goto begin**;

Fig. 3. The ACFQ algorithm

service differentiation, whereas the system throughput can be increased by serving f_k. Since we focuses on improving the service differentiation, we choose to serve f_j in this case. Thus, the scheduler does not further evaluate the channel condition of f_j (see line 7 in Figure 3).

When f_i is absent and the scheduler selects flow f_m that has the minimum finish tag among all the backlogged flows, the scheduler treats f_m as a new selected flow and examine whether to serve f_m or not according to the relinquish probabilities of f_m (see line 11 in Figure 3). Finally, it is possible that the time slot is swapped in the form of a chain; i.e., $f_i \rightarrow f_m \rightarrow f_j$. In this case, the service accounting should take place between f_i and f_j since f_j takes the service opportunity of f_i. Although the time slot is swapped from f_m to f_j because of the poor channel condition of f_m, since the service opportunity of f_i is absence related, the service gain of f_j should be counted as the gain due to absence. This is why the absence indicator (ac) is forced to be true in line 20 in Figure 3.

It is worth to study the situation when f_i *leaves* the system with non-zero credit C_i, which can be AC_i or EC_i. In order to make sure that $\sum_{k\in\mathcal{K}} C_k = 0$ is always valid, we need to distribute C_i proportionally to other flows. For example, if f_i with $C_i > 0$ leaves the system, the credits of the losing flows should be increased proportionally. However, since credits are integer, we cannot achieve exact proportional distribution. Our solution is as follows. Suppose $\mathcal{L}$ denotes the set of the losing flows; i.e., $\mathcal{L} = \{l | l \in \mathcal{K} \wedge C_l < 0\}$. After a gaining flow f_i with $C_i > 0$ leaves the system, the credit of each flow in $\mathcal{L}$, denoted by f_j, is increased by $\lfloor \frac{C_j}{\sum_{l\in\mathcal{L}} C_l} C_i \rfloor$. Then, we get

$$\Delta = C_i - \sum_{j\in\mathcal{L}} \lfloor \frac{C_j}{\sum_{l\in\mathcal{L}} C_l} C_i \rfloor \tag{9}$$

If $\Delta > 0$, we randomly generate the subset of $\mathcal{L}$, denoted by $\mathcal{SC}$, with the cardinality of Δ. The credit of each flow in $\mathcal{SC}$ is increased by 1. As a result, the credit of f_j is increased by $\lfloor \frac{C_j}{\sum_{l\in\mathcal{L}} C_l} C_i \rfloor$ or $\lfloor \frac{C_j}{\sum_{k\in\mathcal{L}} C_k} C_i \rfloor + 1$. The correctness is based on the fact that $0 \leq C_i - \sum_{j\in\mathcal{L}} \lfloor \frac{C_j}{\sum_{l\in\mathcal{L}} C_l} C_i \rfloor < |\mathcal{L}|$ when $C_i > 0$. Similar approach can be applied when $C_i < 0$.

For the computational complexity of the ACFQ model, we observe that the selection of the candidate flow is at the cost of $O(log(n))$, which is feasible in many wireless networks that have a moderate number of flows per base station. The main computational overhead of ACFQ is divisions to compute the relinquish probabilities. According to the scheduling scheme, to serve a packet, the scheduler needs to calculate the probability at most three times. For most current microprocessors (>1GHz), the cost is less than 150ns since one floating devision needs less than 50 cycles [4]. Thus, this delay would not be an issue for base stations.

IV. Analysis of ACFQ

In this section, we analyze the properties of ACFQ. For simplicity, we assume all real packets have the same fixed

size L_p and the channel is error-free.

Lemma 1: Consider a gaining flow f_i over a time interval $[t_1, t_2]$. Suppose $AC_i(t_1) = AC_0$, where $AC_i(t_1)$ is the current absence credit of f_i at t_i. Assume there always exists another flow which can take the compensation slot whenever f_i relinquishes the slot during $[t_1, t_2]$. Then, for any $t \in [t_1, t_2]$, the expected value of credit $AC_i(t)$, denoted by $E(AC_i(t))$, is bounded by:

$$AC_0 exp(-\frac{C}{AC_{max} r_j}(t - t_1)) < E(AC_i(t)) \leq AC_0 exp(-\frac{\phi_i C}{AC_{max} r_j}(t - t_1))$$

where $\mathcal{K}$ is the set of all flows, $\phi_i = \frac{r_i}{\sum_{j \in \mathcal{K}} r_j}$, and C is the link capacity.

