INTELLIGENT PACKET SCHEDULER FOR GENERAL PACKET RADIO SERVICE

H. Bhaskar[1], R. Everson[1], M. Witwit[2]*, J. Gil[2]

[1]Department of Computer Science, University of Exeter, UK
[2]Motorola, Swindon, UK

Keywords: GPRS, quality of service, reinforcement learning, scheduling.

Abstract

The General Packet Radio Service (GPRS) augments GSM to provide packet switched data services to the mobile users. Packet scheduling in GPRS is dynamic and several scheduling techniques have been implemented, for example round robin; however, these generally assure only best effort quality of service. In this paper we compare prominent scheduling algorithms by simulation of web and email traffic finding that Earliest Deadline First and First Come First Served scheduling perform well with few users, but round robin is preferable with large numbers of users. We introduce a novel scheduling algorithm, based on reinforcement learning, for scheduling packets according to quality of service. Simulation studies show that it outperforms a naïve prioritised round robin algorithm and can adapt to changing network conditions.

1 Introduction

Mobile communication systems have been revolutionized by technological advances in the last decade. The General Packet Radio Service [2, 1, 5] has augmented GSM-enabled [see, for example, 3] mobile stations to provide packet switched data services to the mobile user. The various data services (e.g., internet browsing, email, fax, unified messaging, etc.) and voice require different levels of service: a few seconds delay delivering an email is unimportant, but makes web browsing annoying at best and conversation almost unsustainable. Despite this clear need for prioritised scheduling, popular scheduling algorithms, for example round robin, assure only best effort quality of service. Though best effort service allows unbiased servicing to users, it fails to provide value-based scheduling.

In this paper we describe a simple reinforcement learning scheduler which provides appropriate levels of service to according to the traffic priority. We first compare the performance of well-known scheduling algorithms in a simulation of internet traffic. We first describe a simple, prioritised round robin algorithm. A reinforcement learning algorithm is then used to improve the prioritised round robin algorithm and its performance is illustrated, again with simulations of internet traffic; we show that it affords better throughput for high priority traffic and is able to adapt to changing network conditions.

*Corresponding author; email: `Mehdi.Witwit@motorola.com`

1.1 GSM and GPRS

GSM, originally designed for voice traffic, is based on for circuit switched connections. This is achieved by a combination of Frequency Division Multiple Access (FDMA) and Time Division Multiple Access (TDMA). Each Base Transceiver Station (BTS) serving mobile stations in a cell is assigned a number of single carrier channels for uplink and downlink. Each of these frequency channels is then divided into eight time slots comprising a TDMA frame. A mobile station is allocated a particular time slot in successive TDMA frames forming physical channel, and a mobile station is assigned a single Traffic Channel (TCH) for the duration of the call.

Channels allocated to GPRS are called Packet Data Channels (PDCH), and in contrast to voice traffic, a GPRS permits a mobile station to utilise several time slots in the same TDMA frame. Time slot allocation is dynamic [2] and a single time slot may be used by more than one mobile station in different TDMA frames. Dynamic channel allocation is thus very flexible, permitting a single mobile to use upto eight time slots, while at the other extreme upto eight mobiles may use a single time slot. This is especially useful when the traffic is bursty, as is typical with data. Moreover, there is the potential for high priority services to be assigned more TDMA slots in a single TDMA frame or more frequent slots in successive TDMA frames, thus providing a higher quality of service (QoS).

The Packet Control Unit (PCU) acts as the interface between the GPRS packet switched network and the GSM subsystem. Focusing on the uplink from mobile station to Base Station Controller (BSC), traffic arriving from the mobile stations is directed by the PCU into the packet switched network. Since there is generally a large user population data packets do not arrive serially, but in parallel and the PCU must schedule these packets in such a way that users are guaranteed quality of service and fairness.

Four QoS criteria are generally considered important for GPRS networks (see [4] for details): precedence (priority), delay, reliability and throughput. In this study we focus on delay and throughput. Delay is measured as the ratio of the number of packets that have been delayed to the total number of packets received. The throughput is defined as the ratio of the total size of packets that have been serviced to the time taken for the packets to be serviced. It is also useful to define the cell load as the ratio of the total number of packets serviced to the total number of packets arriving in a cell.

