Broadcast Delivery

JOHN W. WONG

Invited Paper

This paper is concerned with the architecture and performance of systems that use a broadcast channel to deliver information to a community of users. Information is organized into units called pages, and at any instant of time, two or more users may request the same page. Broadcast delivery is attractive for such an environment because a single transmission of a page will satisfy all pending requests for that page. Three alternative architectures for broadcast information delivery systems are considered. They are one-way broadcast, two-way interaction, and hybrid one-way broadcast/two-way interaction. An important design issue is the scheduling of page transmissions such that the user response time is minimized. For each architecture, existing scheduling algorithms are described, and their mean response time performance evaluated. Properties of scheduling algorithms that yield optimal mean response time are discussed. A comparative discussion of the performance differences of the three architectures is also provided.

I. Introduction

Broadcast channels have been used to deliver information to a community of users for several decades. This is exemplified by radio and television. Such systems use a policy called *scheduled broadcast* where users are informed as to when a particular piece of information will be transmitted. A user must wait until the scheduled time before the required information can be received. This is a simple and effective approach to serve a large population of users. Radio and television are examples of systems that use the concept of *broadcast delivery* for information dissemination.

Recent advances in computer and communication technologies have led to the development of information delivery systems that provide users with timely access to information. The basic configuration of such systems is shown in Fig. 1. Information is organized into units called *pages*, and stored on disk. Users submit requests and receive the requested pages through their terminals. The service computer retrieves the requested pages and transmits them to the users via a communication network. A database is maintained, and information pages in the database are updated regularly by service providers. These updates are issued locally or remotely at service provider terminals.

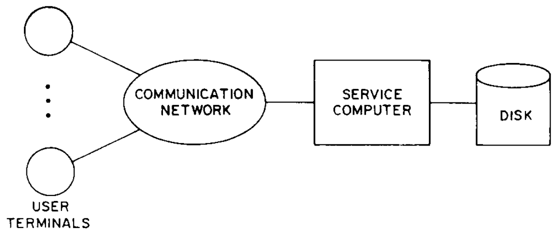


Fig. 1. A typical information delivery system.

A common example of an information delivery system is *teletext* [1]–[4] which uses a one-way broadcast channel. The service computer transmits the available pages to all users in a continuous manner, and there is no preannounced schedule as to when a particular page will be transmitted. When a user request is submitted, the terminal examines the incoming stream of data until the required page is detected. This page is then captured and displayed. Note that teletext uses the concept of broadcast delivery. Due to the one-way broadcast, a request does not propagate beyond the user terminal; this feature is sometimes described as *pseudointeractive*. A key design issue in teletext is the scheduling of page transmissions such that the user response time is minimized. Response time is defined to be the elapsed time from when a request is submitted to when the requested page is received.

Another common example of an information delivery system is *videotex* [1], [2], [5]–[8] which uses a two-way communication network. All user requests are submitted to the service computer where they are processed. Processing a user request involves the retrieval and transmission of the required page. Most videotex systems currently in operation are based on individual response and not on broadcast delivery [9]–[11], i.e., a page is transmitted to the requesting user only. With individual response, the operation of videotex is identical to that of an interactive computer system. It is rather intuitive that the response time performance can be improved if broadcast delivery is used. In this case, all pending requests for the same page will be serviced by a single transmission, resulting in a reduction of response

Manuscript received April 15, 1988; revised October 17, 1988. This work was supported by the Natural Sciences and Engineering Research Council of Canada.

The author is with the Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1. He is currently on leave at IBM Zurich Research Laboratory, CH-8803 Ruschlikon, Switzerland.

IEEE Log Number 8825166.

time when compared to individual response. It is therefore obvious that the performance benefits resulting from broadcast delivery should not be ignored. An example of an information system that uses broadcast delivery can be found in [12].

The key application of information delivery systems is the provision of access to information such as news, weather, financial information, and advertisement. Other useful applications include document retrieval and software distribution [13]. Broadcast delivery has also been suggested as a means to deliver frequently accessed information to processors in a distributed database system [14].

This paper is concerned with the architecture and performance of broadcast information delivery systems. Issues such as hardware, software, database organization, and transmission technology will not be considered. It is felt that the merit and potential of broadcast delivery are best addressed by decoupling the architectural concepts from the technologies used in system implementation. For that reason, the term *page* will be used in a rather general context, namely, it may contain any type of information to be delivered to the users.

Several factors of system design can affect the response time. Among them are the processing speed of the service computer, the capacity of the broadcast channel, and the disk characteristics. However, these factors are of a *static* nature since they depend, to a large extent, on the technologies used in system implementation. A more *dynamic* factor is the scheduling of page transmissions, i.e., the order in which the pages are broadcast to all users. Emphasis is therefore placed on the development of good scheduling algorithms and the evaluation of their response time performance.

Three alternative architectures for broadcast information delivery systems are considered in this paper. They are

i) one-way broadcast,
ii) two-way interaction, and
iii) hybrid one-way broadcast/two-way interaction.

Architecture i) is exemplified by teletext. Due to the one-way broadcast, system state information such as the number of pending requests for the various pages is not available to the service computer where the scheduling decisions are made. One can only rely on information such as how frequently a given page is requested by the users. One-way broadcast has the property that the response time is not affected by the load (or the rate at which requests are made). It can therefore support a potentially infinite user population. It also has the property that the response time tends to increase with the number of pages N. This is due to the fact that when N is large, more pages will likely be transmitted before the required page.

Architecture ii) is exemplified by videotex with broadcast delivery. Since all requests are submitted to the service computer, scheduling decisions can be based on system state information such as the order in which the requests are submitted or the number of pending requests for the various pages. This architecture requires the availability of a request channel (from user terminal to service computer), and has the undesirable property that the request channel may become overloaded when the volume of requests is heavy.

Architecture iii) is hybrid in the sense that some pages are serviced by one-way broadcast while the others are serviced by two-way interaction. Proposals for hybrid architectures can be found in [5], [9], [15].

Scheduling algorithms for page transmission have been developed for the three architectures mentioned above. In this paper, the mean response time performance of these algorithms is evaluated. The techniques used to obtain performance results include probability theory, queueing theory, Markov decision theory, and discrete event simulation.

This paper is organized as follows. Section II is devoted to one-way broadcast. A performance model is developed, and the mean response time of a simple probabilistic scheduling algorithm is derived. In that algorithm, page i is selected with probability p_i at each broadcast instance. A general treatment of the page transmission problem is considered next. Using a Markov decision process formulation, it is shown that transmitting pages according to a *broadcast cycle* (i.e., transmitting a fixed sequence of pages repeatedly) results in optimal mean response time. In general, a broadcast cycle may contain several appearances of each page, and the particular cycle that yields optimal mean response time cannot be determined easily. A procedure for designing a good broadcast cycle, i.e., a cycle that gives near-optimal, if not optimal mean response time, is presented. Finally, a detailed model of user behavior is considered, and the performance advantage of having local memory in the user terminal is discussed.

Section III deals with the two-way interaction architecture. The first scheduling algorithm considered is first-come, first-served (FCFS) with the modification that a new request joins the queue at the same position as an earlier request for the same page if such a request is in the system. Analytic results are derived for the mean response time. A Markov decision process formulation of the page scheduling problem is presented next. This formulation is based on the number of pending requests for the various pages. In general, it is not possible to characterize the scheduling algorithm that yields optimal mean response time. Using results for special cases, three heuristic scheduling algorithms have been proposed. Simulation results comparing the performance of these algorithms are discussed. The last algorithm considered in Section III is broadcast polling. In that algorithm, the service computer polls all user terminals to determine whether a given page is required, and that page is transmitted if a positive response is received. This is a restricted form of two-way interaction because a user request is not submitted until a poll indicating the required page is received.

