On Flow Control and Scheduling in Time-Shared Wireless Packet Data Channels

Patrick Hosein and Rath Vannithamby
Ericsson Inc.
5012 Wateridge Vista Dr., San Diego, CA 92121, US
Email: {patrick.hosein, rath.vannithamby}@ericsson.com

Abstract—In high-speed wireless packet data networks such as 1xEV-DV, 1xEV-DO and HSDPA, the mobiles periodically (on the order of 1-2 ms) report channel quality information to the Base Station which then uses this information to schedule mobiles on the forward link over a slotted time-shared channel. The rate at which a mobile is served varies widely (by orders of magnitude) from slot to slot. Since each time a mobile switches to a new cell all data queued at the old cell must be forwarded on to the new cell for transmission it is desirable to keep only a limited amount of queued data at the cell. Therefore, due to the fluctuating servicing of data, the fluctuating arrival of data and the desire to maintain a small queue at the Base Station, flow control between the Base Station Controller and the Base Station is necessary. In this paper we consider this flow control problem and its interaction with the scheduling mechanism at the Base Station. We show that, for best effort traffic, instead of controlling the flow for each individual user one can maintain good performance by controlling the flow to a shared buffer. This results in reduced signaling overhead. We also show that for QoS traffic, minimal flow control is necessary.

I. INTRODUCTION

High-speed wireless packet data support on the forward link is included in all third generation (3G) standards such as IS-2000 (also called 1xEV-DV [1]), IS-856 (also called 1xEV-DO [2]) and HSDPA [3]. This support was included because of the increase in data usage and the increasing demands of new data applications being introduced by operators. However, increasing capacity alone is insufficient. Traditional wireless networks supported circuit switched services over fixed rate (power controlled) channels. Since the arrival rate and service rate were constant, buffer management was not necessary. In 3G forward link packet data channels, the arrival rate of packets (e.g. for best effort services) as well as the service rate provided to each user both fluctuate widely and hence proper buffer management is critical for optimal performance.

Best effort traffic such as web browsing tends to be bursty by nature. In addition, TCP flow control works by increasing the packet rate until packets are lost which is taken as a sign of congestion. TCP flow control then backs-off and the process is repeated providing the typical saw-tooth throughput. Therefore the packet arrival rate for a mobile user tends to be bursty. In addition, the forward packet data channel of 3G networks is time-shared and users are typically served when in good radio conditions. Therefore the intra-service time of a user is stochastic. Furthermore, when a user is served the rate at which it is served varies by several orders of magnitude. The throughput of a user can therefore be highly variable. Note however, that appropriate scheduling of users can reduce this variation and we will see that such schedulers are required for QoS guaranteed services.

In a typical 3G network (e.g. see Figure 1 for the case of a CDMA2000 based network), data enters the CDMA Radio Access Network (C-RAN) at a Packet Data Switching Node then to the Base Station Controller (BSC), then to the Base Station (BS) and finally to the mobile over the radio link. If no flow control is performed on any of the connecting (wired) links then data will be queued at the BS for transmission over the radio link since this typically forms the bottleneck due to the fact that it is a limited resource (the capacity of the wired links can always be increased but the radio link is limited by the spectrum assigned).

During the lifetime of a mobile, multiple cells may be involved in providing its service. Whenever the mobile changes its forward link serving cell to a neighboring cell all data that was queued at the source cell must be re-transmitted (by the BSC) to the destination cell. This requires bandwidth on the inter-BSC link and the process introduces a pause in service to the mobile. In order to reduce the bandwidth overhead and the delay introduced by cell switching, the amount of data queued at the BS should therefore be kept small. Controlling flow between the BSC and BS to maintain small queues at the BS and instead buffering the data at the BSC can accomplish...
this goal. However, it is also important to ensure that as long as there is data at the BSC, the corresponding queue at the BS should be non-zero. The reason for avoiding underflow is due to the nature of the scheduling mechanism. Users are typically scheduled when in good radio conditions and hence the probability of finding one or more such users at a scheduling decision point increases with the number of users with data for transmission (this is sometimes referred to as user diversity gains). Furthermore, the rate at which a mobile can be served depends on both radio conditions as well as the amount of data in its buffer. Therefore the queue at the BS should always be kept large enough (not only non-zero) to ensure that the service rate of a user is radio limited and not data limited.

