Robust Packet Scheduling in Wireless Cellular Networks

Zhenghua Fu, Xiaoqiao Meng, Hao Yang, Songwu Lu
Computer Science Department
University of California, Los Angeles, CA 90095
{zfu, xqmeng, hyang, slu}@cs.ucla.edu

***Abstract*— This paper addresses the following scheduling problem in wireless packet switched networks: Given that only coarse-grained channel state information is available, how to design a robust scheduler that ensures worst-case optimal performance? To solve this problem, we take a zero-sum game theoretic approach, in which the scheduler and the channel error act as non-cooperative adversaries in the scheduling process. The scheduler tries to maximize the system revenue from serving flows while the channel errors try to minimize it. We give the optimal scheduling strategy in the heavy channel error case and the mild channel error case separately. Such an optimal scheduling strategy is probabilistic, in the sense that the scheduler makes decision based on a probablistc distribution. Our results show that this probabilistic scheduling may lead to higher worst-case performance compared to traditional deterministic policies, especially when the channel errors are dynamic and difficult to predict.**

I. Introduction

Packet scheduling, which arbitrates packet transmission precedences among multiple contending flows, has long been a popular paradigm to ensure fine-grained service guarantees in the wired network (see [3] for a brief survey). In recent years, researchers have proposed many wireless scheduling algorithms to support service assurance over cellular networks [4] [7][8] [9][10]. All these algorithms address the following scheduling issue: assuming fine-grained channel state (i.e., whether the channel is error-free or error-prone at a given time instant) is available, what is the scheduling policy to provide performance bounds (in terms of fairness or network revenue)?

In this paper, we seek to solve a new scheduling problem: Given that only coarse-grained channel state information (e.g., only bounds on channel errors, but not the exact error pattern that varies over time and location) is available, how to design a scheduler that ensures worst-case performance bounds? The end goal is to maximize the aggregate revenue (collected from all scheduled flows) over all the channel error patterns falling into the specified error bound. This research is mainly motivated by two factors. The first one is that we would like to relax the requirement for accurate fine-grained (e.g., slot-by-slot) channel estimation. In practice, the wireless channel may exhibit a wide range of error patterns, as we show in Section II. It is nontrivial to have reliable channel state estimation, in particular for indoor mobile users. The second factor is that we intend to explore a novel robust scheduling approach other than the popular adaptive/opportunistic scheme [4][7][8][9][10]. If the main issue for wireless scheduling is to address the highly dynamic channel error conditions, both adaptive and robust schemes provide valuable solution approaches. While the adaptive scheme has been well studied, the robust approach is not.

To solve the new scheduling problem, we take a zero-sum game theoretic approach. In this approach, the scheduler and the channel error play a two-player zero-sum game. They act as non-cooperative adversaries in the scheduling process. The scheduler seeks to maximize its aggregate revenue generated by all scheduled flows via scheduling the right flow at the right time and minimizing the effect of channel error corruptions. The channel error seeks to minimize the revenue of the scheduler by corrupting the scheduled flow. This is equivalent to a minimax optimization problem that can be solved using game theory techniques. We solve this problems in both single-slot and multi-slot channel error models and give the robust scheduling policy which can ensure the worst-case performance bounds.

The rest of the paper is organized as follows. Section II describes the problem formulation and our general approach. Section III describes our game theoretic approach based on single-slot channel error model. Sections IV extends this game theoretic approach to multi-slot channel error model. Section V evaluates the design via simulations. Section VI concludes this paper.

II. Problem Formulation and Game-theoretic Approach

A. Network Model

We consider a wireless cellular network. The scheduler is implemented at the base station and it is responsible for scheduling both uplink (mobile-to-base-station) and downlink (base-station-to-mobile) flows over the shared wireless channel. In our model, we divide time into multiple time slots, which are the units for channel allocation, as in related works [7][1]. The base station can only serve one flow in each slot.

We use u_k to denote the reward that the user is willing to pay for scheduling flow k in one slot. This value can also be viewed as the price of flow k per slot. Due to the difference in user's satisfaction and type of application carried by each flow, the price varies among different flows. In our model, the

[1]The underlying MAC protocols can be either TDMA or CSMA/CA. In CSMA/CA, one slot corresponds to the time spent to complete a RTS-CTS-DATA-ACK handshake.