Proof: At time $t \in [t_1, t_2]$, suppose f_i has the data rate $\hat{r_i}(t)$ (in bps) under the FFQ model. As described in the absence compensation model, the expected *actual* data rate of f_i, denoted by $E(\hat{r_i}(t))$, is adjusted as: $E(\hat{r_i}(t)) = \hat{r_i}(t)(1 - E(AC_i(t))/(AC_{max} r_i))$. Thus, the expected rate of service compensation for f_i follows:

$$\frac{dE(AC_i(t))}{dt} = -\frac{E(AC_i(t))\hat{r_i}(t)}{AC_{max} r_i}$$

Since $\phi_i C \leq \hat{r_i}(t) < C$, where $\phi_i = \frac{r_i}{\sum_{j \in \mathcal{K}} r_j}$, we have:

$$-\frac{C}{AC_{max} r_i} dt < \frac{dE(AC_i(t))}{E(AC_i(t))} \leq -\frac{\phi_i C}{AC_{max} r_i} dt \qquad (10)$$

By integration over $[t_1, t]$, we arrive at:

$$AC_0 exp(-\frac{C}{AC_{max} r_i}(t - t_1)) < E(AC_i(t)) \leq AC_0 exp(-\frac{\phi_i C}{AC_{max} r_i}(t - t_1))$$

□

Lemma 1 shows the graceful absence compensation of a leading flow.

Theorem 1: Consider a losing flow f_i over a time interval $[t_1, t_2]$. Assume that all the flows are continuously backlogged during $[t_1, t_2]$ and $\forall (k \in \mathcal{K}) AC_k(t_1) = AC_k^0$. Its expected aggregated service is bounded by:

$$E(W_i(t_1, t_2)) \geq \phi_i C(t_2 - t_1) - \phi_i |\mathcal{K}| L_p - L_p + SL_p \qquad (11)$$

where S is calculated by:

$$S = \arg\max_{n=0,1,2...} \{\sum_{m=0}^{n} \sum_{l \in \mathcal{L} \wedge l \neq i} \lceil AC_i^m \frac{r_i}{r_l} - AC_l^m \rceil^+ + n \leq \sum_{j \in \mathcal{G}} AC_j^0 (1 - exp(-\frac{\phi_j C}{AC_{max} r_j}(t_2 - t_1)))\} \qquad (12)$$

where $\lceil x \rceil^+$ is defined as: max(0,$\lceil x \rceil$), $\mathcal{L} = \{l | l \in \mathcal{K} \wedge AC_l < 0\}$, $\mathcal{G} = \{j | j \in \mathcal{K} \wedge AC_j > 0\}$. For each flow f_l in $\mathcal{L}$, AC_l^m is defined as:

$$AC_l^m = \begin{cases} AC_l^m = AC_l^{m-1} + \lceil AC_l^m \frac{r_i}{r_l} - AC_l^{m-1} \rceil^+, \\ \qquad m > 0 \wedge l \neq i \\ AC_l^m = AC_k^{m-1} + 1, \quad m > 0 \wedge l = i \end{cases} \qquad (13)$$

Proof: With Lemma 1, we can see that the expected total number of slots relinquished by all the gaining flows is no less than: $\sum_{j \in \mathcal{G}} AC_{j,0}(1 - exp(-\frac{\phi_j C}{AC_{max} r_j}(t_2 - t_1)))$. According to Eq (8), at time t_1, before the losing flow f_i can be compensated with a time slot, the number of relinquished time slots that have passed is: $\sum_{l \in \mathcal{L} \wedge l \neq i} \lceil AC_{i,0} \frac{r_i}{r_l} - AC_{l,0} \rceil^+$. Following the same reasoning, after f_i gets its m^{th} relinquished time slot, the absence credit of each losing flow f_l can be computed as:

$$AC_l^m = \begin{cases} AC_l^{m-1} + 1 & l = i \\ AC_l^{m-1} + \lceil AC_i^{m-1} \frac{r_i}{r_j} - AC_l^{m-1} \rceil^+ & l \neq i \end{cases}$$

Hence, the total number of relinquished slots that f_i can get during $[t_1, t_2]$ can be computed by Eq (12). Similar to the proof of Theorem 4.2 in [13], if no compensation occurs during $[t_1, t_2]$, the aggregated service of f_i during the period follows:

$$W_i(t_1, t_2) \geq \phi_i C(t_2 - t_1) - \phi_i |\mathcal{K}| L_p - L_p \qquad (14)$$

Since f_i gets two sources of service: the allocated service according to r_i and the compensated service from the gaining flows, with Ineq (14) and Eq (12), we conclude the proof. □

Theorem 1 shows the expected throughput guarantee for a backlogged losing flow during a time period $[t_1, t_2]$.