Since each mobile may simultaneously support multiple application flows then each of these flows will need to be independently controlled. The rapidly changing service rate of a flow (caused by fast fading of radio links, variation in the number of flows being supported over the link, the nature and demands of these flows etc) together with the possibly rapid changes in the offered load (e.g. in the case of best effort traffic) means that the flow between the BSC and BS must be very tightly controlled and hence will require significant signaling overhead in the BS to BSC link. Even with such a stringent control there will be times when performance degrades. In present networks this flow control is performed on a per flow basis (i.e. for each user’s application). In this paper we will illustrate that this is not necessary. First we show that for those flows requiring QoS guarantees, a prudent choice of buffer size for the application flow is sufficient and no (or minimal) flow control is required for such services. Secondly, we show that for non-QoS flows (i.e., best effort traffic), a single shared buffer can be used at the BS and only this shared buffer needs to be flow controlled. We present a simple flow control mechanism for this shared buffer. Using these two approaches results in a simple, non-stringent control of a single flow between BSC and BS. This results in reduced signaling needs reduced implementation complexity, and increased fault tolerance while maintaining comparable performance. Finally, as we will later see, it allows for a simple congestion control mechanism.

Note that our approach follows the philosophy used on the Internet for the treatment of best effort traffic. At routers, IP flows are not independently queued and managed but all share a buffer and are collectively managed with mechanisms such as Random Early Discard (RED). We believe that as long as such traffic is treated in a best effort manner before its arrival to the C-RAN then it should continue to be treated in a like manner within the C-RAN.

In Section II we describe the basics of the scheduling mechanisms that are typically used in 3G forward packet data channels. Next we describe a generic flow control scheme for the per-application scenario as well as our proposed shared buffer flow control scheme and finally discuss the performance trade-offs.

II. THE BS SCHEDULER

All 3G wireless forward link packet data channels use similar techniques for improving data transport efficiency. These include, fast rate adaptation, fast time slot user scheduling and Hybrid ARQ (H-ARQ). Each mobile periodically reports its forward link radio conditions to the BS. The BS determines the most appropriate modulation and coding scheme for the given radio conditions and uses these to serve the mobile if chosen by the scheduler. In some cases the modulation and coding scheme is determined by the mobile and reported to the BS. The modulation and coding scheme as well as the number of repetitions needed for a successful transmission determines the resulting packet transmission rate. Hybrid ARQ is used to take advantage of time diversity. Incorrectly received packets are NAKed and the BS retransmits the packet (in the case of Chase Combining) or some subset of the packet (in the case of Incremental Redundancy) to the mobile. The mobile combines the packet information with any earlier transmissions of the packet to increase the probability of correctly decoding the packet.

In this section we focus on the scheduling mechanism. Several papers have been written on scheduling over time-shared wireless packet data channels (e.g., see [4], [5]). The basic idea behind these so called opportunistic schedulers is to use the channel quality information reported by each mobile to serve them only when they are in good forward link channel conditions. Sector throughput can be maximized by always serving the user with the best forward link radio conditions or the maximum Carrier to Interference (C/I) ratio. This scheduler, typically referred to as a Max C/I scheduler, provides unfair service since users in poor radio conditions rarely get served. Fairness can be achieved by serving users in a round robin fashion but this approach does not take advantage of the variations in the users’ radio conditions. An approach in between these two extremes, called the Proportional Fair scheduler, provides each user with a throughput that is proportional to the value of their achievable rate (which in turn depends on their forward link radio conditions).

The above schedulers are suitable for best effort traffic since none provide any performance (throughput, delay, jitter etc.) guarantees. However, many applications (e.g., Voice over IP or VoIP, streaming media, etc) require some performance guarantees to ensure acceptable user performance. One approach for supporting such services is to use barrier function schedulers as proposed in [4]. Later in the paper we will argue that QoS application flows require minimal flow control and this argument will rest on the assumption that the scheduler can in fact support the necessary guarantees. We therefore briefly illustrate the effectiveness of these schedulers.