Section IV is concerned with the hybrid one-way broadcast/two-way interaction architecture. This architecture is designed to avoid the performance penalties due to a large number of pages in one-way broadcast and request channel overload in two-way interaction. Pages are classified as either frequently or infrequently requested. Requests for frequently requested pages are serviced by one-way broadcast while those for infrequently requested pages are submitted to the service computer for processing. The service computer repeatedly transmits the next K frequently requested pages (according to a broadcast cycle) followed by a transmission of an infrequently requested page (if at least one such page is requested). Analytic results for the mean response time are derived, and the performance characteristics of the hybrid architecture are discussed.

In Section V, the performance of the three alternative architectures is compared. This is accomplished by characterizing the conditions under which a particular architecture is superior. Finally, Section VI contains some concluding remarks and a discussion of future research directions.

II. One-Way Broadcast

In this section, the modeling and performance evaluation of scheduling algorithms for the one-way broadcast architecture are discussed.

A. Performance Model

Most performance results are based on the model in [1], [16]. In that model, time on the broadcast channel is divided into equal size *slots*. The slot length represents the time to transmit a page of information. The arrival process of user requests is assumed to be Poisson. This assumption is accurate for systems that serve a large population of users. For each request, the probability that page i is requested is assumed to be q_i, $i = 1, 2, \cdots, N$, where N is the total number of pages. This assumption reflects the page usage statistics by all users (i.e., the frequency that a given page is requested), but not the page request pattern by a particular user (e.g., the sequence of pages requested). A model which takes the latter into consideration will be discussed in Section II-E.

B. Probabilistic Page Selection

A simple algorithm for scheduling page transmissions is probabilistic page selection [1]. That algorithm operates as follows. At each broadcast instance, page i is selected with probability p_i, $i = 1, 2, \cdots, N$. Let S_i be the mean response time of a request for page i. In deriving S_i, it is assumed that a request arriving during a page i transmission must wait for the next full transmission of page i. S_i is then given by

$$S_i = \frac{1}{p_i} + \frac{1}{2}. \tag{1}$$

Equation (1) is based on that fact that a request arriving at random (a consequence of the Poisson arrival process assumption) will wait on average half a slot for the current transmission to complete, plus $1/p_i$, the mean number of slots until the end of the next page i transmission.

Let S be the mean response time over all requests, we have

$$S = \sum_{i=1}^{N} \frac{q_i}{p_i} + \frac{1}{2}. \tag{2}$$

S is minimized when $p_i/p_j = \sqrt{q_i}/\sqrt{q_j}$ for all i, j. Since $\Sigma_{i=1}^{N} p_i = 1$, the best setting for p_i is given by

$$p_i = \frac{\sqrt{q_i}}{\sum_{j=1}^{N} \sqrt{q_j}} \tag{3}$$

for $i = 1, 2, \cdots, N$, and the resulting minimum mean response time is

$$\min S = \left(\sum_{j=1}^{N} \sqrt{q_j}\right)^2 + \frac{1}{2}. \tag{4}$$

A drawback of probabilistic page selection is that the response time may be arbitrarily large. Also, as we shall see later, probabilistic page selection is inferior to other algorithms for the one-way broadcast architecture.

C. Optimal Scheduling Policy

An important theorem on the scheduling of page transmissions is proved in [16]. It is based on a Markov decision process formulation and states that a cyclic policy is optimal as far as minimizing the mean response time is concerned. An outline of the proof is presented in this section.

In the Markov decision process formulation, the model described in Section II-A is used. Time slots on the broadcast channel are assumed to be numbered from slot 1, with slot k starting at time $k - 1$ and ending at time k. The following variables are also defined:

λ = arrival rate of user requests

$n_i(k)$ = number of requests for page i waiting at the beginning of slot k

$b_i(k)$ = number of arrivals requesting page i during slot k

$$u_i(k) = \begin{cases} 1, & \text{if page } i \text{ is transmitted during slot } k \\ 0, & \text{otherwise} \end{cases}$$

and

$Z_i(k)$ = number of slots elapsed since the beginning of the last page i transmission, until the beginning of slot k.

For convenience, it is assumed that initially, $Z_i(1) = 1$ for all i. This assumption does not affect the result because the optimal scheduling policy is derived for a system that runs for an infinitely long time.

The behavior of the model can be described by the following equations:

$$n_i(k + 1) = [1 - u_i(k)]n_i(k) + b_i(k) \tag{5}$$

and

$$Z_i(k + 1) = 1 + [1 - u_i(k)]Z_i(k). \tag{6}$$

To decide which page to transmit in slot k, it is useful to have information given by the $n_i(k)$'s. Due to the one-way broadcast, this information is not available to the service computer where the scheduling decisions are made. However, for a given sequence of page transmissions, the variables $Z_i(k)$ ($i = 1, 2, \cdots, N$; $k = 1, 2, \cdots$) are completely characterized (see (6)), and the service computer can determine the probability distribution of $n_i(k)$ for all i. Specifically,

$$\Pr\,[n_i(k) = n] = \frac{[q_i \lambda Z_i(k)]^n \exp\,[-q_i \lambda Z_i(k)]}{n!} \tag{7}$$

since the arrival process is assumed to be Poisson.

The vector $\underline{Z}(k) = (Z_1(k), Z_2(k), \cdots, Z_N(k))$ is used as the state in the Markov decision process formulation. For a given sequence of page transmissions, the expected cost of running the system until the end of slot t is defined to be

$$V_t = \sum_{k=1}^{t} \sum_{i=1}^{N} E[n_i(k)] \tag{8}$$

where $E[\cdot]$ is the expected value taken over all request arrival patterns. Using (7), V_t is given by

$$V_t = \sum_{k=1}^{t} \sum_{i=1}^{N} q_i \lambda Z_i(k), \tag{9}$$

and the long run average cost per unit time is

$$V = \lim_{t \to \infty} \frac{1}{t} V_t. \tag{10}$$

In [16], it is shown that if the above limit exists, V is related to the mean response time S as follows:

$$S = \frac{V}{\lambda} + 0.5. \tag{11}$$

Minimizing V will therefore minimize the mean response time also.

Two properties of the cost function V are proved in [16]. First, it is suboptimal to stop transmitting a page altogether, regardless of how low the page request probability is. Secondly, for a given set of page request probabilities, an upper bound on the inter-appearance gap (i.e., the time between successive transmissions of the same page) can be defined. For any page transmission sequence where the inter-appearance gaps of some pages exceed the upper bound, the cost can always be reduced by using a revised sequence with no gaps exceeding the bound.

The second property implies that a policy can be improved by revising the sequence of page transmissions such that $Z_i(k)$ is bounded from above for all i and k. Under this condition, the number of combinations of $Z_i(k)$'s is finite. This corresponds to a finite-state Markov decision process, and it is known that an optimal nonrandomized stationary policy exists [17]. For such a policy, the scheduling decision for slot k is uniquely determined by $\underline{Z}(k)$ and is independent of k. This implies that if the states at times k and l are equal, then those at times $k + m$ and $l + m$ are also equal for all m. Since the number of states is finite, some state must repeat. Thus the optimal policy is cyclic and the cycle length is the smallest integer L such that $\underline{Z}(k) = \underline{Z}(k + L)$ for all k.