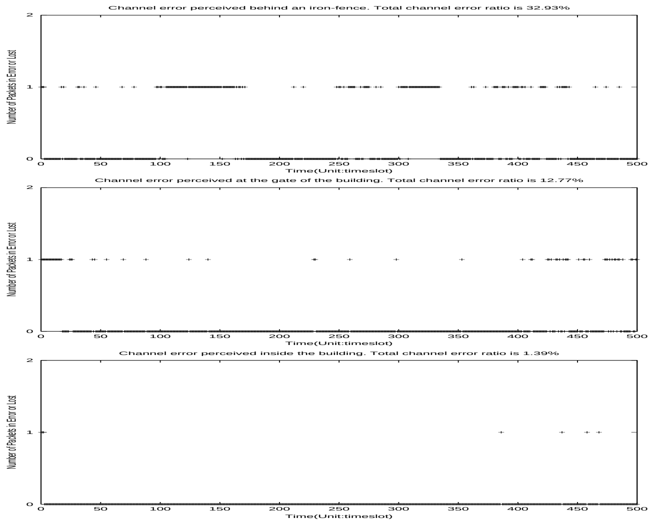

Fig. 1. Channel error patterns measured at three different locations within the range of a access point

base station seeks to optimize its total revenue, which is the sum of rewards collected from all the flows being scheduled.

B. Channel Error Model

Recent studies [11][12] have reported a wide range of wireless channel error patterns, from highly bursty to quite sporadic. For example, the measurements in [12] show that each error burst lasts for more than twenty packets, while [11] reports a quite random pattern with the dominant error length as two or three packets. In order to investigate the realistic channel error pattern, we measured the channel quality in three different locations within a cell in an IEEE 802.11 WLAN. Figure 1 shows our real measurement results, which indicate that the channel error patterns vary greatly from one location to another, even though they are within a single cell and share the same base station/access point.

Given that the wireless channel may exhibit a wide range of error patterns, we adopt a coarse-grained and elastic channel error model to eliminate the challenging problem of real-time channel state prediction.

Multi-slot channel error model (n, δ)*:* Given n slots, the scheduler will not perceive channel error in more than δ slots.

The above model does not seek to characterize or predict the exact error pattern. Instead, it gives an overall estimation on the capability of channel error to disrupt transmission. Given that such a constraint is known to both the base station and channel error, the base station is always preparing for the worst case while it has no information on what the real channel error distribution will be. In this work, we seek to derive a robust scheduling policy for the base station to optimize its worst-case performance.

C. Zero-sum Game

We model the interactions between the scheduler[2] and the channel error as a two-player zero-sum game (We refer to [1] for an introduction to the basics of game theory). In this game, the two players, scheduler and channel error, are non-cooperative adversaries against each other. The scheduler seeks to maximize its total revenue by scheduling m flows, while the channel error (CE) always seeks to minimize the scheduler's total revenue. The two players act simultaneously, and neither of them knows the other player's action in advance. To this end, the CE adversary is empowered to exhibit any error pattern that conforms to the (n, δ) model.

We admit that in such a game theoretic approach, the CE player might be assumed too *intelligent* and the real channel error might not match this *intelligence*. However, the scheduling strategy derived from this assumption achieves worst-case optimality and it is especially useful for performance-critical applications, and in situations where it is hard to accurately predict the channel state.

III. Single-slot Channel Error Model and Robust Scheduling

Before introducing the multi-slot channel error model (n, δ), we first study a simple single-slot model in this section; this is also the model adopted in a recent work [4]. In this model, we consider every individual time slot in which the scheduler must choose one to transmission from m backlogged flows. We assume that channel error will corrupt one and only one flow among these m flows in each slot. Each flow has an advertised price representing the reward that the user is willing to pay for scheduling this flow per unit slot. The same assumptions have also be used in [5]. We denote the price of flow i as u_i. Without loss of generality, we always assume $u_1 \geq u_2 \geq ... \geq u_m$ in this paper.

We study this single-slot model due to two reasons. First, it models the scenarios in which channel error is heavy. Secondly, due to its simplicity, it can help us to gain some insights on the structure and characteristics of the worst-case otpimal scheduling policy.

A. 2-flow Case

We start with the simplified two-flow scenario and illustrate the nature of the zero-sum game between the scheduler and CE. There are only two flows with prices u_1 and u_2 respectively ($u_1 \geq u_2$). CE may corrupt either of them to minimize the scheduler's total revenue. On the other side, the scheduler seeks to defeat CE by choosing the other flow in order to maximize its revenue. This can be modeled by a matrix game, as shown in Table I. Each matrix entry denotes the value when the scheduler and CE take the corresponding decisions.