V. Performance Evaluation

In this section, we evaluate the performance of the proposed ACFQ and demonstrate its effectiveness by comparing to strict priority queuing (SPQ) [4] and weighted fair queuing (WFQ) [3]. The simulation is based on ns-2 [7]. Following the results of [17], when the channel is error-prone, a five-state Markov chain is used to emulate the process of the channel condition with fast fading. As shown in Figure 4, the

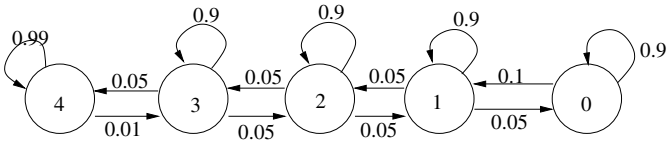


Fig. 4. The error-prone channel model

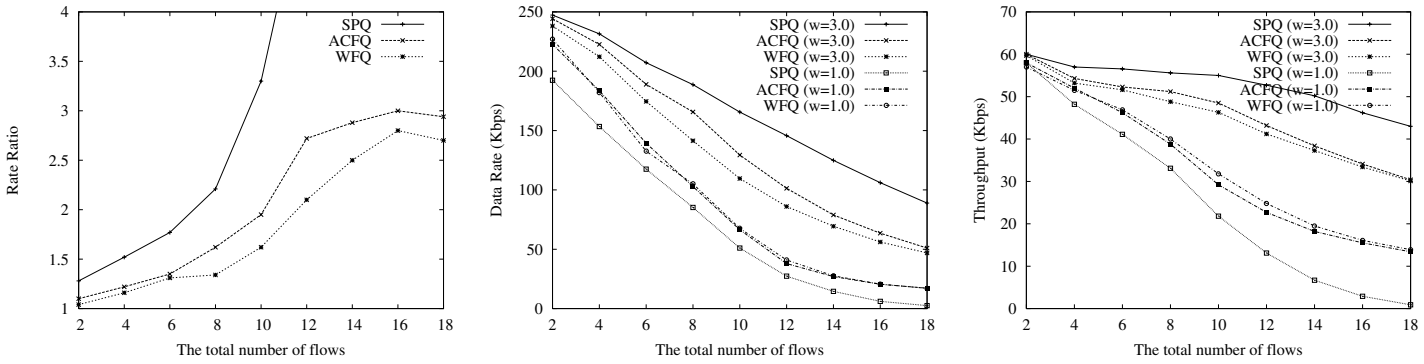
marked line or curve shows the transition probability from one state to another. The wireless link is based on TDMA. The transmission rate in state 4 is equal to the channel capacity which is assumed to be $384Kbps$, and is zero when the channel condition is in state 0. Following the standard of EDGE [16], each time slot is $2.5ms$, which can be used to transmit 112, 74, 56, 44 bytes of data (not including the header) in state 4, 3, 2, 1 respectively.

Since we focus on improving service differentiation under bursty data traffic, Web browsing is used as a case study. In order to reduce the overhead of TCP hand-shaking and slow start, we assume that all connections are persistent [9]. By setting different seeds, we generate different traffic trace for each flow. Similar to [8], the parameters of the data traffic model are shown in table I. For simplicity, we assume that there are two kinds of flows in the system: high-weight flows

Component	Model	PDF	Parameters
File Sizes - Body	Lognomal	$p(x) = \frac{1}{x\sigma\sqrt[2]{2\pi}} e^{-(ln(x)-\mu)^2/2\sigma^2}$	$\mu = 7.881; \sigma = 1.339$
File Sizes - Tail	Pareto	$p(x) = \alpha k^\alpha x^{-(\alpha+1)}$	$k = 34102; \alpha = 1.177; max = 100Kbytes$
Embedded References	Pareto	$p(x) = \alpha k^\alpha x^{-(\alpha+1)}$	$k = 2; \alpha = 1.245; max = 30$
OFF Times	Pareto	$p(x) = \alpha k^\alpha x^{-(\alpha+1)}$	$k = 1; \alpha = 1.4; max = 300sec$

