Polling Systems: Applications, Modeling, and Optimization

HANOCH LEVY, MEMBER, IEEE, AND MOSHE SIDI, SENIOR MEMBER, IEEE

(Invited Paper)

Abstract—**Polling models have been extensively studied in the last two decades. As a result, a large body of knowledge has been built in this area and they became a powerful tool for the performance analysis of a wide variety of important applications.**

The goal of this paper is to expose the application oriented reader to the modeling and analysis capabilities of polling systems. We review the state-of-the-art in this field, focusing on applications and their representation by polling models, and on their analysis and optimization issues. Examples include token rings, ARQ and time-sharing schemes, random-access protocols, robotics, and manufacturing systems, etc. The emphasis is not on the analytical derivations but rather on the description of the capabilities and limitations of the different polling models.

I. Introduction

POLLING systems were first introduced in the early 1970's when the *cyclic polling system* was used to model time-sharing computer systems. This triggered extensive research efforts which continued over the last two decades. During this period numerous researchers from various areas, mainly engineering, discovered that polling systems are powerful tools in modeling a wide variety of applications. As a result of the flourishing research, a rich source of knowledge and understanding emerged in this area, computational techniques have been improved and new analytical approaches have been developed, increasing the attractiveness and the usefulness of the model. In addition, the original models have been extended, and their modeling power has been increased.

Today, the field of polling systems offers both practitioners and researchers a large set of tools suitable for modeling and analyzing a wide variety of interesting and important applications. The common denominator of all polling systems is that they consist of a *single* resource of service *shared* by multiple queues. The range of applications in which polling models can be used is very broad. They include computer communications, robotics, traffic and transportation, manufacturing, production, mail distribution, etc.

The goal of this paper is to survey the field of polling systems, putting the emphasis on applications and keeping in mind the application oriented user. The objective is to draw the attention of the reader to the capabilities as well as to the limitations of polling models in representing various applications. Our approach is to review the different applications and to examine how they can be represented and analyzed via known models and techniques in the area of polling systems.

Among polling models the most common one is the cyclic polling model that was first used in the analysis of time-sharing computer systems. This model regained much attention in the early 1980's in the performance analysis of token passing systems such as token ring and token bus and other demand-based channel-access schemes in local area networks. This model, which we call the *basic model*, and some of its applications are described in Section II. In recent years the basic model has been extended and enhanced in various directions allowing the analysis of relatively complex and new applications. One enhancement allows the routing of customers between the queues within the system. Customer routing allows to represent a variety of features which cannot be represented by the basic model. Examples include the modeling of acknowledgments and distributed algorithms in token ring networks and complicated robotics and manufacturing systems. This extension and its applications are described in Section III. The replacement of a *fixed polling order* of visits by a *random polling order* allows to model distributed control systems in which stochastic algorithms are used to determine which station will be served next. Packet-ratio networks serve as a good example for an application of random polling and this is described in Section IV. Another set of extensions eliminates several independence assumptions used in the basic model. Examples of applications in which such extensions are required are token ring networks with token-acquisition overhead, disc systems with spread data, etc. These extensions and their applications are described in Section V.

In addition to the modeling aspects of polling systems we are interested in the aspects of performance improvement and system optimization issues. Traditionally, these are the ultimate goals of any modeling and analysis effort. The basic polling system leaves little room for effective performance improvement or system optimization. In recent years several approaches which enhance the optimization capabilities of polling systems have been introduced. Such approaches include polling according to a prespecified order, servicing according to the Bernoulli or the binomial service disciplines and semidynamic polling orders, and they are described in Section VI. "Pseudo" conservation laws for polling systems, which have recently been introduced, play an important role in both analysis and optimization, and they are described in Section VI as well.

The performance measure that is of great importance in most systems is the *response time* of the system. In computer and communication systems where the units that get service are *messages*, this corresponds to the message delay time—the time elapsed from the instant the message enters the system until it is delivered to its destination. In particular, we will be interested in the *mean delay* observed in the various queues of the system. Another performance measure that is of interest is the *amount of work* in the system—the amount of time the server (or servers) will have to invest in order to clear the system. A performance measure often used in the analysis of polling systems is the *cycle time*—the time between two successive visits of the same queue. In the sequel, we will mainly focus on these performance measures, in particular on their mean values.

Last, a word about related literature. Several tutorial works on polling (Watson [73], Takagi [66] and Takagi [68]) have been published in recent years. The goal of this paper is not to provide an alternative to those works, but rather a complement. The emphasis in those works is on detailed and comprehensive description of the analysis methodology and on providing a thorough literature review, and thus they appeal mainly to researchers. In contrast, as stated above, our emphasis is on discussing all aspects related to applications, namely modeling, optimization issues and practical considera-

Paper approved by the Editor-in-Chief of the IEEE Communications Society. Manuscript received May 11, 1989; revised November 15, 1989.

H. Levy is with the Department of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel.

M. Sidi is with the Department of Electrical Engineering, Technion—Israel Institute of Technology, Haifa 32000, Israel.

IEEE Log Number 9038363.

0090-6778/90/1100-1750$01.00

tions. For completeness, we summarize very briefly the analytic methodologies used in polling systems in Sections II-C and -D. For a detailed description of these methodologies the reader should refer to Takagi [66] and for a thorough reference list to Takagi [68].

II. The "Traditional" Cyclic Polling Model

A. The Time Sharing System: A Single Computer and N Terminals

Time-sharing computer systems triggered the research in the area of polling systems at the early 1970's. A time-sharing computer system consists of N terminals connected by multidrop lines to a central computer. The data transfer from the terminals to the computer (and back) is controlled via a "polling scheme" in which the computer "polls"[1] the terminals, requesting their data, one terminal at a time. The order in which the computer polls the terminals is usually cyclic. This system and many others have been modeled and analyzed by the "traditional" polling system, which has been extensively studied in the last two decades. This model is described below.