To complete the informal argument, it is easy to see that the mean response time exists for a cyclic policy. This is supported by the fact that the worst-case response time is given by the cycle length L. One thus concludes that among all policies for which the mean response time exists, a cyclic policy is optimal. A sufficient condition for the existence of the mean response time is given in [16].

D. Design of Broadcast Cycle

A cyclic policy is characterized by transmitting pages according to a broadcast cycle. A simple example of a broadcast cycle is $(1, 2, \cdots, N)$. In general, a broadcast cycle may contain several appearances of each page, and the particular cycle that yields optimal mean response time cannot be determined easily. This is due to the time complexity in evaluating a large number of possible cycles. In [18], analytic results for the mean response time of any given cycle are derived. These results are then used to obtain a lower bound mean response time. A procedure to design a good broadcast cycle (i.e., a cycle that yields near-optimal, if not optimal, mean response time) is also presented. These results are discussed below.

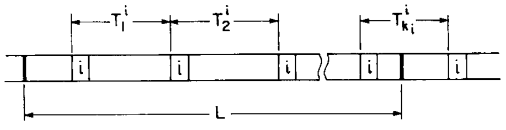


Fig. 2. A broadcast cycle.

Consider a cycle with finite length L and each page appearing at least once. The following useful parameters can be identified (see Fig. 2):

i) Appearance frequencies k_i, $i = 1, 2, \cdots, N$; k_i is the number of appearances of page i in the cycle.
ii) Inter-appearance gaps T_r^i, $r = 1, 2, \cdots, k_i$ and $i = 1, 2, \cdots, N$; T_r^i is the number of slots between the beginning of the rth and $(r + 1)$st appearance of page i in the cycle when $r < k_i$, and between the beginning of the last appearance in the current cycle and the beginning of the first appearance in the next cycle when $r = k_i$. Note that $\Sigma_{r=1}^{k_i} T_r^i = L$.

Consider a page i request arriving at random; the probability that this request will fall into the gap represented by T_r^i is T_r^i/L. Under this condition, the mean time until the beginning of the next full transmission of page i is $T_r^i/2$. The mean response time of a request for page i is therefore given by

$$S_i = \sum_{r=1}^{k_i} \frac{T_r^i}{L} \left(\frac{T_r^i}{2} + 1 \right), \tag{12}$$

and the mean response time over all requests is

$$S = \sum_{i=1}^{N} q_i S_i = \frac{1}{2L} \sum_{i=1}^{N} q_i \sum_{r=1}^{k_i} (T_r^i)^2 + 1. \tag{13}$$

Equation (13) can be used to compute the mean response time for any given cycle. It can also be used to develop a lower bound for the mean response time. Suppose the k_i's are given, the right-hand side of (13) is minimized when all the T_r^i's are equal, i.e.,

$$T_r^i = \frac{L}{k_i}, \qquad \text{for all } r. \tag{14}$$

We can thus write the following relationship:

$$S \geq \frac{L}{2} \sum_{i=1}^{N} \frac{q_i}{k_i} + 1. \tag{15}$$

The right-hand side of (15) is minimized when the k_i's are chosen such that

$$\frac{k_i}{k_j} = \frac{\sqrt{q_i}}{\sqrt{q_j}}, \qquad \text{for all } i, j. \tag{16}$$

Substituting (16) into (15) and noting that $\Sigma_{r=1}^{k_i} T_r^i = L$, we get

$$S \geq \frac{1}{2} \left(\sum_{i=1}^{N} \sqrt{q_i} \right)^2 + 1. \tag{17}$$

The lower bound in (17) corresponds to the best possible mean response time for any broadcast cycle. It is a useful result because the merit of any given cycle can be assessed by comparing its mean response time to the lower bound.

We now give a brief description of the cycle design procedure presented in [18]. The following criteria are used in the design:

i) The cycle length L must not exceed L^* (L^* is related to the amount of memory used to store the cycle);
ii) the inter-appearance gap of any page must not exceed T^*; this implies that the worst-case response time is $T^* + 1$; and
iii) the mean response time S is minimized.

The basic approach is to design a good cycle for each cycle length L in the range $N \leq L \leq L^*$, and select from among the cycles that meet criterion ii), the one with the lowest mean response time.

The algorithm to design a good cycle for a given L is given below [18]. Without loss of generality, the pages are ordered such that $q_1 \geq q_2 \geq \cdots \geq q_N$, and the cycle positions are numbered $0, 1, \cdots, L-1$.

1) Select integer k_i's such that $\Sigma_{i=1}^{N} k_i = L$ and k_i/k_j is as close to $\sqrt{q_i}/\sqrt{q_j}$ as possible for all i, j (see (16)).
2) For $i = 1$ to N, select integer T_r^i's such that $\Sigma_{r=1}^{k_i} T_r^i = L$ and that T_r^i is as close to L/k_i as possible for all r (see (14)).
3) For $i = 1$ to $N-1$, assign cycle positions to page i with the objective of matching the inter-appearance gaps obtained in step 2. Note that this is not always possible when $i > 1$ because one or more cycle positions needed for page i may have been assigned to pages with lower indices already (see [18] for more details).
4) Assign page N to the remaining free positions in the cycle.

The above procedure attempts to reduce the major contribution to the mean response time by assigning the popular pages first. The less popular pages receive less favorable treatment, and finally, the least popular page is assigned the remaining cycle positions, which may not be a good assignment for that page.

The cycle design procedure has been shown to yield cycles with near-optimal mean response time. As a numerical example, consider the following selection of parameter values: $N = 100$, $L^* = 1000$, $T^* = 300$, and q_i given by Zipf's law [19], i.e., $q_i = c/i$ where c is a normalization constant given by $c = (\Sigma_{j=1}^{N} 1/j)^{-1}$. Zipf's law has been shown to closely approximate the page request frequencies in teletext [1]. The results are shown in Fig. 3. It is observed that the best cycle has length $L = 743$. The mean response time of this

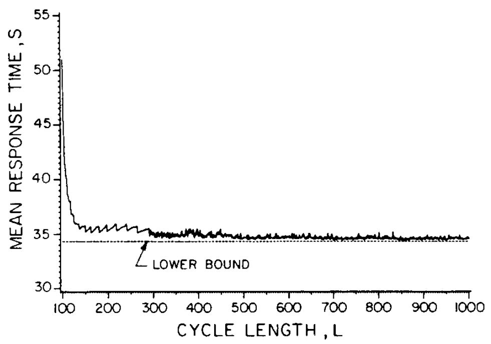


Fig. 3. Mean response time versus cycle length.

cycle is 34.4 which is very close to the lower bound of 34.3, as given by (17).

For completeness, the same parameter values are used to compare the performance of probabilistic page selection and cyclic page transmission. For probabilistic page selection, the minimum mean response time, as computed from (4), is 67.1. This is substantially higher than that for page transmissions using a good broadcast cycle.

Finally, it should be mentioned that one-way broadcast has the property that a given transmission may not satisfy any outstanding requests. Such a transmission is referred to as *superfluous*. From the performance point of view, superfluous transmissions are undesirable because they represent wastage of resources. An analytic expression for the fraction of transmissions that are superfluous is derived in [20]. In general, this fraction is a decreasing function of the arrival rate λ because as λ increases, the chance of a transmission satisfying at least one request is also increased.

