PERFORMANCE EVALUATION OF SCHEDULING ALGORITHMS FOR BANDWIDTH RESERVATION

J. J. Harms and J. W. Wong

Department of Computer Science, University of Waterloo
Waterloo, Ontario, Canada N2L 3G1

The performance of scheduling algorithms for a bandwidth reservation system is investigated. In this system, a user request is characterized by its start time, bandwidth requirement, and holding time. Of interest are scheduling algorithms to handle user requests such that the channel bandwidth is effectively utilized. A loss system is considered, and a Markov decision process formulation is used to obtain the optimal scheduling decisions. Two special cases are considered in depth; they correspond to optimal algorithms that minimize the blocking probability and maximize the channel utilization respectively. Analytic results are also obtained for the blocking probability and channel utilization for an arbitrary scheduling algorithm. Using these results, the performance of FCFS and the two optimal algorithms is compared.

1. INTRODUCTION

A useful service offered by a telecommunication network is bandwidth reservation where communication channels are allocated to users on an advanced reservation basis. Such service finds applications in information broadcast to a community of users [1] and in videoconferencing [2]. An important aspect of a reservation system is the scheduling of user requests such that the network resources are effectively utilized. This paper is concerned with the performance of scheduling algorithms for reservation systems.

In a reservation system, a user request can be characterized by its start time, bandwidth requirement, and holding time [3]. The difference between the start time and the arrival time of a request is referred to as the notice interval. When the bandwidth available is finite, it is not always possible to accommodate all user requests. One approach to handling conflicts is to find an alternate feasible start time, preferably close to the requested start time. Another approach is to reject any requests that cannot be accommodated and leave it up to the user to resubmit a request for an alternate start time. The former is called a delay system while the latter is called a loss system.

An important aspect of a reservation system is the acknowledgement delay, which is the elapsed time from when a request arrives to when a scheduling decision on this request is made. In general, a user would prefer an acknowledgement delay of zero because the fate of his request is known immediately. A disadvantage of this approach is the reduced flexibility in bandwidth allocation. Flexibility can be gained by delaying the scheduling decisions until a batch of requests have been collected. This would, however, delay the acknowledgement of user requests. Our investigation is focused on loss systems with zero acknowledgement delay.

In general, exact analytic results for the performance of reservation systems are difficult to obtain; one often has to rely on the use of simple models, approximate analysis, or simulation. Most studies reported in the open literature are based on the assumption that each request requires the use of a single channel. These studies are also based on a slotted time model, i.e., channel time is slotted, the start time is at the beginning of a slot, and the holding time is an integer multiple of slots.

In [4], simulation is used to study the effect of notice interval on the performance of a loss system. Time is slotted, and scheduling decisions are made at the end of each slot. This implies that requests arriving during a particular time slot are batched for scheduling purposes. Another simulation study on the effect of notice interval is reported in [5]. That study is concerned with both loss and delay systems. In [6], dynamic programming is used to find exact analytic results for a delay system where a request is rejected if it is not possible to find an alternate start time within a given time period. The acknowledgement delay is assumed to be zero. Because of computational complexity, only systems with a small number of requests can be handled. Approximate results for large systems can be found in [7].

A loss system with zero acknowledgement delay is analyzed in [8]. In that system, each request has a preferred start time, but the user will accept an alternative within a flexibility interval. Approximate results for the blocking probability and the mean delay within the flexibility interval are obtained. In [9], an analysis of mean waiting time is presented for a delay system where the holding time of each request is exactly one slot. Performance evaluation of systems that serve a mixture of reservation requests and demand traffic (no advanced reservation) are available in [10,11]. Results are obtained for the case of zero acknowledgement delay and a first-come, first-served (FCFS) scheduling algorithm.