B. The "Traditional" Cyclic Polling System

The "traditional" polling system, to be called in the sequel *the basic model*, consists of N infinite size[2] queues and a single server which serves them one at a time. The model has been treated both in a continuous-time framework and in a discrete-time framework which are equivalent in most aspects; our discussion below focuses mainly on the continuous time framework. The arrival process to queue i is assumed to be an *independent Poisson stream* with rate λ_i. The customers arriving to queue i are called type-i customers, and are assumed to have service time B_i which is a random variable whose Laplace–Stieltjes transform (LST), first moment and second moment are given by $B_i^*(s)$, b_i, and $b_i^{(2)}$, respectively. The actual service time of a specific customer is assumed to be independent of the other system variables. After being served at queue i, a type-i customer is assumed to leave the system.

The server *visits* (or, *polls*) the queues in a *cyclic* order (namely, in the order $1, 2, \cdots, N, 1, 2, \cdots$) and for convenience of notation all references to visit index are done modulo N. After completing a visit to queue i, the server incurs a switch-over period (walking time) whose duration is S_i (a random variable) with LST, mean and second moment $S_i^*(s)$, s_i, and $s_i^{(2)}$, respectively; we let $s = \sum_{i=1}^{N} s_i$. The period during which the server continuously serves queue i is called a *service period* of queue i and the succeeding period is called a *switchover period* of queue i. In the text we sometimes distinguish between systems with zero switch-over periods and systems with nonzero switchover periods; in the former *all* switchover periods are deterministically of zero length, while in the latter they are not. A system in which all switchover periods are deterministically of zero length is called a system with zero switch over periods. The model is schematically depicted in Fig. 1.

This system models the time-sharing computer system as follows. The server represents the central computer, the queues represent the terminals and the customers represent the messages (or data units) transferred between the terminals and the computer.

Several service policies have been examined and studied in the context of polling systems. Among them the most common ones are the *exhaustive*, *gated*, and *limited-1* policies. In the exhaustive policy the server serves queue i until the queue becomes completely empty. In the gated policy the server serves in a given service period all the customers which were present at the queue when the queue was polled (polling instant). In a limited-1 policy at most one customer is served in each visit to the queue.

[1]This is probably the source for the term *polling systems*.

[2]An alternative model is one in which the queue size is limited; a special case is a model with a single buffer in each queue. The latter is used to represent applications called clearing systems in which the queue may contain a large number of buffers but the service time of a queue does not depend on the number of customers in the queue (see Takine, Takahashi, and Hasegawa [70], [71]).

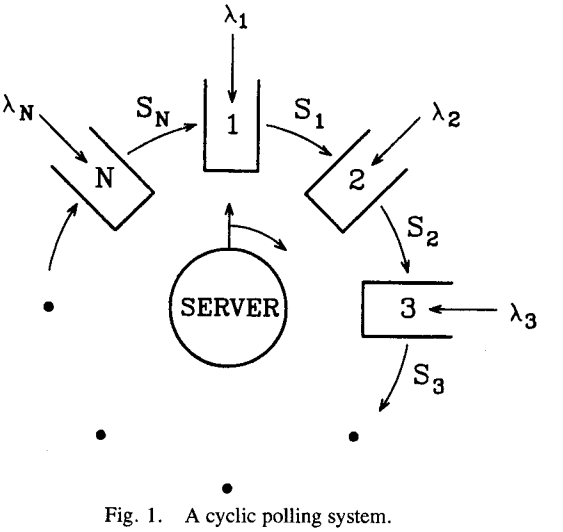


Fig. 1. A cyclic polling system.

C. Analysis

From the analysis aspect, polling systems with limited-1 service are inherently different from those with exhaustive or gated service. We, therefore, divide our discussion into two parts. The first is devoted to exhaustive and gated polling systems and the second to limited-1 polling systems. Note that some of the analysis of polling systems has used results from the analysis of single server queueing models with vacations. We do not elaborate on this subject and the reader is referred to the survey in Doshi [23].

1) Exhaustive and Gated Polling Systems: Cyclic polling systems with either the exhaustive or gated service policies are stable as long as $\rho \triangleq \sum_{i=1}^{N} \rho_i \triangleq \sum_{i=1}^{N} \lambda_i b_i < 1$. Note that this condition does not depend on the switchover periods. The reason is that once the server arrives to a queue it is dedicated to serve all the work accumulated in the queue without any interruption. Thus, when the system becomes heavily loaded, the wasted time during switchover periods is negligible compared to the time the server is busy in service.

Several methods have been developed for computing the mean delay in the various queues in exhaustive and gated polling systems. All these methods require the solution of sets of linear equations. The first method is the *buffer occupancy method* which was used by many researchers (Cooper and Murray [22] and Cooper [21] for systems with zero switch-over periods, Eisenberg [25], Hashida [31], Konheim and Meister [41], and Rubin and DeMoraes [56] for systems with nonzero switchover periods). In this method the mean waiting times are computed using the set of variables $\{X_i^j\}$, $(1 \le i, j \le N)$, in which X_i^j represents the number of customers present at queue j when queue i is polled. We call these variables the *buffer occupancy variables*. Closed-form expressions are known for the first moments of the buffer occupancy variables ($E[X_i^j]$). The cross correlations of these variables, $E[X_i^j X_i^k]$, are obtained by solving a set of N^3 linear equations with N^3 unknowns[3] (see a summary of these equations in the Appendix). It is recommended to solve this set of equations iteratively, and it is known (Levy [47]) that this procedure converges and it requires $O(N^3 \log_\rho \epsilon)$ computation steps (where ρ is the system utilization and ϵ is the accuracy required). Once the values of $E[X_i^j X_i^k]$ are known, a closed-form expression for the mean delay in each queue is readily available (see the Appendix). The buffer occupancy method was the most commonly used method until more sophisticated techniques have been developed. The mean delay analysis of most variations of polling systems is available via this technique.

Another method that has been developed a few years later is the *station-time* method presented in Humblet [33] and Ferguson and

[3]Note that for systems with zero switchover periods the sets can be reduced (Cooper and Murray [22] and Cooper [21]) to contain only N^2 variables.