Three classes of QoS may be recognised:

Best effort: This level of QoS merely guarantees that resources are shared fairly among the data flows. This is the QoS provided by the round robin algorithm used by current schedulers in the PCU.

Absolute: The absolute QoS guarantee provides a QoS to a flow irrespective of the QoS of other flows. This offers isolation to network services. The QoS may be specified either deterministically or statistically, however, the model requires a resource reservation mechanism, which is not currently available in the GPRS protocol.

Adaptive: The adaptive QoS guarantee level of service permits the QoS provided to any flow to adapt to the load conditions of the network, potentially with reference to upper and lower bounds. In contrast to the absolute guarantee, this mechanism requires adaptive feedback mechanism and offers fairness to the network. Our reinforcement learning scheduler provides an adaptive QoS.

2 Simulation

In order to study the efficacy of scheduling algorithms we have simulated the downlink traffic within a cell. In addition to the scheduling discipline, the simulator requires a traffic model and a channel model, which we describe in turn.

Traffic Model. Traffic may be broadly divided into four classes, namely: Conversational, Streaming, Interactive, Background. We have chosen to model web traffic as an exemplar of the interactive class, requiring a high QoS, and email traffic for the background class, requiring a low QoS.

A cell typically involves several users (between 30 and 210 in our simulations) connecting through GPRS. Following [7], we group all requests at a particular instant into the appropriate QoS category and add the totals to the to the PCU queue.

Furthermore [7], we model HTTP traffic as a two state Markov process. An ON phase is initiated by the HTTP request for a URL; the OFF phase represents the quiescent period following the retrieval of all HTTP objects. Traffic during the ON phase is determined by the number of inline objects retrieved (distributed according to a Gamma distribution), the sizes of the main object and inline objects, which are modelled as being log-normally distributed, and the viewing time of each page, which is assumed to be Weibull-distributed. All parameters are empirically determined [7].

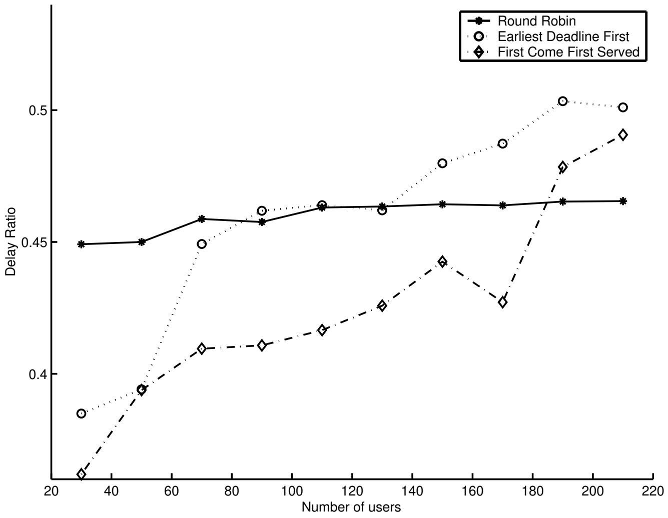

Figure 1: Probability that a packet is delayed when scheduled by Round Robin, Earliest Deadline First or First Come First Served algorithms.

The distribution of email sizes is approximated by a Cauchy distribution [7], while the arrival times are modelled by a Poisson process.

Channel Model. In the work reported here we have modelled each mobile station as being capable of using upto four of the eight available time slots in each TMDA frame. We further assume that four time slots in each frame are reserved for GPRS data traffic, while the other 4 time slots maybe utilised if they are available. We model the data rate over each channel as 21.4 kb/s. The number of available channels is modelled as being equal to the number of mobile stations.

2.1 Standard scheduling algorithms

We compared the efficacy of the following three well known scheduling algorithms:

First Come First Served (FCFS): Incoming packets are scheduled according to their arrival time at the scheduler. Clearly, high volumes sources are given effectively higher priority than low volume sources.

Round Robin (RR): Here incoming packets are placed in queues according to their source; packets are scheduled from queues one at a time for each queue in turn, round robin fashion. This scheduler is fair in the sense that it gives equal chance of service among the data flows.