In this paper, we use a slotted time model to investigate the performance of scheduling algorithms for a loss system with zero acknowledgement delay. The holding time of each request is assumed to be exactly one slot. An important
feature considered in our model is that a user's bandwidth requirement is characterized by a discrete probability distribution (i.e., $\Pr$ [bandwidth requirement is $c$ units] = $y_c$, $c = 1, 2, \ldots, N$, where $N$ is the total bandwidth available). This feature is very useful in a broadband network environment where a user may request a fraction of the bandwidth of a high-speed channel. A Markov decision process formulation is used to obtain the optimal scheduling decisions. Two special cases are considered in depth; they correspond to optimal algorithms that minimize the blocking probability and maximize the channel utilization respectively. Analytic results are also obtained for the blocking probability and channel utilization for an arbitrary scheduling algorithm. Using these results, the performance of FCFS and the two optimal algorithms is compared.

This paper is organized as follows. Our reservation system model is described in Section 2. The Markov decision process formulation is presented in Section 3. Analytic results for an arbitrary scheduling algorithm are derived in Section 4, and numerical results showing the performance difference of FCFS and the two optimal scheduling algorithms are presented in Section 5. Finally, Section 6 contains a summary of our findings.

2. PERFORMANCE MODEL

In our model, time is slotted, and the holding time of each request is exactly one slot. A slot is further divided into $k$ minislots, and the minislot is used as our time unit (see Figure 1). At most one request may arrive in a minislot and such an arrival occurs with probability $\lambda$ ($\lambda$ is also the arrival rate). As mentioned previously, the bandwidth requirement is specified by a discrete probability distribution, i.e., a total of $N$ units of bandwidth is available, and $y_c$ is the probability that $c$ units are requested. The start time is characterized as follows. For a given arrival, the requested slot is selected from the next $L$ slots according to a probability distribution. The feasible start times are therefore the beginning of these slots (see Figure 1). For convenience, we assume that these slots are numbered from 1 to $L$, and use the random variable $x$ to denote the slot requested.

An important consequence of our single slot holding time assumption is that the reservation status of one slot can be treated independently of the others. We can therefore analyze each slot separately. Consider an arbitrary slot (say slot $\tau$). We show in Figure 2 the time period (of length $L$ slots) during which arrivals requesting that slot may occur. The notice intervals of these potential arrivals range from $T$ to 1 minislots, where $T = kL$ ($T$ and 1 correspond to the earliest and latest possible arrivals respectively). For an arrival with notice interval $t$, the corresponding random variable $x$ takes on a value $\lfloor t/k \rfloor$. Hence, the probability that an arrival has notice $t$ is given by:

$$q_t = \lambda \Pr[x=\lfloor t/k \rfloor]$$

where $l = \lfloor t/k \rfloor$. $q_t$ is a useful model parameter for our analysis in subsequent sections of this paper.

3. MARKOV DECISION PROCESS FORMULATION

In this section, a Markov decision process formulation is used to obtain the optimal scheduling decisions. This formulation is based on the observation that for our model, each slot can be analyzed separately. Consider slot $\tau$ in Figure 2. The state of the Markov decision process is defined to be $(i,j,n)$ where $1 \leq i \leq T$, $1 \leq j \leq N$ and $0 \leq n \leq N$. This corresponds to an arrival with notice $t$ and bandwidth requirement $c$ finding $n$ units of unallocated bandwidth at slot $\tau$. Note that the arriving request will be rejected if $c > n$; otherwise acceptance is feasible and a scheduling decision is required. We use an algorithm in [12] to obtain the optimal decision for each state (the decision is to accept or reject the arriving request). The optimal scheduling algorithm is then defined by the set of decisions for all feasible states.

Our approach is to study the system behavior in decreasing notice interval $t$. For convenience, we assume that the states are numbered from 1 to $M-1$ and state $i$ is denoted by $S_i = (t_i, c_j, n_i)$. We also define an initial state $S_0$ with $t_0 = T+1$ and $n_0 = N$, and a final state $S_M$ with $t_M = 0$ (the other parameters are not defined). Transition to a new state occurs when a request arrives; such an arrival will have a smaller notice interval since at most one arrival can occur in a minislot. From the initial state, the next state is determined by the first arrival requesting slot $\tau$. The final state is introduced to handle the case where no more arrivals for slot $\tau$ can occur.