Aminetzah [27] (a different variation of the method is given in Carsten, Newhall, and Posner [18]). In this method the mean delay values are computed using the *station-time variables*. The station time variable, U_j ($1 \le j \le N$), is the sum of two variables: V_j—the duration of a visit to queue j, and S_{j-1}—the duration of the preceding switchover period. Similarly to the case of the buffer occupancy method, closed-form expressions are known for $E[U_j]$. The cross correlations of these variables, $E[U_jU_i]$ (where U_j and U_i are the station times of stations j and i, respectively, and station j is visited after station i), are obtained by solving a set of N^2 linear equations with N^2 unknowns (see a summary of these equations in the Appendix). Once the values of $E[U_jU_i]$ are known, a closed-form expression for the mean delay in each queue is again readily available. The structure of these equations is simpler than that of the buffer occupancy equations and thus the iterative solution version of the corresponding set of equations seems to be slightly more efficient. The method has been applied to many variations of polling systems.

The third method is a very efficient iterative variation of the buffer occupancy method. The method has been developed by Swartz [64] and requires $O(N \log_\rho \epsilon)$ computations to solve for the mean delay of a single station. Unfortunately, the method has been derived only for the discrete-time exhaustive polling system.

The fourth method has recently been developed by Sarkar and Zangwill [57] and it is an efficient variation of the station-time approach, which applies to many variations of polling systems. The method requires the solution of only N linear equations to derive the mean delays in all N stations. Nevertheless, it seems that since this set of equations is very dense, the equations cannot benefit from the use of iterative approaches for their solution and thus the method requires $O(N^3)$ computations.

From a practical point of view, the above methods allow the exact computation of the mean delays for most reasonable size systems. With the slower methods one can easily handle systems consisting of 30–40 stations on a minicomputer. With the faster methods, even systems consisting of 500 stations can be easily handled. For a more detailed discussion of these methods and their merits see Levy [47] and Levy [48].

2) Limited-1 Polling Systems: Similarly to the exhaustive and gated systems, cyclic polling systems with limited-1 service require that $\rho < 1$ for their stability. However, this condition is not sufficient. Since in each cycle at most one customer is served in each of the queues, it is also required that the expected number of customers arriving at queue i during a cycle is smaller than one (this guarantees that each queue will be stable). This condition is translated to $\lambda_i s/(1-\rho) < 1$, since $s/(1-\rho)$ is the expected length of a cycle when the system is stable (see Kuehn [43], Servi [58], [59]).

In contrast to the exhaustive and gated polling systems the limited-1 polling systems do not lend themselves to a simple delay analysis. These systems seem to be inherently difficult to analyze and thus exact solutions are not available for general cases. Exact solutions do exist for the following two types of special configurations: 1) systems consisting of two queues with general parameters; 2) fully symmetric systems with N queues (in which the parameters of all staions are identical). The analysis of the first type involves translation of the problem into a two-dimensional boundary value problem of mathematics and physics, like a Riemann-Hilbert problem. The analysis can be found in Eisenberg [24], Cohen and Boxma [19], Boxma [7], and others. The analysis of the second type of systems leads to a closed form expression for the mean waiting time which is given in the Appendix (for details refer to Nomura and Tsukamoto [55], Fuhrmann [28], and Takagi [65]). Solutions that are based on numerical calculations and can achieve high accuracy require massive computations (exponential in the number of queues), and thus are limited to systems with a small number of queues, about 5–10. Such solutions are available in Blanc [6] and Leung [45].

Unfortunately, the problem of computing exact mean delays in general limited-1 systems seems to be unsolvable. As a result, the tools available to the researcher or the application user are either to use simulations or to develop approximations for general configurations. Several such approximations have been suggested in the last decade, most of them sharing the following properties.

1) They consist of a few equations and mathematical expressions. Thus, their evaluation does not require heavy computations (in contrast to the exhaustive and gated systems).

2) The expressions provided are exact for some special cases (e.g., fully symmetric systems).

3) Their accuracy is quite good either when the total system utilization is relatively low ($\rho > 0.5$) or when the system is relatively symmetric (i.e., the parameters of the different stations are relatively close to each other). By good accuracy we mean that the relative error in predicting the mean delay of the specific stations is up to 5–10%.

4) For heavily loaded ($\rho > 0.8$) and relatively unbalanced systems (e.g., $\rho_1 = \rho_2 = \cdots = \rho_{10} = 0.03$ and $\rho_{11} = 0.5$ in an eleven station system) the approximations may be very inaccurate and can reach relative errors of 50% and up compared to simulations.

The approximate solutions available today are given in Kuehn [43], Boxma and Meister [12], Fuhrmann and Wang [30], Groenendijk [74], Srinivasan [63] and Ibe and Cheng [34]. Recent approximate solution methods are based on the "pseudo" conservation law derived by Boxma and Groenendijk [9] (see description in Section VI). It is hard to come up with definite statements regarding the relative quality of the different methods since it depends on the specific parameters of the system. For detailed numerical comparisons of the various approximations the reader is referred to the references mentioned above.

D. Comparison of the Service Disciplines

The various service disciplines differ in their characteristics. From a global point of view (namely, considering the system as a whole) the exhaustive policy is considered more "efficient" than the gated policy, which is more "efficient" than the limited-1 policy. In fact, if the criterion of efficiency is the weighted sum of the mean waiting times (where the weights are the variables $\{\rho_i\}$), then the policies are ranked according to this order (this is a direct result of the conservation law, described in Section VI). This also implies that in fully symmetric systems this ranking holds for the mean waiting time as well. Moreover, if the efficiency criterion is the amount of unfinished work in the system, it can be shown that the exhaustive system stochastically dominates the gated system which stochastically dominates the limited-1 system (see Levy, Sidi, and Boxma [52]). On the other hand, if the performance criterion is "fairness," it is commonly believed that the policies should be ranked in reverse order. Note however that "fairness" has not been well defined (and therefore not analyzed), so this belief should be considered accordingly. The reasons for this belief are twofold. 1) In the exhaustive policy (and somewhat in the gated policy) the heavier stations receive better attention by the server and thus their customers suffer less delay than those of the light stations. 2) The exhaustive policy (and the gated policy) tends to serve many of the customers in what amounts to a Last Come-First Served order.

E. Applications

The basic polling model has been used in numerous applications. In the following we review a few of these applications.