Let us consider the determinstic strategy for the scheduler and CE: the scheduler chooses to schedule either flow 1 or

[2]We use 'scheduler' and 'base station' interchangeably in this paper.

	Corrupt flow 1	Corrupt flow 2
Schedule flow 1	0	u_1
Schedule flow 2	u_2	0

TABLE I

GAME MATRIX IN TWO-FLOW SCENARIO

flow 2, while the CE chooses to corrupt either flow 1 or flow 2. Because the scheduler and CE make their decisions simultaneously, if they take such a determinstic strategy, there is no equilibrium point (or so-called saddle point in game-theoretic terminology [1]) in the sense whatever action each player takes, he will regret after the game is over.

We now enlarge the feasible scheduling policy space to a probabilistic setting, which will lead to the equilibrium property in the zero-sum game. In order to decrease the risk that its decision is guessed by the adversary, each player independently chooses both flows with certain probability, In game theoretic terminology, this is called the *probabilistic strategy*. The probabilistic strategy for the scheduler is defined as $\{$schedule flow 1 with probability q, schedule flow 2 with $1-q\}$, while the probabilistic strategy for the CE is $\{$corrupt flow 1 with probability p, corrupt flow 2 with $1-p\}$, where $p, q \in [0,1]$. The expectation of the total revenue is given by $f(q,p) = q(1-p)u_1 + (1-q)pu_2$.

In the above game, the scheduler is the maximizer and the CE is the minimizer. We can formulate it as a minimax problem $\max_q \min_p f(q,p)$. It is easy to solve this minimax problem and get the solution as follows: the optimal strategy for the CE is $p^* = \frac{u_1}{u_1+u_2}$; the optimal strategy for the scheduler is $q^* = \frac{u_2}{u_1+u_2}$ and the optimal total revenue is $f^* = \frac{u_1 u_2}{u_1+u_2}$. Note that this revenue is in a statistical sense and represents the expected revenue that the scheduler can achieve.

We can show that the aforementioned strategies p^* and q^* indeed reach an equilibrium point. At this equilibrium point, the scheduler's optimal strategy q^* can ensure the system revenue will be no less than a threshold $f(q^*)$, no matter what the real channel error distribution is. In this sense, we call it robust scheduling policy with optimal worst-case performance.

B. General Case

We now generalize to the case when there are m flows ($m > 2$) and each of them may have different exposure to the attack of the CE. We use λ_i to represent mobile users' exposure to channel errors. λ_i means that if CE decides to corrupt flow i, its success probability is $1-\lambda_i$ ($0 < \lambda_i < 1$). When λ_i increases, flow i becomes more resistant to channel errors (if $\lambda_i = 1$, flow i will be error-free). We assume these λ_i are known *a priori* to the scheduler and CE. In a similar way as the previous 2-flow case, we model the scheduling problem as a zero-sum game with the following game matrix:

$$A = \left| \begin{array}{ccccc} \lambda_1 u_1 & u_1 & u_1 & \cdots & u_1 \\ u_2 & \lambda_2 u_2 & u_2 & \cdots & u_2 \\ u_3 & u_3 & \lambda_3 u_3 & \cdots & u_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_m & u_m & u_m & \cdots & \lambda_m u_m \end{array} \right|_{m \times m} \quad (1)$$

We are again interested in probabilistic strategies. Now the scheduler's total revenue is given by

$$f = \sum_{i=1}^{m} [q_i(1-p_i)u_i + \lambda_i p_i q_i u_i]$$

where q_i and p_i are the probability with which the scheduler schedules and the CE corrupts flow i, respectively. Let f^* be the optimal worst-case revenue. Since this is a two-player zero-sum game, justified by Minimax theory [1], there should be

$$\begin{aligned} f^* &= \max_{\{q_i\}}(\min_{\{p_i\}} \sum_{i=1}^{m} [q_i(1-p_i)u_i + \lambda_i p_i q_i u_i]) \\ &= \min_{\{p_i\}}(\max_{\{q_i\}} \sum_{i=1}^{m} [q_i(1-p_i)u_i + \lambda_i p_i q_i u_i]) \quad (2) \end{aligned}$$

We have solved this minimax problem in [13]. For clarity of illustration, in Theorem 1, we give the solution when all the λ_i in matrix (1) are equal to zero. This special case does not change solution structure.