A transition from $S_i$ to $S_j$ ($i, j < M$) is possible if the next request has notice $t_j < t_i$ and bandwidth requirement $c_j$, and the unallocated bandwidth in slot $\tau$ after the decision for $S_i$ is $n_j$. Let $q_d$ be the probability that the next request has notice $t_j$ when the current request has notice $t_i$, for $0 < t_j < t_i \leq T+1$.
(ti = T + 1) corresponds to the current state being S0 and Qij is the probability that the first arrival for slot τ has notice tj. Then,

$$Q_{ij} = q_i \prod_{r=t+1}^{t_i-1} (1-q_r)$$

(2)

where qi is given by (1).

We now consider the state transition probabilities. These probabilities depend on whether the decision is to accept or reject the request. The decision for Si (0 < i < M) is defined to be:

$$d_i = \begin{cases} 1 & \text{if the arrival in } S_i \text{ is accepted} \\ 0 & \text{otherwise} \end{cases}$$

(3)

Let pij(dij) be the transition probability from Si to Sj given decision di, 0 < i < M and 0 < j < M. Consider the case dj = 1 (accepting in Sj). The available bandwidth after acceptance is nj = ni - ci. Hence, the next state Sj must have nj = ni - ci. The transition probability pij(1) is then given by Qij yc,i where Qij and yc,i are the probabilities that the next request has notice tj and bandwidth requirement cj respectively. We thus have

$$p_{ij}(1) = \begin{cases} Q_{ij} y_{c,i} & \text{if } 0 < t_j < t_i \text{ and } n_j = n_i - c_i \\ 0 & \text{otherwise} \end{cases}$$

(4)

Similarly, for di = 0, we have

$$p_{ij}(0) = \begin{cases} Q_{ij} y_{c,i} & \text{if } 0 < t_j < t_i \text{ and } n_j = n_i \\ 0 & \text{otherwise} \end{cases}$$

(5)

The optimal decision for each state is found by choosing the decision that gives the higher expected reward. We consider an immediate reward function g(dj, cj) that depends on the bandwidth requirement of the request and the decision made in Si. The optimal scheduling algorithm is defined by the decision for each state that maximizes the overall expected gain. In [12], it is shown that the overall expected gain can be maximized by selecting di such that the following equation is satisfied (ties are broken in favor of accepting the request).

$$\nu_i = \max_{d_i} \left\{ g(d_i, c_i) + \sum_{j=1}^{M-1} v_j p_{ij}(d_i) \right\}$$

(6)

vj can be interpreted as the expected reward gained by starting the system in Sj (0 < i < M) and making the best decisions for Sj and for all states reached from Sj. If the states are ordered with non-increasing notice interval, (6) can be solved recursively by starting at state M-1.

Two special cases of the immediate reward function g(dj, cj) are now considered. The first case is to minimize the blocking probability. The corresponding g(dj, cj) is given by:

$$g(d_j, c_j) = \begin{cases} 1 & \text{if } d_j = 1 \\ 0 & \text{otherwise} \end{cases}$$

(7)

Equation (7) indicates that accepting a customer would yield a reward of serving one additional request. Maximizing the expected reward is therefore equivalent to minimizing the number of requests served, or minimizing the number of requests rejected. The optimal scheduling algorithm for this case will be referred to as BD.

The second case is to maximize the channel utilization, and the corresponding g(dj, cj) is

$$g(d_j, c_j) = \begin{cases} c_i & \text{if } d_j = 1 \\ 0 & \text{otherwise} \end{cases}$$

(8)

In this case, a higher reward occurs when more channel bandwidth is used. The optimal scheduling algorithm will be referred to as UD.

4. PERFORMANCE ANALYSIS

In this section, analytic results for the blocking probability and channel utilization are derived. The scheduling algorithm under consideration is arbitrary in the sense that only the decisions for the various states are required. The results will be used in the next section to compare the performance of FCFS, BD, and UD.