Earliest Deadline First (EDF): At any scheduling decision time, the packet with the earliest deadline is scheduled. Deadline is calculated as the upper bound of the tolerable end-to-end delay, and thus measures the usefulness of data packets at the destination. Ties are broken randomly.

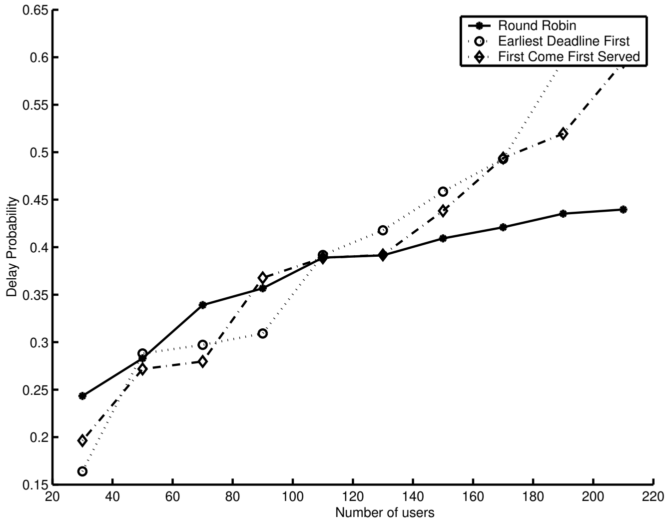

Figure 2: Probability that a packet is delayed when scheduled by RR, EDF and FCFS algorithms. Traffic per user is half that simulated in Figure 1.

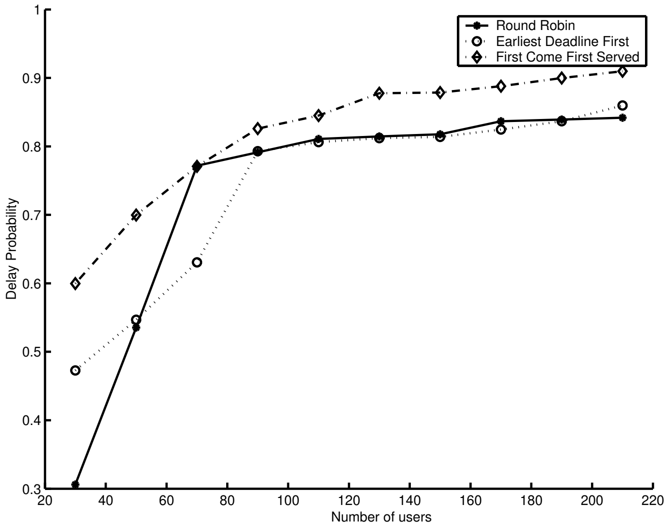

Figure 3: Probability that a packet is delayed when scheduled by RR, EDF and FCFS algorithms. Traffic per user is double that simulated in Figure 1.

Figure 1 compares the performance of the scheduling algorithms as the number of users is increased. Note that the number of available TMDA slots is 4 times the number of users; the traffic per user is 500 packets for email and 500 HTTP packets. Since the traffic is 'bursty' there is an effective increase in overall network load as the number of users rises because there are more collisions between bursts generated by different users.

As the figure shows, the probability of a packet being delayed is almost constant with increasing network load. The FCFS algorithm performs best except when the network is most heavily loaded. FCFS might be expected to be effective when the network is not saturated because it can always schedule immediately schedule the first available packet.

At lower traffic per user (Figure 2) the algorithms all perform similarly with an increasing probability of delayed packets as the number of users, and thus collision of bursts, increases. At high traffic levels (Figure 3) the algorithms all saturate with more than approximately 80 users.