E. An Individual User's Perspective of Response Time

The analytic results presented so far are based on the assumption that page i is requested with probability q_i. While this assumption is accurate for the derivation of mean response time over all requests, it does not provide sufficient detail to study the response time from the perspective of a single user. To study this perspective, a more detailed model of user behavior is developed in [21]. Two cases are considered:

i) Given request sequence case: The user has a prespecified sequence of requests, e.g., $i_1, i_2, \cdots, i_n$ where i_j is a page index and n is the number of requests in the sequence.
ii) Average request sequence case: The page request pattern is probabilistic and governed by the following parameters:

$$f_i = \Pr\,[\text{first request is for page } i] \text{ and}$$

$$p_{ij} = [(m+1)\text{th request is for page } j \mid m\text{th request is for page } i] \quad \text{for } m \geq 1 \text{ and } i, j = 1, 2, \cdots, N.$$

Page transmissions are based on a broadcast cycle. For both cases, the arrival of the first request is assumed to occur at random. When a request for page i is satisfied, a user will spend a think time at his terminal before submitting his next request. The think time is assumed to be exponentially distributed with mean dependent on the page index i. The exponential think time assumption is commonly used in performance analysis of interactive computer systems [22].

In [21], analytic expressions for the mean response time are derived for the two page request patterns mentioned above. These expressions are rather complicated and will not be presented here. Through numerical examples, it is shown that the mean response time seen by a single user can be influenced by a variety of parameters such as the user think time, the user request pattern, and the broadcast cycle used [21].

A one-way broadcast system where the user terminal has local memory is also considered in [21]. Performance improvement is possible if this local memory is managed in such a way that it contains the requested page most of the time. A strategy for memory management, called the *linked pages* scheme, is proposed in [21]. That scheme requires the availability of space in each page to store control information. The control information in page *i* is a list of linked pages; these are the most likely referenced pages after page *i*.

Let *D* be the size of local storage in unit of pages. After a request for a page (say page *i*) is satisfied, the user terminal enters a phase to prefetch the *D* most likely referenced pages associated with page *i*. This phase is terminated when the *D* pages are fetched or when the user submits a new request. Analytic results for the mean response time of the linked pages scheme are derived in [21]. A numerical example is also presented to show the performance improvement when the linked pages scheme is used.

III. Two-Way Interaction

In two-way interaction, user requests are submitted to the service computer where they are processed. Processing a request involves the retrieval of the required page and the transmission of that page to the user. As mentioned previously, current implementations of two-way systems are mostly based on individual response, but the performance advantage of broadcast delivery should not be ignored. For the same reason, it is expected that the use of broadcast delivery in two-way systems will gain popularity in the future.

A consequence of the two-way architecture is that detailed state information such as the order in which the requests are submitted or the number of outstanding requests for the various pages is available to the service computer. This is in contrast with one-way broadcast where such information is not available. In this section, the performance of a number of scheduling algorithms for the two-way architecture with broadcast delivery is discussed.

A. First-Come First-Served Scheduling

The first algorithm considered is first-come first-served (FCFS) with the modification that a new request joins the queue at the same position as an earlier request for the same page if such a request is in the system. Analytic results for the mean response time are derived in [23]. The model used is a single-server infinite-capacity queue with different types of requests, each corresponding to a particular information page (see Fig. 4). Once again, the arrival process of user

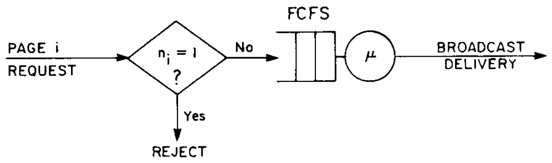


Fig. 4. Queueing model for FCFS scheduling.

requests is assumed to be Poisson with rate λ, and the probability that a request is for page *i* is q_i, $i = 1, 2, \cdots, N$.

The processing time of a request (i.e., the time to retrieve the requested page plus the time to transmit that page) is assumed to be exponentially distributed. The service rate μ is assumed to be the same for all request types. For mathematical tractability, the following assumptions are also made:

i) The delay at the request channel (from user terminal to service computer) is small compared to the processing time and can be ignored.
ii) At the end of processing a given page, all pending requests for that page are satisfied, regardless of whether these requests arrived before the start of processing or not. This is different from the one-way broadcast model where it was assumed that a request can only be satisfied by the first full transmission of the required page.

Let n_i be the number of page *i* requests in the system. The FCFS algorithm is modeled by rejecting a new request for page *i* if $n_i = 1$ (see Fig. 4). This will not affect the behavior of the model as far as the order in which requests are processed is concerned. Although a request may be rejected, it is responded to when the request for the same page in the system is processed.

The resulting model is a single server queue with different classes of customers (class *i* corresponds to requests for page *i*) and population size constraints [24]. Let $\underline{n} = (n_1, n_2, \cdots, n_N)$ be a state of the model. The population size constraint is characterized by *R*, the set of feasible states

$$R = (\underline{n} | n_i = 0 \text{ or } 1 \text{ for all } i).$$

Let $P(\underline{n})$ be the probability that the system is in state $\underline{n}$ at equilibrium. Using the results in [24], we have

$$P(\underline{n}) = P(0) n! \rho^n \prod_{j=1}^{N} q_j^{n_j} \tag{18}$$

where $\rho = \lambda/\mu$ is the traffic intensity, $n = \sum_{j=1}^{N} n_j$, and $P(0)$ is the probability that the system is idle.

Let $P(n)$ be the equilibrium probability that there are *n* requests in the system, regardless of type. This is a less detailed state description and can be obtained by summing over all state probabilities in (18) such that $\sum_{j=1}^{N} n_j = n$. $P(n)$ is given by

$$P(n) = n! \rho^n P(0) g(N, n) \tag{19}$$

where

$$g(N, n) = \sum_{\underline{n} \in R(n)} \prod_{j=1}^{N} q_j^{n_j}. \tag{20}$$

$R(n)$ is the set of feasible states such that $\sum_{j=1}^{N} n_j = n$. Since $\sum_{n=0}^{N} P(n) = 1$, we have

$$P(0) = \left(\sum_{n=0}^{N} n! \rho^n g(N, n) \right)^{-1}. \tag{21}$$

To derive the mean response time, the following events are defined for $i = 1, 2, \cdots, N$:

$A_i(n)$ = *n* requests in system and none of type *i*, $0 \le n < N$, and

$B_i(n, k)$ = *n* requests in system and *k*th request in FCFS order is of type *i*, $1 \le n \le N$ and $1 \le k \le n$.

By summing the appropriate state probabilities, it can be shown that [23]:

$$\Pr[A_i(n)] = n! \rho^n P(0) g^{-i}(N, n) \tag{22}$$

and

$$\Pr[B_i(n, k)] = (n-1)! \rho^n P(0) q_i g^{-i}(N, n-1) \tag{23}$$

where

$$g^{-i}(N, n) = \sum_{\underline{n} \in R(n) s.t. n_i = 0} \prod_{j=1}^{N} q_j^{n_j}. \quad (24)$$

If at the moment of a type i arrival, event $A_i(n)$ is satisfied, then the request is accepted and its mean response time will be $(n + 1)/\mu$. On the other hand, if event $B_i(n, k)$ is satisfied, the request is rejected, and its mean response time is given by k/μ. The mean response time of a request for page i can therefore be obtained by

$$S_i = \sum_{n=0}^{N-1} \Pr [A_i(n)] \frac{n+1}{\mu} + \sum_{n=1}^{N} \sum_{k=1}^{n} \Pr [B_i(n, k)] \frac{k}{\mu}. \quad (25)$$

Substituting (22) and (23) into (25) and after simplification, we get

$$S_i = \frac{P(0)}{\mu} \sum_{n=0}^{N} (n + 1)! \rho^n \left(g^{-i}(N, n) + \frac{q_i}{2} g^{-i}(N, n - 1) \right) \quad (26)$$

where $P(0)$ is given by (21), and $g^{-i}(N, N) = g^{-i}(N, -1) = 0$. The mean response time over all requests is given by $S = \Sigma_{i=1}^{N} q_i S_i$.