1) Token Ring Networks: A ring network can be characterized as a sequence of point-to-point links between stations, closed on itself. All messages travel over a fixed route from station to station around the loop. In token ring networks the access to the ring for transmissions is controlled by a *token*, which is a dedicated bit structure that can be in one of two possible states: occupied or free. When the ring is first activated, a free token circulates around the ring from station to station. A station with data to transmit reads the free token and changes it to the occupied state before retransmitting it. The occupied token is then incorporated as part of the header of data transmitted on the ring by the station. Thus, other stations on

the ring can read the header, note the occupied token and refrain from transmitting. In most applications, the station that changed the token to occupied will change the token to free when it decides to transfer the right for transmission to another station (see Bux [15]).

The modeling of a token ring network by the basic polling system is very natural. The stations of the ring are the stations of the polling model. The server is the channel on which data are transmitted, and the polling mechanism is achieved by the token that cyclically transfers the use of the channel. The switchover time in the polling system corresponds to the time interval that begins at the instant a station finishes its transmission of data and releases a free token, and concludes when the next station receives the free token.

2) Robotics Systems: Consider a robotics system in which a single robot accepts as input N types of parts arriving in N different streams. Each type requires different processing, which is conducted by a different tool. When switching from the processing of one part to another, the robot incurs a switchover period required for changing the tools.

The modeling of this system by the basic polling system is as follows. The robot is modeled by the server, the N types of parts are modeled by the N types of customers, being queued in the N queues, and the time for changing tools is modeled by the switchover periods.

3) Various Nongeneric Computer and Communication Systems: In many nongeneric computer and communication systems (see Kruskal [42]), one often encounters the situation in which a single processor serves N different types of jobs. A common practice is to accumulate the jobs of each type separately, and to process the jobs, one type at a time. It is also common to use a fixed order of service (between the different types), in particular, the cyclic order. Obviously, a cyclic polling model will fit such systems in which jobs are represented by the customers and the processor is represented by the server.

III. Enhancing Polling Systems by Customer Routing

A. Motivation

While the basic polling model described in Section II is suitable for modeling a large variety of systems, it may fail to capture several important features of certain applications. A major limitation of the basic model is that the customers of each queue are assumed to leave the system upon service completion. Consequently, simple system features such as the generation of a job at one queue upon the completion of a job in another queue cannot be modeled. In many applications, which we describe in Section III-C, these features are very important. Examples include the modeling of acknowledgments and distributed algorithms in token ring networks and the modeling of robotics systems with production stages.

B. Routing in Polling Systems

The basic polling model can be extended in a framework similar to that used in the analysis of queueing networks. The generalization is done by extending the assumptions about the fate of the customers upon service completion. Rather than assuming that a type-i customer leaves the system when its service is completed, one can assume that upon service completion the customer is routed to queue j $(1 \le j \le N)$ with probability $r_{i,j}$; with probability $r_{i,0}$ the customer leaves the system. Once routed to queue j, the customer now behaves as a type-j customer, and waits for its service at queue j. We call this system a *polling system with probabilistic routing*.

As in the case of the basic polling system, an exact analysis of this system exists for the exhaustive and the gated service policies (see Sidi and Levy [60]). The stability condition of this system is derived by considering the *equivalent service time* of a type-i customer; this is the total amount of service to be given to a type-i customer from its entrance to queue i until its departure from the system. Denoting by $\tilde{b}_i$ the mean value of the equivalent service time of a type-i customer, the stability condition is

$$\tilde{\rho} \triangleq \sum_{i=1}^{N} \lambda_i \tilde{b}_i < 1.$$

Note that $\tilde{b}_i$ can be derived from the system parameters by solving the simple set of N linear equations:

$$\tilde{b}_i = b_i + \sum_{j=1}^{N} r_{i,j} \tilde{b}_j \quad 1 \le i \le N.$$

The mean delay analysis follows the approach described in Section II, and requires the use of the buffer occupancy method (rather than that of the station-time method). The basic variables used in this analysis are the set $\{X_i^j\}(1 \le i, j \le N)$ which is the number of customers present at queue j when queue i is polled (at equilibrium). As in the basic system, the key for the mean delay analysis is the derivation of $E[X_i^j]$ and $E[X_i^j X_i^k]$ $(1 \le i, j, k \le N)$. These values can be calculated by solving a linear equation set consisting of N^3 variables whose solution (in an iterative approach) requires $O(N^3 \log_{\tilde{\rho}} \epsilon)$ steps where $\tilde{\rho}$ is the utilization defined above and ϵ is the required accuracy. From these values the mean delay observed by the customers of queue i $(1 \le i \le N)$ is readily available.

Unlike in common queueing networks (like Jackson-type networks) the delay of a customer at queue i is *not independent* of the customer source (namely, of the place from which the customer arrived to queue i). Using additional computation which requires $O(N^3)$ computational steps, one can compute the mean values of *specific waiting times*, $E[W_{i,j}]$, which is the mean delay suffered at queue j by customers who arrived (to queue j) from queue i. Another important measure, whose computation is more complex, is the mean time in system (*system sojourn time*) of a customer which follows a specific path. The detailed analysis of this system is found in Sidi and Levy [60].

C. Customer Branching

A generalization of the model described above is a system with customer branching. In this model, the departure of a customer from queue i may trigger many concurrent arrivals to the system. Formally, we say that the number of customers arriving to the system at such an epoch is given by a vector $L_i = (L_{i,1}, L_{i,2}, \cdots, L_{i,N})$ in which $L_{i,j}$ is a random variable indicating the number of customers arriving to queue j as a result of a departure from queue i. If $L_{i,j} = 0$ for $1 \le j \le N$, then no branching occurs and the customer that completed its service at queue i leaves the system. The system with routing is obviously a special case of this system in which L_i takes the value e_j (e_j is a vector whose jth component equals 1 and whose other components equal 0) with probability $r_{i,j}$ $(1 \le i, j \le N)$.

The analysis of systems with customer branching is not provided in the literature, but can be carried out along the same lines as the analysis of the system with routing.

D. Modeling Via Customer Routing and Branching

Below we describe several applications which cannot be represented by the basic polling model, and demonstrate how the feature of customer routing can be used to model them.