4.1. State Probabilities

Suppose that for an arbitrary scheduling algorithm A, the decision for Si is given by di(A). Consider again the timing diagram in Figure 2. Let yj be the probability that the first arrival for slot τ places the system in Sj. Since this arrival will find N units of unallocated bandwidth, we have, for 0 < j < M,

$$y_j = \begin{cases} Q_{ij} y_{c,i} & \text{if } n_j = N \\ 0 & \text{otherwise} \end{cases}$$

(9)

In general, let πj(r) be the probability that Sj is entered after r transitions. πj(0) = yj, and for 0 < r < T,

$$\pi_j(r) = \sum_{i=1}^{M-1} \pi_i(r-1) p_{ij}(d_i(A))$$

(10)

We are interested in determining πj, the probability that an arriving request with notice interval tj and bandwidth requirement cj, finds nj units of unallocated bandwidth at the requested start time. Since a state with notice interval tj cannot be entered with more than T - tj transitions, we have

$$\pi_j = \sum_{r=0}^{T-t_j} \pi_j(r).$$

(11)

Substituting (10) into (11), we get

$$\pi_j = y_j + \sum_{i=1}^{M-1} \sum_{r=1}^{M-1} \pi_i(r-1) p_{ij}(d_i(A))$$

(12)

Changing the order of summation, (12) becomes

$$\pi_j = y_j + \sum_{i=1}^{M-1} p_{ij}(d_i(A)) \sum_{l=0}^{T-t_j} \pi_l(l)$$

(13)
Since \( p_{ij}(d_i(A)) = 0 \) when \( t_i \leq t_j \), the upper limit of the second summation can be replaced by \( T-t_i \). Equation (13) is therefore reduced to
\[
\pi_j = \gamma_j + \sum_{i=1}^{M-1} p_{ij}(d_i(A)) \pi_i
\]  
(14)
Similar to \( \nu_i \), (14) can be solved recursively by starting at state 1, if the states are ordered with non-increasing notice interval.

From the state probabilities, performance measures such as blocking probability and channel utilization can be derived.

4.2. Blocking Probability

For our reservation system, blocking probability is given by the fraction of arriving requests that are rejected. It can be derived by considering the arrivals in a tagged slot (see Figure 3). Let the minislots in this tagged slot be numbered from 1 to \( k \). The feasible notice intervals for an arrival in minislot \( r \) are \( ik-r+1 \) for \( 1 \leq l \leq L \). This corresponds to a system state from the set \( S_l \). Let \( \alpha_r \) be the probability that an arrival in minislot \( r \) is rejected.
\[
\alpha_r = \sum_{S \in R_r} \pi_i (1 - d_i(A)).
\]  
(15)
The mean number of rejected arrivals in the tagged slot (denoted by \( \alpha \)) can be obtained by summing (15) over all minislots \( r \). This is identical to summing over all states \( S_l \). We thus have
\[
\alpha = \sum_{i=1}^{M-1} \pi_i (1 - d_i(A))
\]  
(16)
Finally, the mean number of arrivals in the tagged slot is \( k \lambda \).
The blocking probability \( b \) is therefore given by:
\[
b = \frac{\alpha}{k \lambda}
\]  
(17)

4.3. Channel Utilization

For our reservation system, channel utilization is given by the fraction of the channel bandwidth allocated to users. Suppose the system makes a transition from \( S_l \) to the final state \( S_M \). This happens with probability
\[
F_l = \prod_{r=1}^{L-l} (1 - q_r).
\]  
(18)
The fraction of channel bandwidth allocated is \( (N - n_l) / N \) if the decision is reject (i.e., \( d_i(A) = 0 \)) or \( (N - n_l + c_i) / N \) if \( d_i(A) = 1 \). The channel utilization is therefore given by:
\[
U = \frac{\sum_{l=1}^{M-1} \pi_l F_l [(1 - d_i(A))(N - n_l) + \alpha_i (N - n_l + c_i)]}{N}
\]  
(19)
Simplifying (19) yields
\[
U = \frac{\sum_{l=1}^{M-1} \pi_l F_l [N - n_l + c_i \alpha_i (N - n_l + c_i)]}{N}
\]  
(20)