Theorem 1: Consider m flows with their prices in descending order ($u_1 \geq u_2 \geq \ldots \geq u_m$). Let k be the largest integer within the range $[2, m]$ such that $\frac{k-2}{u_k} \leq \sum_{i=1}^{k-1} \frac{1}{u_i}$. Then, the scheduler's optimal strategy q_j^* has the following form

$$q_j^* = \begin{cases} \frac{1/u_j}{\sum_{i=1}^{k} 1/u_i} & \text{for } j = 1, \ldots, k \\ 0 & \text{for } j = k+1, \ldots, m. \end{cases} \quad (3)$$

while for the scheduler, the worst case happens when the CE uses p_j to corrupt flow j where

$$p_j^* = \begin{cases} 1 - \frac{(k-1)/u_j}{\sum_{i=1}^{k} 1/u_i} & \text{for } j = 1, \ldots, k \\ 0 & \text{for } j = k+1, \ldots, m. \end{cases} \quad (4)$$

the optimal worst-case revenue is given as $f^* = \frac{k-1}{\sum_{i=1}^{k} 1/u_i}$.

Theorem 1 gives the expected optimal worst-case revenue for the scheduler. If this matrix game is independently repeated for a sufficiently large number of times, and the worst-case channel error distribution always happens in each slot, then the arithmetic mean of the outcome earned by the scheduler is equal to the expected optimal worst-case revenue in Theorem 1. Such a single-slot channel error model may be overly conservative, because the CE is assumed to be able to launch attacking in every slot while the scheduler always has to be prepared for the worst situation. We can see that such a single-slot error model is appropriate for modeling situations with high channel error ratio.

Theorem 1 also reveals several interesting points regarding the scheduler's optimal decision. (1) There always exists a *cut-off* price u_k. To achieve the optimal worst-case performance, the scheduler will not schedule flows with prices lower than u_k. Such a threshold-based scheduling policy reflects the collective agreement involving all the flows in order to maximize the total revenue while taking the least risk. (2) The scheduler tends to diversify its scheduling decisions among those flows with prices higher than u_k. This diversification is used to be illusive to the CE adversary and will hopefully decrease the risk of being corrupted by CE. (3) Among the flows that the scheduler chooses for scheduling, a flow with higher price has a comparatively lower probability to be scheduled. This counter-intuitive property results from the scheduler's inherent property to achieve both high performance and low risk.

We have provided in [13] the solution to the general case when $\lambda_i \neq 0$. From the solution, we can see that each λ_i acts as the weight of flow i's price. It does not change the *structure* of the scheduler's robust scheduling policy, and there still exists a threshold u_k playing the same role as in Theorem 1.

C. Location Dependent Channel Error

Our framework can be readily extended to accommodate location-dependent channel errors, which are common in wireless networks due to interference, fades and multipath effects. When location-dependent channel errors happen, a subset of backlogged flows which are destined to physically neighbouring mobile users usually experience the same error patterns. In this scenario, we treat these correlated flows as one group. Whenever any flow is corrupted by CE, all the other flows in the same group will also be corrupted. For a m-flow case, if we have G groups in total, we can show that it is equivalent to a G-flow scenario. Then we can still apply our previous approach.

We illustrate this by a simple example. Considering a $\{m = 3, G = 2\}$ scenario with flows 1 and 2 in a group and flow 3 in the other group, the matrix game is written as Table II.

	Corrupt flow 1	Corrupt flow 2	Corrupt flow 3
Schedule flow 1	0	0	u_1
Schedule flow 2	0	0	u_2
Schedule flow 3	u_3	u_3	0

TABLE II
GAME MATRIX IN $\{m = 3, G = 2\}$ SCENARIO WITH LOCATION-DEPENDENT ERRORS

From Table II, we can see that scheduling flow 1 is always preferable for the scheduler than scheduling flow 2 due to $u_1 > u_2$, while for CE, whether corrupting flow 1 or flow 2 brings in the same damage to the system revenue. Therefore, flow 2 can be eliminated from the consideration of both the scheduler and CE. This simplifies the problem to a $\{m = 2, G = 2\}$ scenario which is well solved by previous approach.