1) Modeling Selective-Repeat ARQ in a Token Ring

The token ring network allows the transmission of packets from one station to another in a conflict-free manner, since at any instant only a single station (the one that holds the token) is transmitting. Yet, transmissions of packets may still fail due to errors and distortions on the ring itself. Typically, such errors are rare and a selective-repeat ARQ (SR-ARQ) scheme would be used to recover these errors. In an SR-ARQ scheme, a station that receives an erroneous message transmits a negative acknowledgment to the transmitting station to indicate that the message has to be retransmitted.

A token ring network is depicted in Fig. 2(a). To study the performance of a network employing an SR-ARQ we model the network by a polling system [Fig. 2(b)] in which each station is represented by two queues, one for messages which need to be sent out and one for negative acknowledgments to be sent back when erroneous messages are received. We name the queue of data messages at station i by msg (i) [message queue], and the queue for

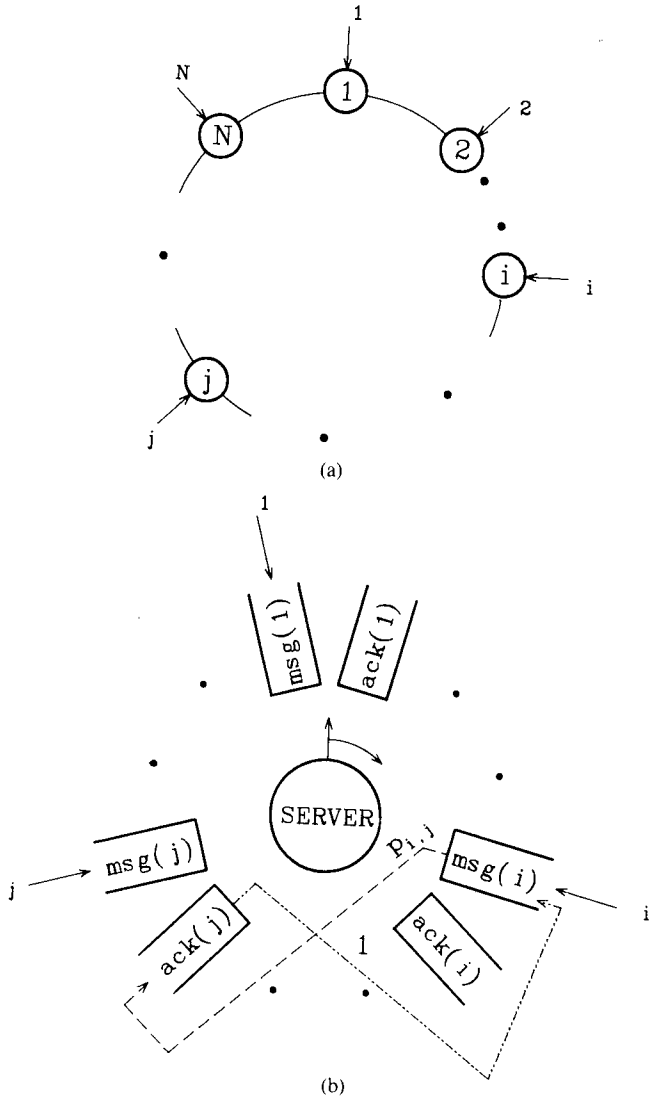


Fig. 2. (a) A token ring. (b) The queueing model.

negative acknowledgments at queue i by ack (i) [acknowledgment queue].

The parameters of the polling model are derived from the token ring parameters. The arrival rates of new messages into the corresponding queues are $\lambda_{\text{msg}(i)} = \lambda_i$ and $\lambda_{\text{ack}(i)} = 0$ for $i = 1, 2, \cdots, N$. The reason for $\lambda_{\text{ack}(i)} = 0$ is obvious; no negative acknowledgments arrive at a station from outside the network. Negative acknowledgments are generated as a response of receiving an erroneous message. The service times of messages are $B_{\text{msg}(i)} = B_i$ for data messages and $B_{\text{ack}(i)} = B$ for negative acknowledgments messages. Here it is assumed that the transmission time of all acknowledgment messages is the same. Consider a station, say i, that transmits its data messages to station j (and is the only station transmitting to j) and assume that the probability that a transmitted message will arrive at j in error is $p_{i,j}$. In the polling model this corresponds to the routing variables $r_{\text{msg}(i),\text{ack}(j)} = p_{i,j}$ (in practice, this parameter will be identical for all i and j) and $r_{\text{ack}(j),\text{msg}(i)} = 1$. This message routing is depicted in Fig. 2(b). The reason is that whenever a data message is transmitted by station i to station j, with probability $p_{i,j}$ a negative acknowledgment message will be generated at ack (j), and when this message is transmitted from station j to station i then a message is generated at station i, representing the message that has to be transmitted to station j. Thus, using the feature of customer routing, it is possible to study the performance of an SR-ARQ in token ring networks. Note that if more than one station is transmitting to station j, say, we need different ack queues at j, each corresponds to a different transmitting station.

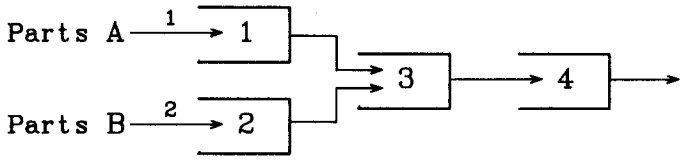


Fig. 3. Flow of parts.

2) Distributed Algorithms on Token Ring Networks: The token-ring is a distributed control system. As such, it requires the use of certain distributed algorithms to control its operation. Examples include algorithms for the selection of one active station as a designated station (leader election) having responsibility for token regeneration (when the token gets lost), algorithms for the allocation of shared resources (e.g., shared printer) and others.

Important features of such algorithms are that the events of packet creation are strongly correlated to each other. In fact, the common property of distributed algorithms is that when a packet is received by its destination, it triggers the destination to generate a response packet; moreover, in many instances packet reception will trigger the generation of several packets to be sent to various destinations.

The modeling of a distributed algorithm depends on the algorithm itself and may require the feature of branching. For example, if a message from station i is to be received by all other stations and requires them to respond, then L_i in this case is equal to the vector $(1, \cdots, 1) - e_i$.