5. NUMERICAL RESULTS

In this section, numerical examples are presented to compare the performance of FCFS, BD, and UD. Under FCFS, a request is accepted whenever acceptance is feasible. Our examples are based on a system with a total bandwidth \( N \) of 24 units. Each slot is divided into 10 minislots \( (k = 10) \). The random variable \( x \) which characterizes the start time is uniformly distributed over the values \( (1, 2, \ldots, 8) \). \( L \) is therefore given by 8. Three distributions for the bandwidth requirement are considered. They are (i) uniform over the values \( (1, 2, \ldots, 2\epsilon - 1) \), (ii) binomial with mean \( \epsilon \), and (iii) a two-value distribution where the bandwidth requirement has equal probability of taking on the values of 1 or \( 2\epsilon - 1 \). The mean bandwidth requirement is \( \epsilon \) for all three distributions. The arrival rate \( \lambda \) is selected such that it is not larger than \( \lambda^* = \min (1, N/k\epsilon) \), where \( N/k\epsilon \) is the arrival rate that would saturate the system under the condition that no requests are rejected.

We first consider the case of \( \epsilon = 4 \); the corresponding value of \( \lambda^* \) is 0.6. The blocking probability and channel utilization for each of the three bandwidth requirement distributions and for \( \lambda = 0.1, 0.2, \cdots, 0.6 \), are shown in Tables 1 to 3. Consider first the blocking probability. When the load is light (or \( \lambda \) is small), all three algorithms yield similar performance for the three bandwidth requirement distributions considered. As the load increases, the superiority of BD becomes noticeable. However, the performance difference of BD and FCFS is not significant, leading to the conclusion that FCFS is close to optimal with respect to minimizing blocking probability. The performance difference between BD and UD is more significant, especially in the case of the two-value distribution. This observation indicates that UD, which is optimized for channel utilization, may not perform as well when compared to FCFS.
similar performance when the load is light. As the load increases, the performance of UD and FCFS is similar except in the case of the two-value distribution where UD is superior. FCFS is still close to the optimal. The superiority of UD over BD is more noticeable. This indicates that BD, which is optimized for blocking probability, may not perform as well when compared to FCFS.

Among the three bandwidth requirement distributions considered, the two-value distribution yields the most significant performance difference between FCFS, BD, and UD. This can be explained as follows. BD tends to reject requests with large bandwidth requirements in order to accommodate more small requests in the future. This would have a negative impact on the channel utilization. UD, on the other hand, tends to reject requests with small bandwidth requirements in order to leave room for large requests that may utilize the bandwidth more fully. This would have a negative impact on blocking probability. The two-value distribution leads to the most significant performance difference between FCFS, BD, and UD because it represents the largest distinction between large and small requests.

We next consider the effect of the mean bandwidth requirement \( \bar{c} \) on performance. For our numerical examples, the value of \( \lambda \) is selected such that the channel utilization is 2/3 under the condition that no requests are rejected. Three values of \( \bar{c} \) are considered: they are 2, 4, and 6 units respectively. The corresponding values for \( \lambda \) are 0.8, 0.4, and 0.267. In Tables 4 and 5, we show the blocking probability and channel utilization for the three channel requirement distributions considered. These results are consistent with those in Tables 1 to 3 as far as the relative performance of FCFS, BD, and UD is concerned. In addition, we observe that as \( \bar{c} \) increases, the blocking probability also increases and the channel becomes less utilized. This is due to the fact that a larger mean bandwidth requirement (while keeping the total bandwidth available unchanged) would result in less flexibility in scheduling requests.