3 Reinforcement Learning

Here we give a very brief introduction to reinforcement learning (RL); readers are referred to Sutton & Barto's expository book [6] for an extensive discussion. Unlike a supervised learning agent, a RL agent is not told the correct action to take in a particular situation when it is learning. Instead, the agent is given a goal and learns by experience how best to achieve it. More precisely, an RL agent at time step t is characterised by a state, $s_t \in \mathcal{S}$. The state signal the agent receives represents the knowledge possessed by the agent about its environment. The particular action, $a_t \in \mathcal{A}(s_t)$ taken depends upon the *policy*, $\pi(s_t, a_t)$, the probability of taking action a_t in state s_t. The policy constitutes a mapping from the state to the action and learning the optimal policy is the essence of reinforcement learning. While choosing actions $a_t = \arg\max_a \pi(s_t, a)$ for all t will yield a sequence of actions that optimally exploit the current estimates of a 'good' action, exploration is ensured by requiring that all actions $\mathcal{A}(s)$ from state s have a non-zero probability $\pi(s, a)$.

The agent's goal is to maximise the *expected return*:

$$R_t = \sum_{\tau=0}^{\infty} \gamma^\tau r_{t+\tau+1} \tag{1}$$

where r_{t+1} is the reward obtained in response to the action a_t and γ is the rate at which future rewards are discounted, $0 \leq \gamma \leq 1$. Of course, learning often stops after a finite number of steps, T.

It is generally assumed that the environment of the agent has the Markov property, so that the state and reward at time $t + 1$ depend only upon the state and action at time t:

The merit of taking an action a from a state s under a policy π may be assessed by the *action-value function*, which is the expected return under π:

$$\begin{aligned} Q(s,a) &= E_\pi \left\{ R_t \,|\, s_t = s, a_t = a \right\} && (2) \\ &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \,|\, s_t = s, a_t = a \right\} && (3) \end{aligned}$$

The action-value utility may be estimated by keeping track of the rewards accrued starting from (s, a). Here we use *Q-learning* [8], a temporal differencing method for updating the action-value function. Having taken action a_t from state s_t to state s_{t+1} and received reward r_{t+1}, the estimate of the utility

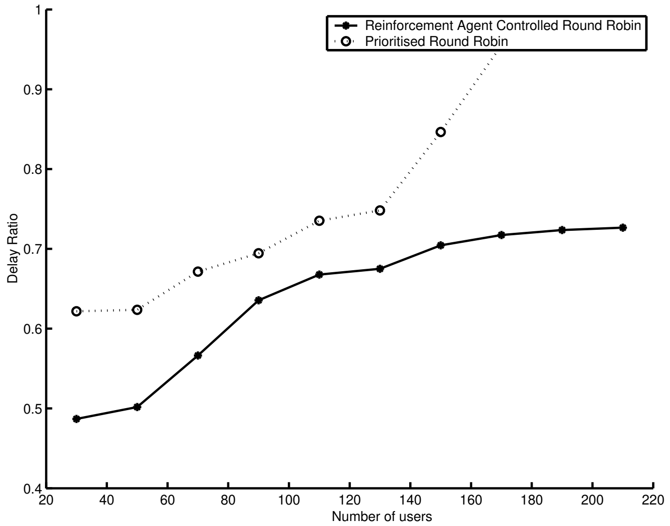

Figure 4: Probability that a packet is delayed when scheduled by 'prioritised round robin' and RL schedulers.

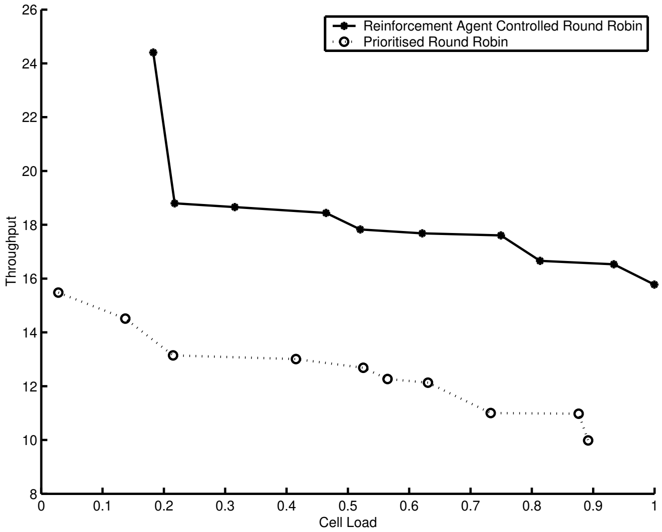

Figure 5: Overall throughput per user versus cell load for prioritised round robin and RL schedulers.