Note that the result for S_i is expressed in terms of $g(N, n)$ and $g^{-i}(N, n)$. Both terms involve a sum of products over a set of feasible states. Since the number of states grows exponentially with N, it is important that an efficient algorithm to compute the $g(\cdot)$ and $g^{-i}(\cdot)$ functions be available. Such a procedure can be found in [23]. It is based on the following recursive relationships:

i) $g(N, n) = g(N - 1, n) + q_N g(N - 1, n - 1)$
ii) $g^{-N}(N, n) = g(N - 1, n)$.

The derivation of these relationships is an extension of Buzen's work on computational algorithms for queueing network models [25]. Although relationship ii) is for page N only, numerical results for page $i \neq N$ can be obtained by re-ordering the page indices such that page i is the last page. The operation count to compute S_i for all i is $O(N^3)$.

As a numerical example, consider a model with the following parameter values: $N = 100$, $\mu = 1.0$, and page request probabilities given by Zipf's law [19]. The mean response time under broadcast delivery is shown in Fig. 5. The corresponding results for individual response are also shown. These results are based on a direct application of the M/M/1 queueing model [26]. It is observed that broadcast delivery yields significantly better response time performance than

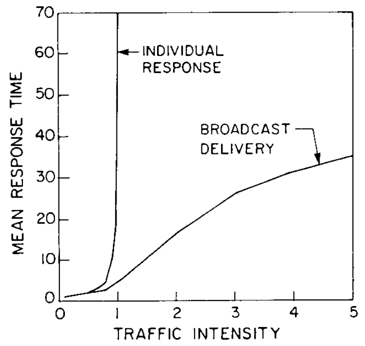


Fig. 5. Mean response time versus traffic intensity.

individual response. Furthermore, the mean response time under individual response becomes unbounded when the traffic intensity approaches 1.0. On the other hand, a system with broadcast delivery can handle a traffic intensity larger than 1.0. The above results indicate that for a given system configuration, the response time can be improved if broadcast delivery is used, and such an improvement can be achieved without increasing the processing capacity of the system.

It is important to note that the derivation of (26) is based on the assumption that the delay at the request channel is negligible. Therefore the conclusions drawn from the results in Fig. 5 are accurate only when the request channel is not overloaded (or when the request arrival rate is small compared to the service rate at the request channel).

B. Heuristic Scheduling Algorithms

An attempt is made in [27] to use a Markov decision process formulation to determine the scheduling algorithm that yields optimal mean response time. In that formulation, scheduling decisions are based on the number of pending requests for the various pages. The model used is shown in Fig. 6, and the assumptions are identical to those

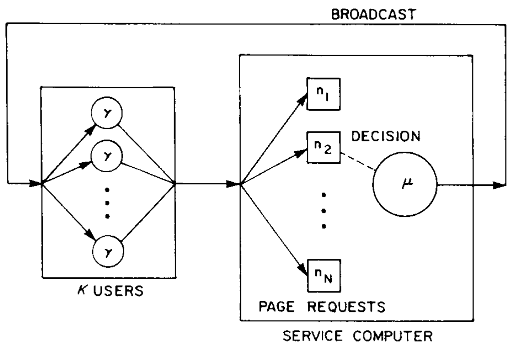


Fig. 6. Modeling of scheduling decisions in a two-way system.

described in Section III-A except that individual users are explicitly represented. More specifically, there are K independent users interacting with a service computer. The user think times are assumed to be independent and exponentially distributed with mean $1/\gamma$. With the explicit modeling of individual users, the number of states in the decision process is finite, and existing solution techniques can be applied directly.

The state of the decision process is defined to be $[(n_1, n_2, \cdots, n_N), j]$ where n_i is the number of pending requests for page i, $0 \le n_i \le K$, and $\Sigma_{i=1}^{N} n_i \le K$. The parameter j represents the state of the service computer. When $j = 0$, the computer is idle, and when $j > 0$, the service computer is processing the requests for page j. State transitions are caused by request arrivals and service completions. Consider a system in state $[(n_1, n_2, \cdots, n_N), j]$. When a page i request arrives, the system will enter state $[(n_1, \cdots, n_i + 1, \cdots, n_N), j]$. The transition rate is given by $q_i (K - \Sigma_{j=1}^{N} n_j)\gamma$. Transition out of a state due to service completion occurs at rate μ and is only possible when $j > 0$. The next state entered is $[(n_1, \cdots, n_j = 0, \cdots, n_N), j']$, since the broadcast of page j satisfies all pending requests for that page. Implicit in the transition is that the system has made a decision to process the requests for page j' next. The set of decisions for all feasible states defines a *scheduling policy*. The objective is to determine the scheduling policy that yields optimal mean response time.

In the Markov decision process formulation, cost is incurred at a rate of $\Sigma_{i=1}^{N} n_i$ per unit time. For a given scheduling policy, let $C(x)$ be the instantaneous rate at which cost is being incurred at time x. $C(x)$ is given by

$$C(x) = \sum_{i=1}^{N} n_i(x) \tag{27}$$

where $n_i(x)$ is the number of pending requests for page i at time x. If V_t is the total cost incurred up to time t, we have

$$V_t = \int_0^t C(x)\, dx, \tag{28}$$

and the average cost per unit time is given by

$$V = \lim_{t \to \infty} \frac{1}{t} \int_0^t C(x)\, dx. \tag{29}$$

Similar to (10), V is directly related to the mean response time if it exists. Therefore minimizing V will also minimize the mean response time.

Howard [28] has developed a policy-iteration algorithm which can be used to determine the optimal scheduling policy. That algorithm is based on an iterative procedure which starts with an arbitrary policy and selectively changes decisions until a policy that minimizes V is obtained. An immediate difficulty in applying Howard's algorithm is that the number of states grows rapidly with N and K, and consequently, it is not possible to obtain optimal scheduling policies for realistic settings of the model parameters.

The approach used in [27] is to apply Howard's algorithm to examples with small N and K, and try to identify properties of the optimal scheduling policy. These properties are used to develop the following heuristic scheduling algorithms for the two-way architecture:

i) Most Request First (MRF): Select the page with the largest number of pending requests; break ties in an arbitrary manner.
ii) MRF-Low (MRFL): Select the page with the largest number of pending requests; break ties in favor of the page with the lowest request probability.
iii) Longest Wait First (LWF): Select the page for which the total waiting time of all pending requests is largest.

It is conjectured that MRF is optimal when the page request probabilities are equal (i.e., $q_i = 1/N$ for all i). The other two algorithms are motivated by the observation that the optimal scheduling policy tends to give preferential treatment to pages with low request probabilities when the page request probabilities are not equal [27].

Numerical results are presented in [27] to compare the performance of the above scheduling algorithms and FCFS. The key observations are as follows. When the load is light, the mean response time is insensitive to the scheduling algorithm used. This is to be expected because few scheduling decisions are required. As the load increases, MRF yields the best response time performance for the case of equal page request probabilities. This is consistent with the conjecture mentioned above. For the more interesting case of page request probabilities given by Zipf's law [19], the mean response time results are shown in Fig. 7. These results

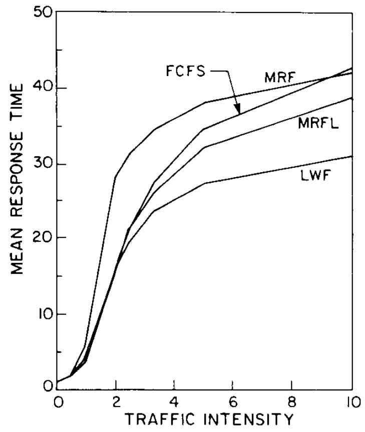


Fig. 7. Mean response time versus traffic intensity.

are for a system with 100 pages and $\mu = 1$ (the curve for FCFS is identical to that in Fig. 5). We observe that LWF has the best performance, and MRFL is second best. It should be noted, however, that from the implementation point of view, LWF incurs more scheduling overhead than the other algorithms. If this overhead is an important issue, then MRFL is a good alternative.