3) Robotics Systems with Stages: Consider a robotics production system as described in Section II-E2). Now, relax the assumption that all work requests arrive from outside of the system. Rather, some of the requests are produced by the system itself. For example, parts processed in one stage will be connected to other parts in a later stage. Therefore, this is a system in which work done in some of the stages can produce work requirements for other stages. An important performance question, related to this system, is how to order the different stages and robot route, so as to improve its performance.

As a simplistic example consider a robotics system that accepts as input two streams of parts (denoted A and B) and needs to drill in each of them three holes. The holes to be drilled in Part A are of the types 1, 3, and 4 (according to this order) and the holes to be drilled in Part B are of the types 2, 3, and 4 (according to this order). Whenever a new type of hole needs to be drilled, the drill needs to be adjusted appropriately; This operation requires some set-up time. The system is organized in 4 stations indexed 1, 2, 3, and 4, and hole type i is drilled at station i. The flow of parts is depicted in Fig. 3. The robot visits the station according to a predetermined order and processes the parts there according to either the exhaustive policy or the gated policy. When switching from one station to another the robot incurs a switchover period due to the need to readjust the drill. We assume that type A parts and type B parts arrive according to a Poisson process at station 1 and station 2, respectively; both arrival rates are 0.1. The times required for drilling a hole are deterministic and are 3.0, 0.5, 1.0, and 1.0, for the types 1, 2, 3, and 4, respectively. The time for readjusting the drill is deterministic and requires one unit of time.

We model this system using a polling system with routing, consisting of 4 queues, representing the four stations and having the same index. The system parameters are therefore as follows: $\lambda_1 = \lambda_2 = 0.1$, $\lambda_3 = \lambda_4 = 0$, $b_1 = 3$, $b_2 = 0.5$, $b_3 = b_4 = 1$, $b_1^{(2)} = 9$, $b_2^{(2)} = 0.25$, $b_3^{(2)} = b_4^{(2)} = 1$, $s_i = s_i^{(2)} = 1$ (for every i). The nonzero entries in the transition matrix are $r_{1,3} = 1$, $r_{2,3} = 1$, and $r_{3,4} = 1$. Note that when station 4 is eliminated, the resulting system is the one considered by Katayama [35].

We now evaluate the system performance for the gated service policy, as a function of the cycle order used by the server. We consider three orders: (i) 1, 2, 3, 4, (ii) 2, 1, 3, 4, and (iii) 4, 3, 1, 2. In Table I we present the mean sojourn time in each of the queues

TABLE I
SOJOURN TIMES AS A FUNCTION OF THE CYCLE ORDER

Order	Station 1 Soj. Time	Station 2 Soj. Time	Station 3 Soj. Time	Station 4 Soj. Time	Network Soj. Time
1,2,3,4	18.69	12.70	7.74	6.98	30.41
2,1,3,4	18.17	12.77	10.47	6.72	32.66
4,3,1,2	17.35	11.73	13.02	16.96	44.52

and the mean sojourn time in the system (averaged over all customers) as a function of the service order. From the table we may learn that the performance of a system with routing is quite sensitive to the visit order. In particular, the performance of the third order is very poor; this results from visiting the stations in a direction opposite to the flow of customers. There are also differences between the first two cases (though not as significant) which result from the different characteristics of stations 1 and 2.

IV. RANDOM POLLING

A. Motivation and Modeling

In most models of polling systems the order in which the server visits the stations is *fixed* and *periodic*. As we shall see, in some applications this order is *random* and is determined by a chance mechanism. The polling model that corresponds to such systems is called *random polling* and its basic feature is that the station polled after station i is determined according to a rule that involves some randomization. In general, the probability that station j will be visited after station i may depend on both i and j and also on part of (or all) the history of previous visits at the stations since the system started to operate.

The main applications of random polling models are in *distributed control* systems. In such systems there is no central controller that can poll the stations in a fixed order. Rather, the decision on the next station to be visited is achieved in a distributed manner, by cooperation among the stations. An example of such a system is a shared communication channel, such as a shared bus or a shared radio link, in which the decision on who will transmit next is based on a random-access algorithm.

The model of a random-polling system is the same as that of the basic polling system except for the order in which the server visits the queues. A simple model for the visit mechanism is that the next station polled is determined according to an irreducible positive recurrent discrete-time Markov chain with homogeneous one-step transition probabilities. According to this model the probability that station j will be visited after station i is $P_{i,j}$ with $\sum_{j=1}^{N} P_{i,j} = 1$. An even simpler model is that the next station to be visited does not depend on the previously visited station, i.e., $P_{i,j} = P_j$ for all i. The analysis that can be carried out for both models is similar to that of the basic systems. This means that the exact analysis of arbitrary systems with either the exhaustive or the gated policies can be done by numerical computations, while the analysis of the limited-1 policy is restricted to fully symmetric systems (exact) or to approximations (general systems). For instance, the analysis of the simpler model via the buffer occupancy method appears in Kleinrock and Levy [38] and a conservation law for these systems appears in Boxma and Weststrate [13]. It is noteworthy mentioning that the buffer occupancy equation set of this system (which needs to be solved for deriving the mean delay figures) is smaller than that of the basic system (only N^2 equations versus N^3 equations) but its solution requires more computation.

B. Applications: Random-Access Schemes in Shared Channel Networks

Random-access techniques are designed for coordinating the access of spatially distributed stations to a shared channel. The ALOHA network (Abramson [1]) is a well-known packet radio network which consists of stations that share a radio channel. The *pure* ALOHA access scheme is very simple. As soon as a packet arrives at a station, it is transmitted. In the case that the transmission fails (since other packets were transmitted on the channel at the same time, causing a *collision*), the packet is scheduled for retransmission after a random period of time. The *slotted* ALOHA access scheme is similar, except that the time axis is divided into slots (that correspond to the time required for transmitting a packet) and a station can start transmission of a packet only at slot boundaries. In both the pure and the slotted ALOHA schemes the stations contend for the shared resource (the channel) for each transmitted packet.