In [4] a barrier function scheduler was presented and its effectiveness was illustrated with a streaming media example. The paper considers a HSPDA forward link. The scheduler uses the following priority function in determining which user to serve at each scheduling decision point

\[ j^* = \arg \max_j \{ d_j(n)(1 - 2/3(r_j(n) - r_{cse})^2) \} \]
where $d_j$ represents the achievable rate of the user, $r_j$ represents the user’s average throughput, $\beta$ is a parameter that determines the stringency of the throughput requirement and $r_{cse}$ represents the desired throughput. Both $\beta$ and $r_{cse}$ will depend on the class (Gold or Silver) of the user. Note that the user’s priority increases as his throughput drops.

The particular example considered 24 streaming media users, 12 are Gold users requiring a constant 128 kbps throughput while the other 12 are Silver users requiring 64 kbps. The Gold users were all placed in worse radio conditions than the Silver users. This means that a Proportional Fair scheduler would have actually provided the Silver users with a higher throughput than the Gold ones. Figure 2 (taken from the paper) contains the throughput as a function of time for one Gold user and one Silver user. We also included the exponentially smoothed throughput of a circuit switched equivalent channel using the same filter constant. One can see that over a time scale of a few seconds, the throughput can be considered constant and any constant bit rate stream will be served with negligible delay.

In the above example the scheduler objective function uses a barrier function to penalize any movement away from the desired throughput. A similar mechanism can be used for delay sensitive traffic. In this case the barrier function is made dependent on the queuing delay at the scheduler and excessive delays incur higher penalties forcing service to the corresponding user. For example, [5] considers the case of VoIP service whereby voice frames are transported using the Internet Protocol. Since the user perceived performance of the conversation is highly dependent on the end-to-end delay, then a strict delay constraint must be imposed on the VoIP packets at the scheduler. In this case the priority function used by the scheduler is given by

$$j^* = \arg\max_j \left\{ \frac{d_j(n)}{q_{max} - q_j(n)^k} \right\}$$

where $q_{max}$ is the maximum queue length beyond which the delay becomes unacceptable and $k$ is a constant that determines the trade-off between packets that are lost due to transmission errors and those dropped due to a buffer overflow. Any packets that exceed the maximum queue length should be dropped since they will eventually be dropped at the receiver. As the delay limit is increased, the user capacity (i.e., the number of simultaneous users that can be supported with acceptable packet loss rates) also increases since the scheduler can take more advantage of user diversity gains and multiple voice frames can be packed into the physical layer frame (thus allowing the use of higher data rates). In Figure 3, taken from [5], we plot the user capacity as a function of this delay threshold.

The point of this plot is to illustrate that a scheduler can be designed to provide the QoS guarantees (in this case delay) for the corresponding application. In our subsequent discussions we will assume that if QoS guarantees are provided to a particular user application that the scheduler has the ability to provide those guarantees over the radio link.

Naturally, appropriate admission controls must be used to insure that all accepted application flows can be provided with their requested QoS guarantees for its duration. If this is not properly done then the scheduler will be unable to serve all users while maintaining their QoS guarantees. This will result in buffer overflows and degraded performance for all users. Even if an admission controller is used, there will still be time instances when the QoS guarantees for all users cannot be maintained. In such cases congestion control algorithms should be used to either reduce the number of active flows or decrease the performance guaranteed to one or more users based, for example, on their user class, radio conditions etc. In our subsequent discussion we assume that admission and congestion controls are used together with a QoS based scheduler.

Note that barrier function schedulers are one of many possible schedulers that can be used to provide QoS guarantees. The barrier function scheduler was chosen for its simplicity and wide applicability but the conclusions made in this paper are independent of the specific scheduler implemented.
III. Flow Control for QoS Traffic

In this section we consider the flow control needs for QoS traffic. In particular we consider specific applications and the flow controls that are needed to support them.

A. VoIP Flows

We assume that voice frames are generated every 20ms at the source and encapsulated in an IP packet. These IP packets are encapsulated into Radio Link Protocol (RLP) frames at the BSC and forwarded to the BS for transmission over the air. The BS scheduler may combine multiple voice frames into a physical layer frame before transmission to the mobile. The degree of this packing depends on the delay limit threshold used by the scheduler.