IV. MULTI-SLOT CHANNEL ERROR MODEL AND ROBUST SCHEDULING

In this section we consider a generalized multi-slot (n, δ) error model to catch a wide range of mild channel error patterns. We first divide the time into time windows. Each time window consists of n time slots. The (n, δ) error model means the scheduler perceives channel errors in no more than δ slots in each time window. When $n \gg \delta$, this model provides an elastic constraint on the channel error occurrence, which can accommodate a wide range of error patterns. Thus it is appropriate for modeling scenarios with mild channel error ratio. The goal of this section is to determine the robust scheduling policy under this generalized error model. Again, we formulate the scheduling problem into a matrix game, in which the scheduler seeks to maximize the total revenue over the remaining slots until the end of the time window, while CE seeks to minimize this total revenue.

We still assume m backlogged flows and their prices are sorted by $u_1 \geq u_2 \geq \ldots \geq u_m$. The flow prices are assumed constant during the total n slots. We also assume that the CE can attack at most one flow in each slot.

We start from the first slot of the n-slot time window and denote the state of a slot as (r, ϵ), where r is the amount of remaining slots within the current time window, and ϵ is the maximum number of slots for the CE to launch attacking during the remaining r slots. For the first slot of the time window, there are $r = n$ and $\epsilon = \delta$.

Similar to previous single-slot matrix game, the scheduler's strategy is in probabilistic form and denoted as $\{q_1, \ldots, q_m\}$ where q_i represents the probability of scheduling flow i. However, the objective function that the scheduler seeks to maximize is no longer the revenue in the current slot, but the sum of the expected revenue over all the remaining r slots. We denote this expected aggregate revenue as $f(r, \epsilon)$. On the other side, the CE adversary seeks to minimize $f(r, \epsilon)$. Note that CE has one more choice than the scheduler, i.e., it can either choose to corrupt any of the m flows or simply delay the corruption and keep this attacking credit for future use. The latter choice may indeed happen if a later corruption can bring more damage to the system revenue. We denote the CE's strategy as $\{p_1, \ldots, p_{m+1}\}$, where $p_i (1 \leq i \leq m)$ is the probability that the CE chooses flow i to corrupt, and p_{m+1} is the probability that CE defers its corruption in the current slot.

After the current slot expires and both the scheduler and CE have taken their actions, r is decreased by 1. If CE has decided to corrupt any flow, ϵ is decreased by 1 ; otherwise, if

CE has decided to defer the corruption, ϵ remains unchanged. In this way, the game moves forward to another state and it will continue until $r = 0$ (after $r = 0$, a new time window will start). Such a procedure is well modeled by a dynamic zero-sum game. Solving this game will ensure a bounded worst-case system revenue.

Considering the state (r, ϵ), we easily know that its game matrix is equal to $f(r-1, \epsilon-1)$ plus

$$\begin{vmatrix} 0 & u_1 & u_1 & \cdots & u_1 & u_1-\theta \\ u_2 & 0 & u_2 & \cdots & u_2 & u_2-\theta \\ u_3 & u_3 & 0 & \cdots & u_3 & u_3-\theta \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ u_m & u_m & u_m & \cdots & 0 & u_m-\theta \end{vmatrix}_{m\times(m+1)} \quad (5)$$

where $\theta = f(r-1, \epsilon-1) - f(r-1, \epsilon)$. θ can also be considered as a fixed charge that the CE imposes on the scheduler for deferring the attack in the current slot. By using Lemma 2.2 in [1], we can verify that the optimal strategies $\{q_i\}$, $\{p_j\}$ do not change when we extract $f(r-1, \epsilon-1)$ from the game matrix.

Based on Theorem 1, we give the solution to the game matrix (5) in the following theorem (Proof is provided in [13])

Theorem 2: 1) If $\theta \le (\sum_1^k 1/u_i)^{-1}$ (k is defined in Theorem 1), the scheduler's optimal strategy $\{q_i\}$ and the worst-case revenue are given in Theorem 1. Accordingly, the optimal worst-case revenue is $f^* = \frac{k-1}{\sum_{i=1}^k 1/u_i}$.

2) If $\theta \ge u_1$, the scheduler will use greedy algorithm in the sense that it always schedules flow 1 which provides the maximum price. The worst case happens when there is no channel error in the current slot. Accordingly, the optimal worst-case revenue is $u_1 - \theta$.