TABLE I

DISTRIBUTIONS AND PARAMETERS OF THE TRAFFIC MODEL



(a) Rate ratio (b) Data rate (c) Throughput

Fig. 5. Performance comparisons under three service models

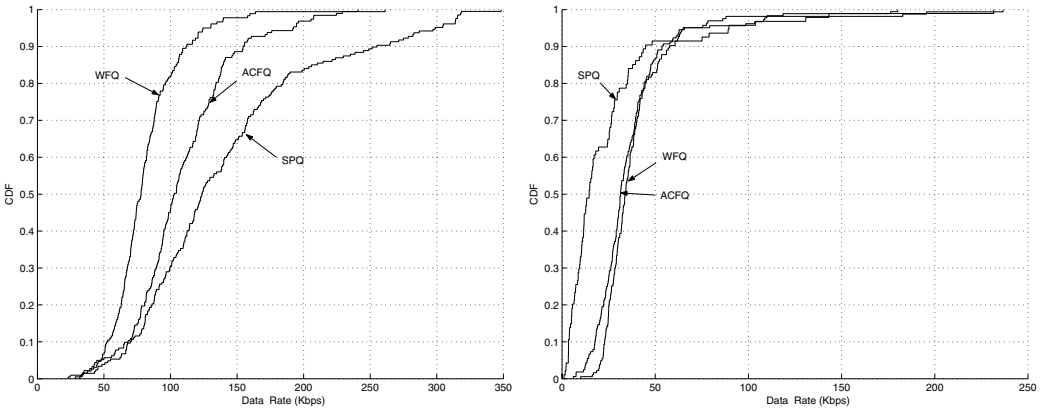
with weight of 3.0 and low-weight flows with weight of 1.0. We assume that the number of high-weight flows and low-weight flows are equal. The total simulation time is 1000 seconds. We set the default AC_{max} and EC_{max} to be 200 and 20 KB respectively. We set the default virtual packet length (L_{vp}) to 1000 bytes, and further evaluate its impacts in Section V-C.

We evaluate the system performance with the following three metrics: the *average data rate*, the *average throughput*, and the *rate ratio*. The data rate of a file is calculated by dividing its file size with the time used to deliver the entire file, which shows how fast the file is delivered. The throughput is obtained by dividing the total amount of data (in bits) delivered by the simulation time. Since there may be multiple flows with the same weight and each flow may have different data rate, we use the *average data rate* to measure the application-level QoS. Similarly we use the *average throughput* instead of throughput for each flow. For simplicity, we will remove $average$ from these terms in the follows of the paper. The *rate ratio* is the ratio of the data rate of the high-weight flows to that of the low-weight flows, and is used as the metrics of the service differentiation. The evaluation considers three scenarios. In the first scenario, the channel is error-free. In the second scenario, the performance is evaluated in the error-prone channel. Finally, we examine the impacts of virtual packet length on the performance of ACFQ.

A. Scenario 1: Error-free Channel

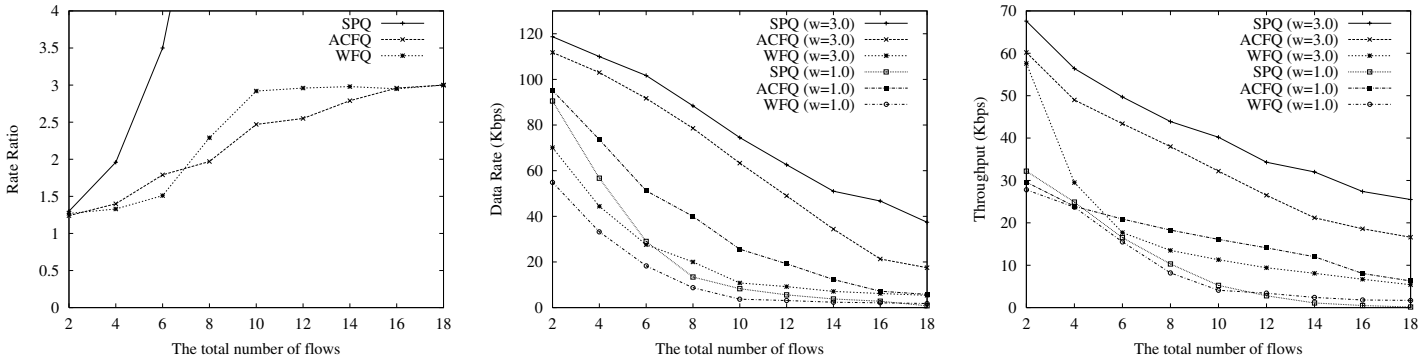
We evaluate the performance of these three service models under different workload, which is represented by the total number of flows, denoted by $N(2 \leq N \leq 18)$. As shown in Figure 5 (a), the rate ratio under SPQ is always higher than that under WFQ and ACFQ. As N is larger than 10, the rate ratio of SPQ becomes much larger than 3.0, which is the ideal rate ratio. It indicates that the high-weight flows get too much service that is not proportional to their weights. This is due to the fact that SPQ gives high-weight flows exclusive priority over the low-weight flows, and cannot provide QoS for the low-weight flows when the traffic of the high-weight flows is heavy. From Figure 5 (a), we can see that the rate ratio of ACFQ is much higher than that of WFQ, especially when N is larger than 6, because ACFQ performs service compensation due to absence, but WFQ does not. In addition, the high-weight flows under ACFQ get preferential treatment under the compensation model. Compared to the low-weight flows, the high-weight gaining flows have slower relinquish rate and the service loss of the high-weight flows can be quickly compensated. As a result, the high-weight flows can transmit their packets at higher speed. This can be further explained by Figure 5 (b) and Figure 5 (c). As shown in the figures, the data rate and the throughput of the high-weight flows under ACFQ are much higher than that under WFQ. When the workload is very heavy $(N = 18)$, the progress of the virtual clock becomes slower (*i.e.,* for the same interval $[t_1, t_2]$, $v(t_2) - v(t_1)$ is smaller). Correspondingly, the absence credit of an absent flow decreases slower, and then the total number of time slots that has to be relinquished reduces. This explained why the data rate difference of the high-weight flows between ACFQ and WFQ when $N = 18$ is smaller than that when $N = 8$.