An alternative for the above schemes is the *reservation* ALOHA access scheme (Lam [44]). In this scheme, a station whose packet is successfully transmitted at slot t, may add a bit into its successfully transmitted packet to indicate that it has a packet ready to be transmitted in slot $t + 1$. In this case, the station is granted the exclusive right to transmit in slot $t + 1$, and this transmission will not be interfered by any other station. By this mechanism, reservation ALOHA overcomes the main deficiency of the slotted ALOHA scheme—the continuous contention among the stations for every transmitted packet. The reservation scheme can be either exhaustive or gated. Exhaustive reservation means that a station that is currently transmitting will continue to reserve the channel until its buffer empties. Gated reservation means that a station that is currently transmitting will continue to reserve the channel only until it transmits all packets that it had when it seized the channel.

When a transmitting station no longer reserves the channel for the next slot, part or all stations of the system start contending in order to seize the channel. The contention may follow various algorithms such as the ALOHA scheme, a collision resolution scheme (Capetanakis [17]), etc. The length of the contention period is random and the next station that will seize the channel is also random.

The reservation ALOHA scheme can be modeled by random polling as follows. The server represents the channel, the service of a customer represents the transmission of a packet, the switchover period represents the contention period, and the station to be visited next is determined according to the contention protocol. When all stations attempt to seize the channel during the contention period and use the same contention algorithm in each period, the system behaves exactly as the simple random-polling model. For instance, if station j attempts to seize the channel with probability q_j in each slot of the contention period, then

$$P_j = \frac{q_j \prod_{i \neq j} (1 - q_i)}{\sum_{l=1}^{N} q_l \prod_{i \neq l} (1 - q_i)} \qquad 1 \leq j \leq N.$$

The length of the switchover period (the contention period) depends on which station seizes the channel at the end of that period. Specifically, the probability that the switchover period will last l slots and station j seizes the channel at the end of the period is $(1 - \sum_{l=1}^{N} q_l \Pi_{i \neq l}(1 - q_i)]^{l-1} \sum_{j=1}^{N} q_j \Pi_{i \neq j}(1 - q_i)$. Accordingly, the duration of the switchover period when switching into queue j, is shifted geometric with parameter $\sum_{l=1}^{N} q_l \Pi_{i \neq l}(1 - q_i)$.

The case in which only stations having packets to transmit are contending to seize the channel is more interesting but, unfortunately, does not fall in the framework described above. The reason is that in this case the duration of a switchover period depends on the state of the system (which of the stations have packets to transmit and which do not) at the beginning of the period. Such models do not lend themselves to exact analysis and one must resort to either simulations or approximations. In certain cases the approximation can be carried out via the random polling system (see Levy [46]).

V. GIVING UP INDEPENDENCE

A. Motivation

Most models of polling systems assume complete independence among the various processes within the systems, such as the arrival processes, the service times, the switchover times, etc. Some of the independence assumptions are essential to allow the exact analysis

of the system performance. Other assumptions are used just to simplify the analysis, but similar analysis can be carried out even when these assumptions are relaxed. From a modeling point of view, giving up independence is extremely important and useful since many of the underlying processes in real applications are naturally dependent. Examples include systems with simultaneous arrivals, systems with correlated switchover times, etc. Some of the models that allow dependence to some extent are discussed in this section along with their applications.

B. Simultaneous Arrivals

The arrival processes to the different queues of a queueing system are not necessarily independent. For instance, let us examine a communication system that can be modeled as a cyclic polling system. Consider a switching node with n input queues, N output channels and a single switch that connects queue i to channel i according to some polling policy. In such a system, when a message that originates at the node has to be broadcast through a subset of the channels, its generation corresponds to an arrival of a copy of that message to each of the corresponding queues at the same time. Therefore, the arrivals to the different queues are not independent of each other in this system.

We model simultaneous arrivals as follows. Along the time axis there are *arrival epochs*. The distribution of the arrival epochs is Poisson with parameter λ, namely, the time between two successive arrival points is exponentially distributed with mean $1/\lambda$ and is independent of any other event in the system. In each arrival epoch, bulks of customers arrive to the different queues according to some probability distribution. Specifically, let $\boldsymbol{K} = (K_1, K_2, \cdots, K_N)$ be a random vector in which component K_i represents the number of customers arriving to queue i at an arrival epoch. The vector $\boldsymbol{K}$ is assumed to have the same joint distribution at each arrival epoch and this distribution is independent of previous or future arrival epochs. The joint probability distribution of the vector $\boldsymbol{K}$, $(\text{Prob}[K_1 = i_1, \cdots, K_N = i_N],\ i_j \geq 0,\ 1 \leq j \leq N)$, is arbitrary (with the constraints $K_i \geq 0$ for every i, and $\boldsymbol{K}$ is not the zero vector). Thus, we have a rather general structure of correlation between the arrivals to different queues. The special case of independent bulk arrivals is represented by a distribution in which the only nonzero probabilities are of the form $\text{Prob}[K_1 = 0, \cdots, K_l = i_l, \cdots, K_N = 0]$ for some l $(i_l \geq 1)$. The case of single independent arrivals is represented by a distribution in which the only nonzero probabilities are of the form $\text{Prob}[K_1 = 0, \cdots, K_l = 1, \cdots, K_N = 0]$ for some l.

As in the case of the basic polling system, an exact analysis of this system exists for the exhaustive and the gated service policies. The analysis uses the buffer occupancy method and the details appear in Levy and Sidi [51].

C. Switchover Times

In most applications the switchover times in polling systems correspond to the overheads incurred when service is switched from one station to another. It is usually assumed that these switchover times are independent of each other and also independent of the system state. Yet, in many applications the switchover times may depend either on the state of the station just visited or on the state of the station to be visited.

For instance, consider a system in which switchover periods may be of two types, short and long, and the type of the switchover period depends in some manner on the type of the previous switchover period. In such a system, the switchover times are correlated. Such systems are typical in manufacturing where the server may move very quickly from one station to another for some period when it is properly functioning, or it may move very slowly once it starts malfunctioning.

Correlated switchover times can be modeled in various ways. The most natural model that also lends itself to exact analysis is to assume that the lengths of the switchover times are modulated by a finite-state Markov chain. The evolution of the Markov chain is independent of the system state and the probability distribution from which the length of the switchover time is sampled depends on the state of the Markov chain (and may also depend on the index of the station just visited or on the index of the station to be visited). The analysis of such a system via the station-time method is straightforward.