C. Broadcast Polling

An interesting algorithm to schedule page transmissions in a two-way system is reported in [20]. That algorithm is called *broadcast polling* and is different from the scheduling algorithms mentioned so far in the way user requests are submitted to the service computer.

Broadcast polling can best be explained by considering its implementation on a unidirectional ring network [20]. The basic configuration is shown in Fig. 8. The service computer polls the users by transmitting a short polling message around the ring. This message contains a page identifier (say page i) and an indicator bit which is initially cleared to zero. A user terminal, upon receiving the polling mes-

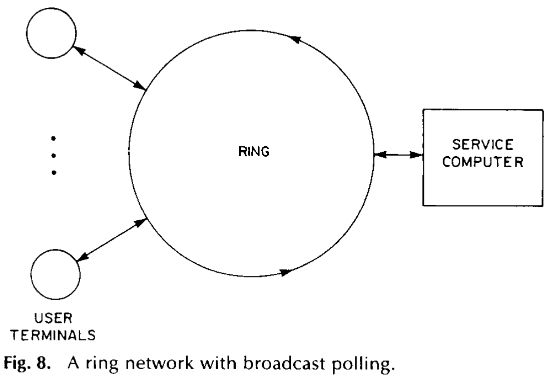


Fig. 8. A ring network with broadcast polling.

sage, sets the indicator bit to one if it has a request for page i. When the polling message returns to the service computer, page i is transmitted if the indicator bit has been set. The order in which the various pages are polled is defined by a cycle.

Broadcast polling has been suggested as an alternative to one-way broadcast [20] because superfluous transmissions are avoided. Simulation results showing the performance characteristics of broadcast polling are presented in [20]. Bounds are also derived for S_{bp}, the mean response time under broadcast polling. The derivations are summarized below.

Suppose the time to transmit an information page is one unit, and that to transmit a polling message is H. At heavy load, almost all polling messages return with a positive indication, and the broadcast polling scheme approaches a one-way architecture with a page transmission time of $1 + H$. If C is the cycle used to poll the various pages, then

$$S_{bp} \leq (1 + H)S(C) \tag{30}$$

where $S(C)$ is the mean response time of a one-way system using broadcast cycle C, and is given by (13). Equation (30) indicates that at heavy load, the broadcast polling scheme is inferior to one-way broadcast.

To derive a lower bound, we note that at light load, almost all polling messages return with a negative indication, and an arriving request experiences what looks like a one-way system with page transmission time equal to H, until the requested page is transmitted. We thus have the following relationship:

$$S_{bp} \geq HS(C) + 1. \tag{31}$$

Since H is typically much smaller than 1 (the page transmission time), broadcast polling is superior to one-way broadcast at light load.

Finally, it should be mentioned that the problem of designing a good cycle for broadcast polling is not straightforward because the best cycle is dependent on the request arrival rate [20].

IV. Hybrid One-Way Broadcast/Two-Way Interaction

It is expected that in the future, information delivery systems will have many pages and will serve large user populations. One-way broadcast is suitable for large user populations but the response time tends to increase with the number of pages. Two-way interaction, on the other hand, may suffer from request channel overload when the request arrival rate increases. A hybrid architecture which is appropriate for large-scale information delivery systems is investigated in [29]. In that architecture, pages are classified as either frequently requested or infrequently requested. Requests for frequently requested pages (f-requests) are serviced using a broadcast cycle (one-way broadcast), while those for infrequently requested pages (i-requests) are submitted to the service computer for processing (two-way interaction).

By servicing i-requests on demand, the hybrid architecture reduces the number of pages that are serviced according to a broadcast cycle. This leads to good response time performance for the f-requests. If the pages are classified such that the volume of i-requests is low, the demand placed on the request channel and service computer will not be significant. This leads to good response time performance for the i-requests also.

A model is developed in [29] to investigate the performance characteristics of the hybrid architecture. That model is shown in Fig. 9. Once again, the arrival process of user

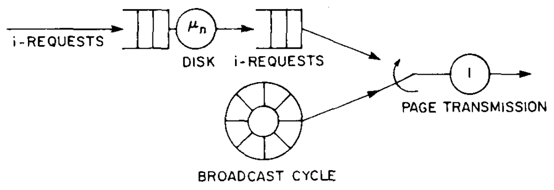


Fig. 9. Model of a hybrid architecture.

requests is assumed to be Poisson with rate λ and the probability that a request is for page i is assumed to be q_i, $i = 1, 2, \cdots, N$. The information pages are ordered such that $q_1 \geq q_2 \geq \cdots \geq q_N$. The first F pages $(0 \leq F \leq N)$ are classified as frequently requested, and the remaining $N - F$ pages as infrequently requested. It follows that the arrival rates of f-requests and i-requests are $\lambda_F = \lambda \sum_{j=1}^{F} q_j$ and $\lambda_I = \lambda \sum_{j=F+1}^{N} q_j$ respectively. The page transmission time is assumed to be constant and equal to one time unit, and the procedure described in Section II-D is used to design a broadcast cycle for the F frequently requested pages.

In addition, the following two assumptions are made for mathematical tractability:

i) The delay experience by an i-request at the request channel is small compared to the disk service time and page transmission time, and can be ignored. This is accurate when the volume of i-request is low.
ii) The i-requests are serviced using individual response even though broadcast delivery is employed. The error in the mean response time is not significant if the volume of i-requests is low because under that condition, the chance of having two or more outstanding i-requests for the same page is minimal.

With the above assumptions, there is no need to model the request channel and therefore an i-request is first processed by the disk server. The retrieved page is placed on the i-request queue. The disk is modeled by an exponential server with service rate μ_n, which is dependent on the number of pages n to be retrieved. The state dependent service rate is an accurate representation of disk scheduling algorithms that are designed to reduce the disk seek time, e.g., SCAN [30]. The algorithm for scheduling page transmissions operates as follows. The service computer makes K consecutive transmissions of frequently requested pages (according to a broadcast cycle), followed by the transmission of the first page in the i-request queue (if at least one such page is waiting). The above step is repeated indefinitely.

Analytic results for the mean response time are derived in [29]; a brief outline of the analysis is given below. For the i-requests, the mean response time has two components. The first component corresponds to the mean response time at the disk server which is given by standard results for an M/M/1 model with state dependent service rates [26]. The second component corresponds to the mean response time at the i-request queue, and is obtained by a discrete time analysis.

The f-requests are serviced according to a broadcast cycle. However, there are instances where a slot may be inserted for the transmission of a page in the i-request queue. The mean response time of f-requests is therefore given by the results in (12) and (13), modified to account for such insertions. Let S_I and S_F be the mean response time of i-requests and f-requests respectively. The mean response time over all requests is given by $S = \lambda_I S_I + \lambda_F S_F / \lambda$.