3) Otherwise if $(\sum_{i=1}^k 1/u_i)^{-1} < \theta < u_1$, we first find d ($2 \le d \le k$) satisfying

$$(\sum_{i=1}^{d} 1/u_i)^{-1} < \theta \le (\sum_{i=1}^{d-1} 1/u_i)^{-1} \quad (6)$$

Then the scheduler's optimal strategy is

$$q_j = \begin{cases} \theta/u_j, & \text{for } j = 1, \ldots, d-1; \\ 1 - \theta \sum_{i=1}^{d-1} 1/u_i, & \text{for } j = d; \\ 0, & \text{for } j > d. \end{cases} \quad (7)$$

while the worst case happens when

$$p_j = \begin{cases} 1 - u_d/u_j, & \text{for } j = 1, \ldots, d-1; \\ 0, & \text{for } j = d, \ldots, m; \\ u_d \sum_{i=1}^{d-1} 1/u_i - (d-2), & \text{for } j = m+1. \end{cases} \quad (8)$$

Accordingly, the optimal worst-case revenue is $f^* = (d-2)\theta + u_d - \theta u_d \sum_{i=1}^{d-1} 1/u_i$,

As we can see from Theorem 2, during each slot, the robust scheduling policy considers not only the prices of all the backlogged flows, but also the potential penalty $\theta = f(r-1, \epsilon-1) - f(r-1, \epsilon)$, which measures the revenue loss if channel errors happen in the next slot. The scheduler then have three choices: (1) if the potential penalty is too large compared to the prices of backlogged flows, namely, $\theta \ge u_1$, the scheduler will make a greedy choice and only schedule the flow with the maximum price. In this case, the worst case happens when there is no channel errors in the current slot. This seems counter-intuitive, however, the underlying reason is that it is more profitable for the CE to defer its attacking to the remaining slots than to attach in this slot. (2) If the potential penalty θ is comparatively small, i.e.,$\theta \le (\sum_1^k 1/u_i)^{-1}$, the scheduler tends to believe the channel error is an inevitable event in this slot, thus it will adopt the conservative scheduling policy, which we have presented in the single-slot case. (3) If the potential penalty is between the previous two extreme cases, the scheduling policy becomes exactly a trade-off between the greedy and conservative scheduling policy.

In a summary, in order to achieve optimal performance in the worst case, the scheduler will always evaluate the potential penalty caused by channel error. If this potential penalty is mild, the scheduler is more likely to take conservative policy. Otherwise, the scheduler is more likely to use greedy policy.

V. Simulation Evaluation

We use simulations to compare the proposed robust algorithm with the greedy algorithm and the adaptive scheduler proposed in [7]. For the robust algorithm, we only consider the multi-slot error model case, as described in Section IV. The greedy scheduler is defined as always scheduling the flow with the maximum price, regardless of the channel state of the flow. The adaptive scheduler, on the other hand, only schedules the flow with the maximum price among all the flows that perceive a clean channel. Details of our simulation settings are referred to [13].

We first evaluate the three algorithms by employing a two-state Markov model to generate channel error patterns. The two-state Markov error model is configured as follows. For each flow k, p_e^k is the probability of perceiving an error given that the current channel state is *clean*, and p_g^k is the probability of perceiving a clean channel given that the current state is in *dirty*. In Table III we give the three schedulers' performance. It shows that, among 1000 simulation runs, the lowest performance achieved by the robust scheduler is 51.27, 40% higher than that of greedy and adaptive schedulers. This better *worst-case* performance is also reflected from the smaller performance variance of the robust scheduler. However, the average performance of

	Lowest	Highest	Average	Variance
Robust scheduling	51.27	61.39	58.94	1.13
Greedy scheduling	36.69	65.95	60.43	2.06
Adaptive scheduling	36.54	67.97	62.59	2.42

TABLE III

PERFORMANCE COMPARISION AMONG ROBUST, GREEDY AND ADAPTIVE SCHEDULERS

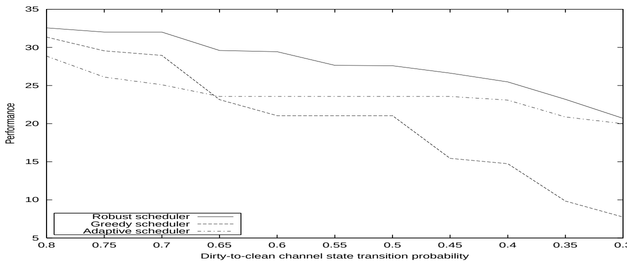

Fig. 2. Worst-case performance for robust, greedy and adaptive algorithms

the robust scheduler is 61.39, slightly worse than the greedy and adaptive schedulers.