$Q_{t+1}(s,a)$ at time $t+1$ is updated as follows:

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha[r_{t+1} + \gamma \max_a Q_t(s_{t+1},a) - Q(s_t,a_t)] \quad (4)$$

where α is a learning rate. The the update looks one step ahead and updates $Q(s_t,a_t)$ with the reward accrued from taking action a_t and the discounted maximum (estimated) return from an action at time $t+1$.

3.1 RL Scheduler

Our RL scheduler is designed to give priority to flows requiring high QoS at the expense of low QoS traffic; for simplicity in this feasibility study we focus on just two QoS categories: high and low. It builds on a naïve prioritised scheduler which consists of a pair round robin scheduling queues, one for each QoS category into which packets are placed according to their QoS requirement. In the naïve scheduler priority is given to the high QoS queue by scheduling two packets from it for every packet scheduled from the low QoS queue.

While this 'prioritised round robin' (PRR) scheduler clearly gives priority to high QoS flows, it is not adaptive and may waste network capacity. We therefore retain the high and low priority RR pools, but use an RL agent to choose from which pool to schedule at each instant.

Our RL scheduler thus has a very simple set of actions, $\mathcal{A}$: schedule a packet from either the high or the low QoS pool. The actions are chosen using an ϵ-greedy policy, so that the action with the highest utility $Q(s_t,a_t)$ in the current state is scheduled with probability $1-\epsilon$, the other action being chosen with probability $\epsilon(=0.1)$.

In this pilot study there are only two states available to the RL agent, namely is the packet that has been waiting longest to be scheduled a high or low QoS packet.

The reward is the instantaneous throughput (for the current TMDA frame) with high QoS traffic weighted by a factor λ; if r_t^h and r_t^l are respectively the number of high and low QoS packets scheduled at time t, the total reward is calculated as:

$$r_t = r_t^l + \lambda r_t^h \quad (5)$$

In the experiments reported here $\lambda = 2$. Clearly it would be straight-forward to generalise the reward function to include several QoS categories, each with an appropriate weighting.

3.2 Results

Figure 4 compares the probability of a packet suffering a delay when scheduled with the prioritised round robin or the RL scheduler. It is clear that the RL scheduler always schedules packets with less average delay than naïve algorithm, especially as the number of users, and thus the cell load, increases. As shown in Figure 5 the throughput (per user) is commensurately higher with the RL scheduler.

Figure 6 shows the probability of packet delay for high QoS and low QoS data flows. It is clear that the RL scheduler is able to provide a superior service to the high QoS flow under a wide range of cell loads; the probability of delay for high QoS users is consistently under one half that of the low QoS users. Increasing λ in the reward (equation 5) gives greater relative importance to the high QoS traffic, which are scheduled with probability of delay. It should be emphasised, however, that the RL scheduler is *adaptive*; it cannot offer absolute guarantees of throughput or delay probability to the high QoS data flows. As Figure 6 illustrates it does adapt to different network conditions to provide a proportionately better service to high priority users.

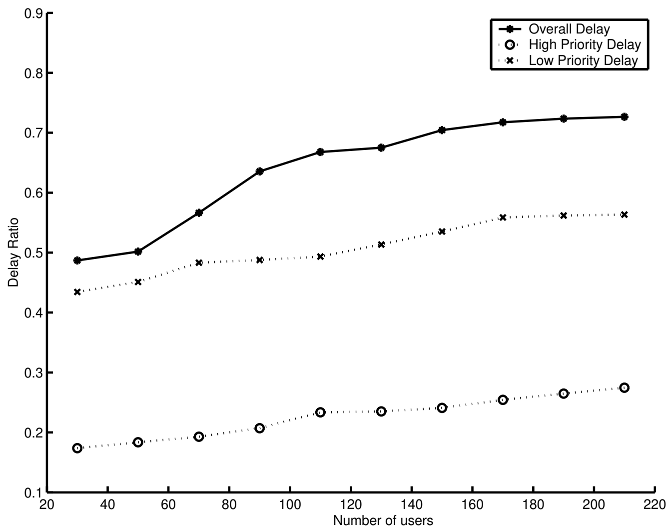

Figure 6: Delay probability versus cell load for high and low QoS flows using RL schedulers.