In order to show the difference of these three models clearly,



(a) high-weight flows (b) low-weight flows

Fig. 6. The CDF of the transmission rate ($N = 12$)



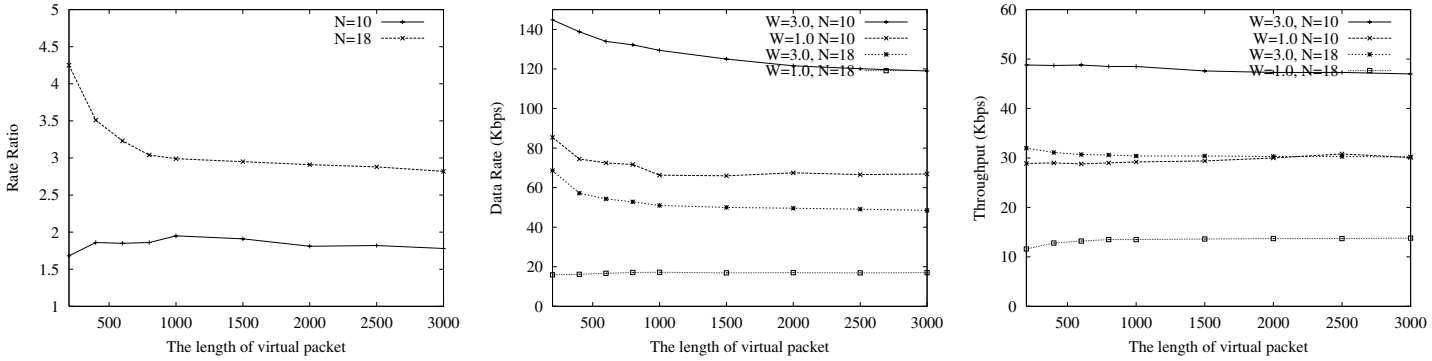
(a) Rate ratio (b) Data rate (c) Throughput

Fig. 7. The performance comparisons with channel errors

we draw the cumulative distribution function (CDF) of the data rate when $N = 12$. As shown in Figure 6, more than 80% of the packets in the high-weight flows are transmitted at a data rate of less than 100 Kbps under WFQ. By compensating the service loss due to absence, the percentage is reduced to be less than 50% under ACFQ, which means that more than 30% of the packets are transmitted faster. Due to exclusive priority, only 30% of the packets in high-weight flows are transmitted less than 100Kbps under SPQ. On the other hand, the CDF of the data rate for low-weight flows under ACFQ is similar to that under WFQ. Compared to the high-weight flows, the low-weight gaining flows have higher relinquish rate and the low-weight losing flows have less opportunities to be compensated. Even though the low-weight flows are also compensated for their service loss, they cannot get as much benefit as the high-weight flows. As shown in Figure 5 (b), the data rate difference of low-weight flows between ACFQ and WFQ is very small. This also shows that, in the long run, the action of absence compensations does not incur much negative impact on the QoS to the low-weight flows. Since ACFQ uses WFQ as the base model, each flow still has QoS provision. This explains why the rate ratio under ACFQ is bounded by 3.0 even when the workload is very heavy as shown in Figure 5 (a).