We then keep p_e^k (probability of channel state transition from clean to dirty) as 0.8 and decrease p_g^k (probability of channel state transition from dirty to clean) from 0.8 to 0.3. Such a decrease indicates a more bursty channel error distribution. As we can observe from Figure 2, the worst-case performance for all these three schedulers decrease as the channel error distribution becomes more bursty. However, the worst-case performance of the robust scheduler decreases much more gracefully. We also notice that there exists a critical p_g^k. Above this critical p_g^k, the adaptive scheduler attains a consistently better worst-case performance than the greedy scheduler. This phenomenon is due to the failure of the one-step prediction in bursty channel error case.

VI. CONCLUSIONS

The main design issue for wireless scheduling is to provide service assurances in the presence of dynamic channel error conditions. Current wireless scheduling algorithms [4]-[10] typically take an adaptive approach in which the scheduler adapts its scheduling decision based on the estimated fine-grained channel state. This paper explores an alternative approach to handle channel errors — game-theoretic, robust packet scheduling in wireless cellular networks. In this approach, the scheduler and the channel error play a zero-sum game and act as adversaries in the scheduling process. By solving the corresponding minimax optimization problem, we characterize the worst-case optimal scheduling policies. The merits of this approach include requirements of only coarse-grained channel state, worst-case optimal performance, guaranteed service in the presence of a wide range of error patterns, and balanced trade-off between conservative and greedy policies. Though this work is mainly of theoretic merit, it opens door to further exploring probabilistic scheduling approach to handle channel errors. Ongoing work seeks to further compare the robust and adaptive approaches.

VII. ACKNOWLEDGMENTS

Prof. Thomas S. Ferguson make important contribution to Theorem 2.

VIII. REFERENCES

[1] T. Basar and G.J. Olsder, Dynamic noncooperative game theory, *Academic Press*, 1998.

[2] P.R.Thie, An introduction to linear programming and game theory, *John Wiley & SONS Inc.*, 2rd Edition, 1988.

[3] H. Zhang, Service disciplines for guaranteed performance service in packet-switching networks, in *Proceedings of the IEEE*, 83(10), pp. 1374-1396, October 1995.

[4] X. Liu, E.K.P. Chong and N.B.Shroff, Transmission scheduling for efficient wireless utilization, *IEEE INFOCOM'01*, May 2001.

[5] M.Agarwal and A.Puri, Base station scheduling of requests with fixed deadlines, *IEEE INFOCOM'02*, June 2002.

[6] F.P.Kelly, A.Maulloo and D.Tan, Rate control for communication networks: shadow prices, proportional fairness and stability, *Journal of the Operational Research Society 49*, pp.237-252, 1998.

[7] S. Lu, V. Bharghavan and R. Srikant, Fair scheduling in wireless packet networks, *IEEE Trans. on Networking*, August 1999.

[8] T.S. Ng, I. Stoica and H. Zhang, Packet fair queueing algorithms for wireless networks with location-dependent errors, *IEEE INFOCOM'98*, March 1998.

[9] P. Ramanathan and P. Agrawal, Adapting Packet Fair Queueing Algorithms to Wireless Networks, *ACM MOBICOM'98*, October 1998.

[10] S. Lu, T. Nandagopal, and V. Bharghavan, Fair scheduling in wireless packet networks, *ACM MOBICOM'98*, October 1998.

[11] G. T. Nguyen, R. H. Katz, B. Noble, and M. Satyanarayan, A trace-based approach for modeling wireless channel behavior, in *Proceedings of Winter Simulation Conference*, pp. 597-604, December 1996.

[12] D. Eckhardt and P. Steenkiste, A trace-based evaluation of adaptive error correction for a wireless local area network, to appear in *Mobile Networks and Applications (MONET)*, Special Issue on Adaptive Mobile Networking and Computing.

[13] X.Meng, Z.Fu, H.Yang and S.Lu, Robust packet scheduling in wireless cellular networks, *Technical Report*, CS department of UCLA, http:\\www.cs.ucla.edu\~xqmeng\paper\cdc_tr.ps.