Suppose that no flow control is used between the BSC and the BS. Since the scheduler is designed for a total of 1 or 2% packet loss rate then radio resources must be provided to the flow to avoid packets violating the delay limit. Suppose for example that a delay limit of 120ms is used at the scheduler then at most 6 voice frames will need to be stored at the BS. Therefore no flow control is necessary between the BSC and BS as long as memory for 6 (full rate) voice frames is allocated at the BS. Also note that if a cell switch occurs, any voice frames queued at the BS should not be re-transmitted to the new cell since it is very likely they will be dropped at the mobile due to excessive delays anyway. Underflows are avoided since there is no flow control. Overflows do not matter since the packets would have been dropped anyway due to the delay limit. Hence we conclude that no flow control is needed for VoIP traffic.

B. Streaming Media Flows

We next consider a streaming media flow and assume that the streaming rate is constant at $r$ kbps. We can use a barrier function scheduler as described in the previous section with $r_{sec}$ set to $r$. In order to account for variations in the stream arrival rate as well as variations in the channel throughput, some buffering should be provided at the BS. For example if we want to buffer 0.5 seconds worth of traffic then a buffer of size $0.5r$ should be allocated. This buffering will depend on the play-out buffer size at the mobile. Of course any overhead should also be taken into account when allocating buffer space.

Again we avoid underflow at the BS since no flow control is used. If overflow occurs then the packet would have been dropped at the mobile anyway so it might as well be dropped at the BS to save on limited radio resources. Furthermore these packet drops will indicate to the application layer that the streaming rate should be reduced. Finally, any queued packets in a cell should not be re-transmitted to the new cell when a cell switch occurs since the packets would be to late anyway. Only subsequent frames should be forwarded to the new cell. So again we conclude that flow control is not needed for streaming media.

IV. Flow Control for Best Effort Traffic

In the previous section we concluded that flow control is not necessary for most QoS applications. In this section we address non-QoS or best effort traffic. One option is to control each best effort flow independently. This control has to be fast enough to continuously maintain a small but non-zero queue at the BS.

One simple approach would be to indicate to the BSC when the queue level drops below some threshold so that the BS can start forwarding data and also to indicate when the queue rises above some higher threshold so that the BSC can stop forwarding data. The peak rate that can be allocated to a mobile can be quite high and so sufficient memory must be allocated to each flow at the BS to support a large enough queue that can make use of this high peak rate. If this is not done then at times the rate assigned to a user will be limited by the amount of data in its queue rather than on the prevailing radio conditions. Therefore the minimum memory allocation per flow can be quite high.

Now consider a mobile in relatively bad radio conditions running a TCP application. Due to the nature of TCP flow control, as long as packets are not lost, the congestion window will continue to grow with packets accumulating at the BS buffer. If the mobile tries to run some interactive application using the same flow then the round trip time will be very large due to the large queue at the BS. In other words it would have been better to drop one or more packets earlier rather than wait for the queue to build. On could try to fix this by using different buffer sizes for different users but these would have to be dynamically set and the thresholds used for the flow control will also need to be changed dynamically. This requires additional signaling as well as measurement reports on the health of each flow etc and can become quite intricate.

We instead propose the following. Suppose, as is done on the Internet, we use a shared buffer for all best effort flows. Each flow uses as much memory as it needs. This will ensure that those flows that can make use of high rates have sufficient memory available to store data to support those rates.

Next we need to ensure that the queuing delay for any flow does not grow excessively. This can be monitored as follows. We assume that the BS keeps track of the number of bits $b_i$ queued for flow $i$ along with the average throughput $r_i$ of the flow. Whenever the estimated queuing delay $d_i = b_i / r_i$ exceeds some threshold $D$, then we drop all incoming RLP frames for the flow until the estimated queuing delay once again falls below the threshold $D$. Note that we assume once a flow terminates, either normally or abnormally, all RLP frames belonging to the flow are discarded. If we can distinguish IP packets at the BS then when the delay threshold is exceeded it would be preferable to drop entire IP packets by dropping all of its associated RLP frames.