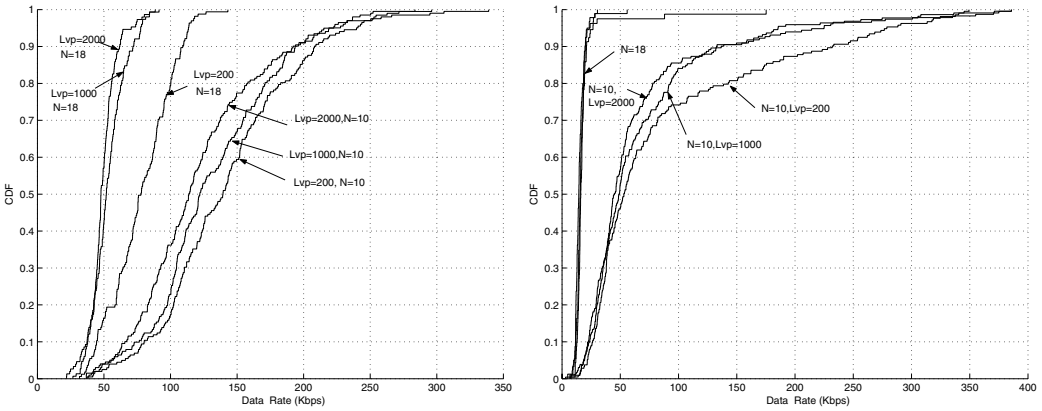
B. Scenario 2: Error-prone Channel

In this scenario, we compare the performance of these service models when the channel is error-prone. We assume that each flow has time-varying channel condition. As shown in Figure 7 (a) and (b), as N increases, the rate ratio of SPQ grows out of bound since the data rate of the low-weight flows drops much faster than that of the high-weight flows. The reason has been explained in Scenario 1 and is till valid here. Since SPQ serves the flows with the same weight in the round robin way, each flow with the same weight can only be served once in each round. This can alleviate the impact of channel errors since the scheduler will skip over the flow when the flow cannot transmit data due to channel errors. As shown in Figure 7 (b) and (c), the performance of WFQ is the worst when channel is error-prone. This is due to the fact that WFQ only tries to achieve the short-term throughput fairness so that the scheduler will continuously serve the flow until the normalized throughput of the flow is no less than that of other flows. As a result, the flow with poor channel condition will take much longer time to get a certain amount of normalized service than the flow with good channel condition. Since we assume that all flows have error-prone channels, the data rates



(a) Rate ratio (b) Transmission rate (c) Throughput

Fig. 8. Performance of ACFQ with different virtual packet length



(a) high-weight flows (b) low-weight flows

Fig. 9. The CDF of the transmission rate ($N = 12$)

of high-weight flows and low-weight flows drop very fast as N increases. However, WFQ still provides fair service to each flow. From Figure 7 (a), the data ratio of WFQ is always bounded by 3.0.

Compared to SPQ and WFQ, ACFQ can improve service differentiation without losing too much system throughput. As shown in Figure 7 (a), ACFQ still has similar rate ratio as in Scenario 1. This shows that the absence compensation model works well in the error-prone channel. From Figure 7 (c), we can see that, when $N \geq 4$, the sum of the throughput of each flow under ACFQ is much higher than that under WFQ, which proves the effectiveness of the error compensation model of ACFQ. Under the fairness constraint, the ACFQ scheduler tries to serve the flow which has the best channel condition so that the system throughput can be increased.

C. Scenario 3: Impact of the Virtual Packet Length

In this scenario, we investigate the impacts of the virtual packet length (L_{vp}) on the performance of ACFQ. We consider two cases: $N = 10$ and $N = 18$, and assume the channel is error-free. As shown in Figure 8, when the workload is not heavy ($N = 10$), the rate ratio and the throughput does not change too much when L_{vp} changes. However, as shown in Figure 8 (b), the date rate of high-weight flows and low-weight flows (lines ($w = 3.0, N = 10$) and ($w = 1.0, N = 10$)) increases when L_{vp} drops. For example, when L_{vp} drops from 1000 to 200, the data rate of high-weight flows increases from 129 Kbps to 145 Kbps and the rate of low-weight flows increases from 67 Kbps to 85 Kbps.