Numerical results on the performance of the hybrid architecture are presented in [29]. The key observations are as follows. A desirable operating condition is that λ_I is small compared to the service rates of the disk and transmission servers. Under this condition, the queueing delay experienced by the i-requests is minimal, and the number of slots inserted into the broadcast cycle for the f-requests is not significant. When the load (or λ) increases, λ_I may also increase and the above condition should be maintained in order to achieve good response time performance. One possibility is to reduce K; this would improve the effective service rate of the transmission server with respect to the i-requests. This approach may not be effective because the disk server may be the bottleneck. A more effective approach, however, is to reduce λ_I by classifying more pages as frequently requested (i.e., by increasing F). The typical behavior is shown in Fig. 10 where the mean response time

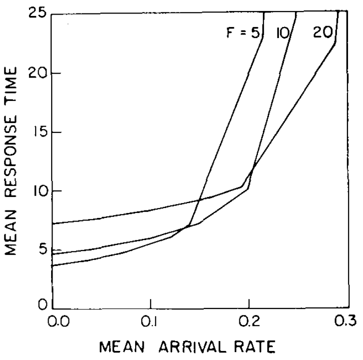


Fig. 10. Mean response time versus arrival rate.

is plotted against the mean arrival rate for the following parameter settings: 1000 pages, Zipf's law page request probabilities, $K = 5$, and a constant disk service rate of 1.0. A simple cycle, i.e., $(1, 2, \cdots, F)$, is used to transmit the frequently requested pages. We observe that as F increases, the system can handle a heavier arrival rate. This is due to the fact that the volume of i-requests is reduced. However, more pages are classified as frequently requested, and as a result, the mean response time of f-requests becomes higher. This explains the degradation in mean response time over all requests at light load.

The above discussion suggests that an adaptive scheme could be used. In that scheme, the number of pages classified as frequently requested is changed dynamically according to the request arrival rate, such that the volume of i-requests is low enough to achieve good response time performance.

V. Comparison of the Three Architectural Alternatives

The discussions in Sections II–IV have focused on the performance of scheduling algorithms for a particular architecture. In this section, the relative merits of the three architectural alternatives are considered.

An immediate observation is that existing results are derived using models that are developed specifically for a given architecture. Very often, assumptions are made for reasons of mathematical tractability. As a result, the assumptions used in one model may not be completely identical to those used in another model. For example, in one-way broadcast, page retrieval is not modeled, the page transmission time is assumed to be constant, and a user request is not satisfied until the next full transmission of the requested page is complete. In two-way interaction, page retrieval and page transmission are combined into a single processing time which is assumed to have an exponential distribution. Also, a user request can be satisfied by the current processing of the requested page. Finally, in the hybrid architecture, page retrieval and page transmission are modeled separately. As a result of these differences in model assumptions, existing results cannot be used to compare the performance of the alternative architectures. However, we can make some general remarks about their performance differences.

First, one-way broadcast has the undesirable property that superfluous transmissions may occur. Such transmissions can be avoided in a two-way system. Hence, the two-way architecture has the potential of better resource utilization. More detailed state information is also available in a two-way system, and in principle, more effective scheduling algorithms for page transmission can be designed. These performance advantages are, however, gained at the expense of requiring a request channel.

An important consideration in the two-way architecture is the amount of traffic on the request channel. This channel may become a bottleneck if the load is heavy. One approach to reduce the amount of traffic is to use the hybrid architecture because the f-requests will not be transmitted to the service computer. Another approach is to use broadcast polling where all pending requests for the same page are submitted in one polling message. Within the two-way architecture, there is also a tradeoff between the volume of traffic on the request channel and the amount of state information available to the service computer. When requests are submitted separately, the service computer will have detailed state information, but all requests will be transmitted on the request channel. On the other hand, when a scheme like broadcast polling is used, the amount of traffic on the request channel is reduced, but the service computer only knows whether there are any pending requests for the page indicated in the poll. It does not have state information for the other pages.

With respect to mean response time performance, the following observations can be made:

i) When the request arrival rate is low, the two-way architecture is best. This is rather intuitive because a request can be processed quickly, and there is no need to wait for the required page to be transmitted according to a broadcast cycle.
ii) When the request arrival rate is high, the two-way architecture may suffer from excessive delay because

the request channel may become overloaded. This can be alleviated by using broadcast polling, the hybrid architecture, or one-way broadcast.

iii) One-way broadcast is not effective when the number of pages is large. However, it is still the best architecture when the arrival rate is very heavy.

iv) The hybrid architecture seems to be best among the three architectural alternatives because by suitable selection of parameters, it possesses the advantages of both one-way broadcast and two-way interaction.

Apart from response time performance, it is important to consider the variety of services that can be provided by the three alternative architectures. A brief discussion is given below. For one-way broadcast, the service provided is restricted to information delivery only. Two-way interaction is more versatile in the sense that transaction oriented services can also be provided (similarly for the hybrid architecture).

In a two-way system, one can visualize a scenario where some requests require individual response (e.g., transaction oriented) while the others can be serviced by broadcast delivery. This is referred to as *mixed delivery* [31]. The model in Fig. 4 has been extended to include mixed delivery in [31]. Specifically, there are $N + 1$ request types: type 0 corresponds to those requiring individual response and for type i, $i = 1, 2, \cdots, N$, the request is for page i which can be serviced by broadcast delivery. The same FCFS discipline is used except that a type 0 request always joins the queue.

Analytic results for the mean response time of mixed delivery are available in [31]. Let λ_0 be the mean arrival rate of type 0 requests and λ_b be the total arrival rate of all other request types. The corresponding traffic intensities are denoted by $\rho_0 = \lambda_0/\mu$ and $\rho_b = \lambda_b/\mu$ respectively, where $1/\mu$ is the mean processing time. The results in [31] indicate that the mean response time becomes unbounded when ρ_0 approaches 1, but remains finite even when ρ_b is larger than 1. It is also observed that for requests that can be satisfied by broadcast delivery, the presence of type 0 requests has the same effect as if the processing capacity were reduced by a fraction equal to ρ_0. The same property has been observed in analytic solutions to mixed queueing network models [32].

VI. Concluding Remarks

Recent research in modeling and analysis has led to a good understanding of the performance of broadcast information delivery systems. The available results have been reviewed in this paper, and their derviations provided wherever appropriate. It is expected that future information delivery systems will become more complex. Additional research is therefore needed to further understand their performance characteristics. Some potential research problems are discussed below.

When modeling one-way broadcast, page retrieval is not included in the model. An implicit assumption made is that a page is always in memory when it is needed for transmission. This might not be true if the system does not have sufficient main memory to store all pages. Additional constraints may therefore be imposed on the broadcast cycle design. As an example, a service computer may be forced to transmit some other page in memory because the required page is still on disk. It is of interest to study the broadcast cycle design problem under the condition that there is insufficient memory to store all pages.

The issue of insufficient main memory to store all pages is also relevant in the two-way interaction architecture. Under this condition, algorithms are needed for disk scheduling and memory page replacement, as well as for page transmission. Some preliminary results on the performance of these algorithms are available in [33]. In that paper, it was observed that good algorithms for page transmission (e.g., MRF or LWF depending on whether the page request probabilities are equal or not) should be used for disk scheduling and page replacement also, except when page retrieval is significantly slower than page transmission. In that case, a disk scheduling algorithm which minimizes the seek time should be used.

For two-way interaction, we have observed a tradeoff between the volume of traffic on the request channel and the amount of information available to the service computer. Further investigation of this tradeoff will undoubtedly lead to new algorithms for scheduling page transmissions. A related problem is the modeling of the request channel. This has been avoided in current performance studies mainly because of mathematical tractability, but becomes important when the volume of user requests is heavy.