TCP flow control increases the window size and hence the IP packet rate until congestion is detected through packet losses. Since the radio link throughput is limited, eventually the estimated queuing delay will increase until the threshold is
reached and RLP frames (and hence IP packets) are dropped resulting in TCP reducing its window size. So each flow will converge to an average queuing delay proportional to the limit $D$. Therefore we can note the following. The amount of storage used by a flow is proportional to its average throughput. Hence if the scheduler provides proportionally fair throughputs the memory allocation will also be proportionally fair. If a Max $C/I$ scheduler is used then those users experiencing high throughputs will also be provided with large buffers. This also means that those users who are capable of high data rates will experience high throughputs and hence be allocated large buffers. This means that data availability for these flows will not limit its high potential rates.

Once the expected delay of a flow exceeds the delay limit, another option to dropping RLP frames is to report the flow congestion to the BSC and have it either reduce traffic to that flow or drop RLP frames or IP packets for the flow. In fact in such a scenario the BSC can use the information to initiate Active Queue Management of IP packets for the flow.

The above mechanism provides limits on the memory resources assigned to users and hence limits any resource hogging (unless the scheduler is also explicitly allowing the user to use excess radio resources). However, even if each flow conforms to its expected queuing delay limit, the collective memory demands of the BS may exceed the available memory. First of all we compute an appropriate size for the shared buffer as follows. As discussed above, the memory allocated to a flow is proportional to its throughput $r_i$. Let $\alpha$ represent the proportionality constant. This can be obtained from measurements or simulations. We then have $b_i = \alpha r_i$. If we denote the sector capacity by $R$ and the total memory requirements by $B$ then we have

$$B = \sum_{i=1}^{N} b_i = \alpha \sum_{i=1}^{N} r_i = \alpha R$$

In other words, given the type of scheduler and the expected traffic mix, one can determine the sector throughput and then determine the appropriate shared buffer size. Note that if QoS traffic also exists then $R$ is determined from the the best effort capacity given the QoS flows.

Although the above computations can be used to estimate an appropriate size for the shared buffer, in practical systems the shared buffer may still be exhausted due to incorrect assumptions about the system. Therefore additional mechanisms must be included to address this issue. One simple mechanism is the following. As the occupancy of the shared buffer increases, we decrease the delay limit $D$ used for shedding the load for individual flows. Another option is to report buffer occupancy to the BSC and have it shed load by dropping IP packets.

**A. Performance Results**

We demonstrate the benefits of a shared buffer approach with a simple example. This example does not necessarily model the situation depicted in this paper but it provides some intuition as to why sharing buffer space (as is done in IP and ATM networks) is beneficial. Consider the case of $N$ flows and assume that each is flow controlled so that the probability of buffer overflow is $P_b$. Suppose we model each flow by a $M/M/1/K$ queuing system. Note that in reality the fact that the flow is controlled implies that the arrival rate is dependent on the queue size which is not the case of an $M/M/1/K$ queuing system. Since all flows have the same buffer size and blocking probability then they must also have the same utilization $\rho$ independent of the radio conditions of the radio link. We therefore have

$$P_b = \frac{(1 - \rho)\rho^K}{1 - \rho^{K+1}}.$$  

Next consider a shared buffer and suppose that the total flow is controlled at the same rate and assume that the sector throughput remains the same. In reality the sector throughput will increase because the probability of underflow of a flow is reduced and hence the user diversity gain increases. So our assumption that the sector throughput remains the same is pessimistic. If the the total offered load as well as the throughput remains the same then the utilization also remains the same namely $\rho$. Therefore a shared buffer of size $K$ is sufficient to maintain the same blocking probability. This is $1/N$ times the buffer requirements of the per flow buffer case.

**V. CONCLUSIONS**

We considered the issue of flow control between a BSC and BS for a wireless packet data channel. We showed that the flow control requirements are closely linked with the scheduler used at the BS for serving each user’s application flows. For the case of QoS flows we demonstrated that minimal flow control is needed since the scheduler has to provide the necessary resources to satisfy the flow’s QoS demands. For such flows, a proper choice of buffer size at the BS is sufficient for optimal performance. In the case of best effort flows we showed that instead of a per-application flow control, it is more efficient to share BS buffer resources among all best effort flows. This approach also has some nice properties in that the allocation of buffer resources maps closely with the allocation of radio resources by the scheduler. Based on the algorithms provided in this paper we believe that the flow controls presently used between the BSC and BS are not necessary.

**REFERENCES**