During the same virtual time interval, when L_{vp} decreases, according to the absence accounting mechanism, the absence credit increases. As a result, both high-weight and low-weight flows with service loss can get more chances for service compensations. In particular, because the workload is not heavy ($N = 10$), there are not many high-weight flows competing the service compensation with low-weight flows. Therefore, low-weight flow can be compensated most of the time, and has relatively high data rate. This can be verified from Figure 9 (b). As can be seen, when $L_{vp} = 2000$, less than 15% of the packets in low-weight flows are transmitted at a rate of more than 100 Kbps under WFQ. When L_{vp} is reduced to 200, more than 25% of the packets are transmitted at the rate of more than 100 Kbps.

Things are different when the workload is heavy ($N = 18$). As shown in Figure 8 (a), the rate ratio increases quickly as L_{vp} decreases. For example, when L_{vp} drops from 2000

to 200, the rate ratio increases from 2.9 to 4.3. This can be explained by Figure 8 (b). As can be seen from lines $(W = 3.0, N = 18)$ and $(W = 1.0, N = 18)$, when L_{vp} drops from 2000 to 200, the data rate of high-weight flows increases from 51 Kbps to 69 Kbps, but the data rate of low-weight flows is almost unchanged. This can be further explained by Figure 9. In Figure 9 (b), when $N = 18$, changing the virtual packet length does not affect the data rate of low-weight flows. However, as shown in Figure 9 (a), when $L_{vp} = 2000$, more than 60% of the packets in high-weight flows are transmitted at the rate of less than 50 Kbps; when L_{vp} drops to 200, this percentage is reduced to be less than 20%. This can be explained as follows. When L_{vp} decreases, the absence credit increases. As a result, the number of relinquished time slots for absence compensation increases. However, in contrast to the case when $N = 10$, the low-weight flow is competed with many high-weight flows for service compensations, and cannot get compensations fast enough to increase its transmission rate . On the other hand, most of the increased service compensations (through reduced L_{vp}) are used to compensate high-weight flows. As a result, the data rate of high-weight flows increases when L_{vp} decreases.

VI. Related Work

In [8], Jiang *et. al.* studied the problem of providing multiple service classes for bursty data traffic in cellular networks. Assuming the channel is error-free, they studied the performance of WFQ in providing multiple service classes for typical Internet users in a cellular data network. They found that WFQ can provide differentiated services only to a limited extent due to the burstiness of the data traffic. They also investigated several factors such as propagation delay, persistent TCP connections, and distribution of users, and found that these factors only have little impact on the service differentiation under WFQ. By increasing the weight assigned to high-weight flows, the service differentiation can be improved. However, high-weight flows get too much service when the system is heavily loaded. Our work was stimulated by their work, and we proposed the ACFQ service model to improve service differentiation for bursty data traffic.

In order to provide fairness to the flows with service loss due to channel errors, a channel-condition independent fair model [14] is proposed. Under this model, each leading flow reserves a minimal fraction of service so that the degradation of service for leading flows is graceful. In [13], a different compensation model was proposed. The leading flow achieves graceful service degradation by dynamically adjusting the amount of compensation service based on the service credit. ACFQ uses some similar ideas to [13], [14]. However, ACFQ focuses on improving the service differentiation under bursty traffic with QoS provision. It is different from [13], [14] in the following aspects. First, [13], [14] are based on a two-state channel model, whereas ACFQ considers the channel with the multi-rate capability. Second, the absence compensation model of ACFQ provides preferential treatment to high-weight flows; that is, with the same amount of credit, high-weight gaining flows have smaller relinquish probability, and high-weight losing flows have higher priority to be compensated.

In [4], Dovrolis *el al.* proposed proportional differentiation service models which guarantee the ratio of packet delay difference between classes within the Differentiated Service architecture. They proposed three models to achieve relative QoS among classes: the proportional average delay scheduling, the waiting time priority scheduling, and the hybrid model, which combines the other two. The ACFQ acts somewhat similar to their hybrid model, but our work is different from theirs from the following aspects: First, ACFQ is based on WFQ and targets at improving the service differentiation with absolute QoS provision (in terms of throughput, delay, and long-term fairness) for each flow. Second, ACFQ also considers the time-varying channel condition, which is an important issue in wireless networks.