Figure 6 illustrates that the RL scheduler can automatically set an operating point for a range of network conditions. In figure 7, we illustrate how the RL scheduler can dynamically adapt to changing network load. The figure shows the transition from a high traffic situation at times before $t = 0.75 \times 10^5$, to a markedly lower overall throughput after $t = 0.75 \times 10^5$. As can be seen from the lower panel, the RL scheduler is able to adapt to the lower overall load and consistently schedule high QoS packets with a lower probability of delay. The scheduler is also able to cope with changes in the ratio of high to low QoS traffic.

4 Conclusion

We have presented an assessment of popular, simple scheduling algorithms for GPRS traffic. We find that although the FCFS and EDF schedulers perform well under some conditions as simple round robin scheme is robust over a wide range of cell loads.

We have also presented a simple reinforcement learning scheduler to differentially schedule data flows with differing QoS requirements. Simulations show that the RL scheduler is capable of preferentially scheduling high priority traffic over a wide range of network loads, and is capable of dynamically adapting to changing network loads and conditions. In this pilot study we have only examined two types of traffic (HTTP and email) together with two QoS classes, but it is straight-forward to extend the RL scheduler to cope with additional QoS classes. A more exciting development will be to extend the state of the RL agent to include additional information such as the number of packets in each of the RR pools or origin of data packets. Finally we remark that although we have concentrated on throughput as the reward, QoS criteria often include several ob-

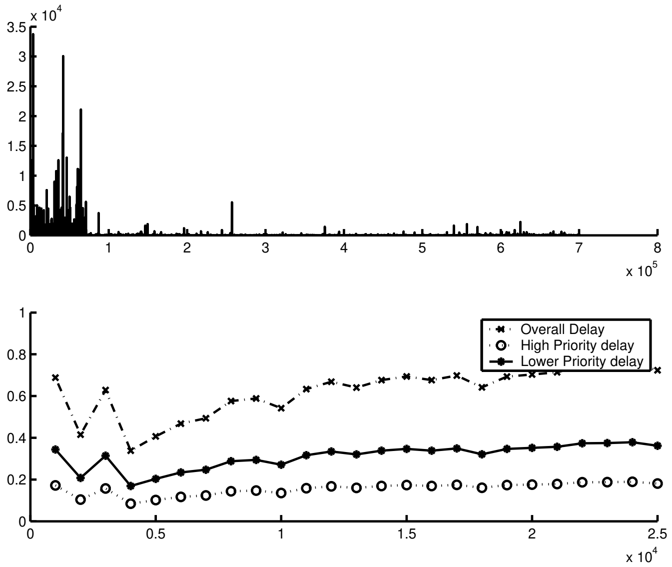

Figure 7: Adaption to changing network conditions. Top panel: packet size versus arrival time as the traffic density drops abruptly at $t = 0.75 \times 10^5$. Bottom panel: Delay probability versus time.

jectives and the development of multi-objective reinforcement learning schedulers will be important to provide QoS guarantees for QPRS and 3G technologies.

References

[1] J. Cai and D. Goodman. General packet radio service in gsm. *IEEE Communications Magazine*, 35:122–131, 1997.

[2] Cisco. *Overview of GSM, GPRS and UMTS*, 2001. Available from `www.cisco.com`.

[3] S.M. Redl, M.W. Oliphant, and M.K. Weber. *Introduction to GSM*. Artech House, 1996.

[4] A. Rizvi. *General Packet Radio Service (GPRS) Network Optimization Measurement Challenges Using Drive Testing*. Agilent Technologies, Ltd., February 2003. Available from `onenetworks.comms.agilent.com/bv/whitepapers.asp`.

[5] E. Seurre, P-J. Pietri, and P. Savelli. *GPRS for Mobile Internet*. Artech House, 2002.

[6] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[7] P. Tran-Gia, D. Staehle, and K. Leibnitz. Source traffic modeling of wireless applications. *Int. Electron. Commun. (AE)*, 55(1):27–36, 2001.

[8] C.J.C.H. Watkins. *Learning from Delayed Rewards*. PhD thesis, University of Cambridge, England, 1989.