The effect of local storage at a user terminal on performance has been discussed in Section II-E. Results are available for one-way broadcast only. The corresponding results for the other two architectures will also be of interest. An idea worth further investigation is the grouping of related pages into a *page set*. When a page is requested, attempts should be made to receive the related pages and store them locally. This can be accomplished by waiting for the transmission of the required pages (an approach similar to the linked pages scheme in Section II-E), or by submitting a request for a collection of related pages. Alternatively, the service computer may broadcast the page set as a group.

Finally, a model which allows a fair performance comparison of the various architectures for information delivery systems is needed. Such a model should include the request channel, finite memory in the service computer, one or more disks, and the communication channel for page transmission.

References

[1] J. Gecsei, *The Architecture of Videotex Systems.* Englewood Cliffs, NJ: Prentice-Hall, 1983.

[2] A. F. Alber, *Videotex/Teletext: Principles and Practices.* New York, NY: McGraw-Hill, 1985.

[3] J. R. Storey, A. Vincent, and R. Fitzgerald, "A description of the broadcast telidon system," *IEEE Trans. Consumer Electron.*, vol. CE-26, no. 3, 1980.

[4] P. Mothersole, "New information systems using the domestic television receiver," *Proc. Inst. Elec. Eng.*, vol. 126, no. 12, 1979.

[5] A. J. Ball, G. V. Bochmann, and J. Gecsei, "Videotex networks," *Computer*, vol. 13, no. 12, pp. 8–14, Dec. 1980.

[6] F. W. Tompa, J. Gecsei, and G. V. Bochmann, "Data structuring facilities for interactive videotex systems," *Computer*, vol. 14, no. 8, pp. 72–81, Aug. 1981.

[7] M. Sugimoto, M. Taniguchi, S. Yokoi, and H. Hata, "Videotex: advancing to higher bandwidth," *IEEE Commun. Mag.*, vol. 26, no. 2, pp. 22–30, Feb. 1988.

[8] G. D. Ott and J. L. Strand, "Videotex tutorial," *J. Telecommunication Networks*, vol. 2, no. 3, pp. 385–398, 1983.

[9] M. L. Ellis, G. W. Gates, J. M. Smith, G. L. Peckham, and H. P. Gray, "INDAX: an operational interactive cabletext sys-

tem," *IEEE J. Selected Areas Commun.*, vol. SAC-1, no. 2, pp. 285-294, Feb. 1983.

[10] R. A. von Vignau, "Bildschirmtext and the CEPT videotex system," *IEEE J. Selected Areas Commun.*, vol. SAC-1, no. 2, pp. 254-259, Feb. 1983.

[11] S. Harashima, T. Kumamoto, and T. Kitamma, "Japanese videotex system captain—experimental service and user outline," *IEEE Trans. Commun.*, vol. COM-29, no. 12, Dec. 1981.

[12] D. K. Gifford, J. M. Lucassen, and S. T. Berlin, "The application of digital broadcast communication to large-scale information systems," *IEEE J. Selected Areas Commun.*, vol. SAC-3, no. 3, pp. 457-467, May 1985.

[13] J. Hedger and R. Eason, "Telesoftware: adding intelligence to teletext," *Proc. Inst. Elec. Eng.*, vol. 126, Dec. 1979.

[14] G. Herman, G. Gopal, K. C. Lee, and A. Weinrib, "The datacycle architecture for very high throughput database systems," in *Proc. ACM SIGMOD International Conf. on Management of Data*, (San Francisco, CA), pp. 97-103, 1987.

[15] G. Robinson and W. Loveless, "Touch-tone teletext: a combined teletext-viewdata system," *IEEE Trans. Consumer Electron.*, vol. CE-25, no. 3, pp. 298-303, July 1979.

[16] M. H. Ammar and J. W. Wong, "On the optimality of cyclic transmission in teletext systems," *IEEE Trans. Commun.*, vol. COM-35, no. 1, pp. 68-73, Jan. 1987.

[17] C. Derman, "On sequential decisions and Markov processes," *Management Sci.*, vol. 9, 1962.

[18] M. H. Ammar and J. W. Wong, "The design of teletext broadcast cycles," *Performance Evaluation*, vol. 5, no. 4, pp. 235-242, Dec. 1985.

[19] G. K. Zipf, *Human Behavior and the Principle of Least Effort.* Reading, MA: Addison-Wesley, 1949.

[20] M. H. Ammar, "Teletext-like information delivery using broadcast polling," *Computer Networks and ISDN Systems*, vol. 12, no. 2, pp. 107-115, 1986.

[21] ——, "Response time in a teletext system: an individual user's perspective," *IEEE Trans. Commun.*, vol. COM-35, no. 11, pp. 1159-1170, Nov. 1987.

[22] R. R. Muntz, "Analytic modeling of interactive systems," *Proc. IEEE*, vol. 63, June 1975.

[23] J. W. Wong and M. H. Ammar, "Analysis of broadcast delivery in a videotex system," *IEEE Trans. Comput.*, vol. C-34, no. 9, pp. 863-866, Sept. 1985.

[24] S. S. Lam, "Queueing networks with population size constraints," *IBM J. Res. Develop.*, vol. 21, no. 7, pp. 370-378, July 1977.

[25] J. P. Buzen, "Computational algorithms for closed queueing networks with exponential servers," *Commun. ACM*, vol. 16, no. 9, pp. 527-531, Sept. 1973.

[26] L. Kleinrock, *Queueing Systems Vol. 1: Theory.* New York, NY: Wiley Interscience, 1975.

[27] H. D. Dykeman, M. H. Ammar, and J. W. Wong, "Scheduling algorithms for videotex systems under broadcast delivery," in *Proc. International Conf. on Communications*, (Toronto, Canada), pp. 1847-1851, 1986.

[28] R. A. Howard, *Dynamic Programming and Markov Processes.* Cambridge, MA: M.I.T. Press, 1960.

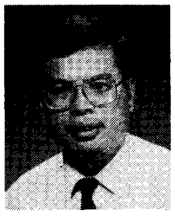
[29] J. W. Wong and H. D. Dykeman, "Architecture and performance of large scale information delivery networks," in *Proc. 12th International Teletraffic Congress*, (Torino, Italy), 1988.

[30] P. J. Denning, "Effects of scheduling on file memory operations," *AFIPS Joint Computer Conference Proceedings*, vol. 30, pp. 9-21, 1967.

[31] J. W. Wong and M. H. Ammar, "Response time performance of videotex systems," *IEEE J. Selected Areas Commun.*, vol. SAC-4, no. 7, pp. 1174-1180, Oct. 1986.

[32] M. Reiser and H. Kobayashi, "Queueing network models with multiple closed chains: theory and computational algorithms," *IBM J. Res. Develop.*, vol. 19, pp. 283-294, 1975.

[33] H. D. Dykeman and J. W. Wong, "A performance study of broadcast information delivery systems," in *Proc. IEEE Infocom '88*, (New Orleans, LA), pp. 739-745, 1988.



John W. Wong received the B.S. degree in engineering and the M.S. and Ph.D. degrees in computer science from the University of California at Los Angeles, in 1970, 1971, and 1975 respectively.

Since 1975, he has been with the University of Waterloo, Canada, where he is currently a Professor of Computer Science and Electrical Engineering. From September 1981 to August 1982, he was a Visiting Scientist at the IBM Zurich Research Laboratory, working on performance analysis of local area networks. His research interests include computer networks, communication protocols, and performance evaluation.

Dr. Wong is a member of the Editorial Board of the *Performance Evaluation* journal. He has served on program committees of several conferences on computer communications; he was Program Chairman of the IEEE INFOCOM '84 Conference.