VII. Conclusions

In this paper, we proposed a new service model, called ACFQ, to improve the service differentiation under bursty data traffic in wireless networks. ACFQ achieves QoS provision on the basis of fair queuing, and improves the service differentiation by using the virtual packet to quantify the amount of service loss/gain due to absence. After the losing flow is backlogged again, it can reclaim the service loss from other gaining flows. The extent of the compensation is gracefully controlled according to the service credit and the service weight of the flow. We also considered channel errors and extended the ACFQ model with a separate error compensation model. By exploiting the channel condition of each flow, the ACFQ scheduler can opportunistically balance the tradeoff between the constraint of fairness and the system throughput. We showed the analytical properties of the ACFQ model, and used simulation to evaluate the performance of the model. Simulation results showed that ACFQ can significantly improve service differentiation without loss of QoS provision.

As future work, we will evaluate the performance of the ACFQ model under mixed traffic; i.e., both bursty traffic and continuous traffic, and study the impacts of the upper bound of the service credits on the performance of the proposed model. It would be interesting to further improve the service differentiation under ACFQ by adaptively changing the absence relinquish probability according to the traffic condition.

References

[1] J. Bennett and H. Zhang, "WF2Q: Worst-case fair weighted fair queueing," *INFOCOM'96*, March 1996.

[2] P. Bhagwat, P. Bhattacharya, A. Krishma and S. Tripathi, "Enhancing throughput over wireless LANs using channel state dependent packet scheduling," *INFOCOM'97*, April 1997.

[3] A. Demers, S. Keshav and S. Shenkar, "Analysis and simulation of a fair queuing algorithm," *ACM SIGCOMM'89*, Sept. 1989.

[4] C. Dovrolis, D. Stiliadis and P. Ramanathan, "Proportional Differentiated Services: Delay Differentiation and Packet Scheduling," *SIGCOMM'99*, pp. 109–120, Sept 1999.

[5] S. Golestani, "A self-clocked fair queueing scheme for broadband applications," *INFOCOM'94*, June 1994.

[6] P. Goyal, H. Vin, and H. Cheng, "Start-Time Fair Queuing: A Scheduling Algorithm for Integrated Serv ices Packet Switching Networks," *ACM SIGCOMM'96*, August 1996.

[7] VINT group, "UCB/LBNL/VINT Network Simulator – ns (Version 2)," *http://mash.cs.berkeley.edu/ns*.
[8] Z. Jiang, L. Chang, and N. Shankaranarayanan, "Providing Multiple Service Classes for Bursty Data Traffic in Cellular Networks," *IEEE INFOCOM'00*, March 2000.
[9] J. Kurose and K. W. Rose, "Computer networking: A top-down approach featuring the Internet, 2nd Edition," *Addison-Wesley Publishers*, 2002.
[10] Y. Liu and E. Knightly, "Opportunistic fair scheduling over multiple wireless channels," *IEEE INFOCOM'03*, 2003.
[11] X. Liu, E. K. P. Chong and N. B. Shroff, "Transmission scheduling for efficient wireless utilization," *IEEE INFOCOM'01*, 2001.
[12] Y. Liu, S. Gruhl and E. Knightly, "WCFQ: an Opportunistic Wireless Scheduler with Statistical Fairness Bounds," *IEEE Transactions on Wireless Communication (to appear)*, vol. 2, no. 5, September 2003.
[13] S. Lu, T. Nandagopal and V. Bharghavan, "A Wireless Fair Service Algorithm For Packet Cellular Networks," *MobiCOM'98*, October 1998.
[14] T.S.Eugene Ng, I. Stoica and H. Zhang, "Packet fair queueing algorithms for wireless networks with location-dependent errors," *INFOCOM'98*, March 1998.
[15] A. Parekh and R. Gallager, "A generalized processor sharing approach to flow control-the single node case," *INFOCOM'92*, June 1992.
[16] B. H. Walke, "Mobile Radio Networks: Networking, Protocols and Tranffic Performa nce, Second Edition," *John Wiley & Sons, Ltd*, 2002.
[17] H. S. Wang and N. Moayeri, "Finite-state Markov channel-a useful model for radio communication channels," *IEEE Transactions on Vehicular Technology*, vol. 44, no. 1, pp. 163–171, Feb 1995.
[18] H. Zhang, "Service Disciplines for Guaranteed Performance Service in Packet-Sw itching Networks," *Proceedings of the IEEE*, vol. 83, no. 10, October 1995.
[19] L. Zhang, "Virtual clock: a new traffic control algorithm for packet switching networks," *ACM SIGCOMM'90*, Sept. 1990.