An interesting observation from the results in Tables 1 to 5 is that the channel utilization may not be very high when the blocking probability is already significant. For example, at \( \lambda = 0.4 \), the channel utilization is only around 60% while the blocking probability already exceeds 5%. A desirable operating condition for a reservation system is to keep the blocking probability small. In a loss system, this may result in the channel being under-utilized. One strategy to increase the channel utilization is to use the unallocated bandwidth for demand traffic, resulting in mixed traffic environment [10,11]. Another strategy is to consider a delay system where a user will accept any start time within a flexibility interval.

### 6. SUMMARY

We have used a Markov decision process formulation to obtain optimal scheduling decisions for a bandwidth reservation system. Our results indicate that FCFS is close to optimal with respect to minimizing the blocking probability or maximizing the channel utilization. Since FCFS is also simple and fair, it is our choice as the best algorithm for the reservation system under consideration. Our results also indicate that for a loss system, the channel may be under-utilized if the blocking probability is to be kept small. The channel utilization can be improved by considering a mixed reservation/demand traffic environment or a delay system.

### ACKNOWLEDGEMENT

This work was supported by the Natural Sciences and Engineering Research Council of Canada.
Table 4: Effect of $c$ on Blocking Probability

<table>
<thead>
<tr>
<th></th>
<th>Uniform</th>
<th>Binomial</th>
<th>Two-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c$</td>
<td>FCFS</td>
<td>BD</td>
<td>UD</td>
</tr>
<tr>
<td>2</td>
<td>1.87</td>
<td>1.83</td>
<td>1.87</td>
</tr>
<tr>
<td>4</td>
<td>6.72</td>
<td>6.65</td>
<td>6.72</td>
</tr>
<tr>
<td>6</td>
<td>11.66</td>
<td>11.62</td>
<td>11.66</td>
</tr>
<tr>
<td>2</td>
<td>2.06</td>
<td>1.94</td>
<td>2.06</td>
</tr>
<tr>
<td>4</td>
<td>6.51</td>
<td>6.41</td>
<td>6.51</td>
</tr>
<tr>
<td>6</td>
<td>11.28</td>
<td>11.23</td>
<td>11.28</td>
</tr>
<tr>
<td>2</td>
<td>2.05</td>
<td>1.98</td>
<td>2.26</td>
</tr>
<tr>
<td>4</td>
<td>6.56</td>
<td>6.54</td>
<td>7.47</td>
</tr>
<tr>
<td>6</td>
<td>10.25</td>
<td>10.24</td>
<td>12.97</td>
</tr>
</tbody>
</table>

Table 5: Effect of $c$ on Channel Utilization

<table>
<thead>
<tr>
<th></th>
<th>Uniform</th>
<th>Binomial</th>
<th>Two-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c$</td>
<td>FCFS</td>
<td>BD</td>
<td>UD</td>
</tr>
<tr>
<td>2</td>
<td>65.22</td>
<td>65.16</td>
<td>65.22</td>
</tr>
<tr>
<td>4</td>
<td>61.02</td>
<td>60.85</td>
<td>61.02</td>
</tr>
<tr>
<td>6</td>
<td>56.53</td>
<td>56.35</td>
<td>56.53</td>
</tr>
<tr>
<td>2</td>
<td>64.97</td>
<td>64.80</td>
<td>64.97</td>
</tr>
<tr>
<td>4</td>
<td>61.55</td>
<td>61.33</td>
<td>61.55</td>
</tr>
<tr>
<td>6</td>
<td>58.23</td>
<td>58.09</td>
<td>58.23</td>
</tr>
<tr>
<td>2</td>
<td>64.98</td>
<td>64.89</td>
<td>64.98</td>
</tr>
<tr>
<td>4</td>
<td>59.86</td>
<td>59.77</td>
<td>60.00</td>
</tr>
<tr>
<td>6</td>
<td>55.99</td>
<td>55.88</td>
<td>56.46</td>
</tr>
</tbody>
</table>

REFERENCES