A example in which the switch-over time depends on the *state* of the station to be visited has been described by Ferguson [26]. This example corresponds to token ring systems where an additional time may be required in order to seize the token when a station has a message to send, compared to the time required when the token is just passed over to the next station (there are no messages to be sent). In modeling such a system it is assumed that the passing of the token from station i to station $i + 1$ requires a duration of time which is independent of anything in the system. If there is a message at the visited station when the server arrives, an additional *token-acquisition* overhead is incurred. An exact analysis of such systems has not been provided. The main difficulty is that when the station-time method is employed, a boundary condition involving the probability of *no-message* at the station when the token arrives has to be computed, but it is not known how to compute it. As a result, bounds and approximations for the expected delays in such a system have been pursued in Ferguson [26].

VI. Optimization and Conservation Laws

A major goal of any modeling and analysis effort is to develop sufficient understanding of the operation of the systems so that a skilled designer will be able to improve their performance. In many applications in which polling models are used the designer can use several design parameters as to prioritize the queues[4] and improve (optimize) the system performance (e.g., mean customer delay, mean cycle time). Such parameters include the visit order to be used in the system, the duration of visit in each queue, the order of (customer) service within each queue and others.

Naturally, the issues of optimal design and control are more complex than those of analysis. For this reason, these issues are still in a formative and fragmentary stage of research. In this section, we discuss the different approaches which can be used to affect and optimize the performance of polling systems. In addition, we discuss *pseudoconservation laws* which have recently been developed for polling systems. These laws provide closed-form expressions for the weighted sum of the mean waiting times and are useful in the analysis of polling systems as well as in their optimization.

A. Controlling the Service Order

An important parameter affecting the system performance is the order by which the server visits the stations. The visit order can be determined either prior to operation (*static* order) or during operation (*dynamic* order).

1) Static Policies: A generalization of the cyclic order (used in the basic model) is obtained by allowing the server to visit the stations according to a *fixed*—but not necessarily cyclic—*periodic* pattern. The pattern repeats itself every M visits, so in general the polling order is $I(1), I(2), \cdots, I(M), I(1), I(2), \cdots$ where $I(i)$ is the identity of the station polled at the ith visit of the pattern $(1 \leq I(i) \leq N)$. Like the cyclic order, the polling according to a periodic order (or, as called in the literature, polling according to a *polling table*) is a *static policy*, namely, the order is determined prior to the system operation and is fixed during its operation. A specific example of a system that operates with a polling table is the 1A ESS Bell System Switch (see Kruskal [42]).

The analysis of systems with polling tables is similar to that of systems with cyclic polling. This means that the analysis of systems with exhaustive and gated service can be carried out exactly, using either the buffer occupancy approach (leading to the solution of

[4]Another issue of prioritization arises in polling systems in which at every queue there may be several classes of customers and the server's attention to these classes during the visit of the queue is given according to their relative priority. We do not elaborate on this subject and the interested reader may refer to the literature (e.g., Takagi [67]).

MN^2 linear equations) or the station time approach (leading to the solution of $M(M-1)$ linear equations). The general analysis of these systems is provided in Eisenberg [25], Alford and Muntz [2] and Baker and Rubin [3]. Other studies in which specific orders are considered and some of the queues are served in the limited-1 policy, appear in Manfield [53] and Takagi and Murata [69].

Polling systems with a general service order arise naturally in many applications. For instance, in the system consisting of a single computer and N multidrop terminals described in Section II, the computer can serve its internal work after every service of a terminal. Modeling this system can be done via a polling model with $N+1$ queues (where queue 1 represents the computer and queues $2,\cdots,N+1$ represent the terminals) in which the visit pattern is $1,2,1,3,\cdots,1,N+1$. Another example is the scanning of disks in computer systems, in which it is natural to poll the sectors of the disk in a scanning order. Thus, if the N sectors are represented by N queues, then the visit pattern is $1,2,\cdots,N-1,N,N-1,\cdots,2$.

General service orders can be used as an effective means for the optimization of polling systems. They give one the opportunity to assign certain stations high priority by visiting them more frequently than once per cycle. It can be shown that such priority assignments can reduce mean waiting times in high-priority stations, and for systems where the high-priority stations are responsible for a substantial fraction of the traffic, the overall mean waiting time can be reduced as well (see Baker and Rubin [3]). The importance of the general periodic service order is, therefore, twofold: First, it can be used to effectively affect the system performance, and second, the prediction of the system performance under alternative orders is readily available. Thus, one can seek the optimal service order by examining various orders and analyzing them. Nonetheless, the characterization of the optimal service order is still an open and challenging research problem.

As an example which demonstrates the usefulness of polling tables for affecting the system performance we consider a system consisting of 12 small stations (the arrival rate of each of them is relatively low) indexed $1,2,\cdots,12$ and one large station (with high arrival rate). For simplicity we assume that the service times and switchover times in all stations are deterministic with means $b_i=1$ and $s_i=1$ and that the gated service policy is used in all stations. Let us assume that the performance measure of interest is the mean waiting time of arbitrary customers in the system, namely $E[W]=\sum_{i=1}^{N}\lambda_i E[W_i]/\lambda$ (where $\lambda=\sum_{i=1}^{N}\lambda_i$). In the example we examine the effect of the frequency of visit at the large station on $E[W]$. We consider six visit patterns, in which the server visits the large station 1, 2, 3, 4, 6, and 12 times per cycle; in all patterns each of the small stations is visited once in a cycle and the visits of the large station are evenly spaced over the pattern (e.g., in the case of four visits the pattern is 1, 2, 3, *large*, 4, 5, 6, *large*, 7, 8, 9, *large*, 10, 11, 12, *large*). We consider five cases in which the arrival rate of the large station takes on the values 0.02, 0.08, 0.18, 0.32, and 0.72; In all five cases the arrival rates of each of the small queues is 0.02. Thus, the ratio $\gamma \triangleq \sqrt{\rho_{large}/\rho_{small}}$ takes on the values 1, 2, 3, 4, and 6 in these five cases, respectively.

In Table II we depict the mean customer waiting time in each of these cases for several visit frequencies (number of visits per cycle) of the large station. In addition we consider a case (last row in the table) in which the large station is visited four times during a cycle but its visits are unevenly spaced over the pattern (the pattern is 1, *large*, 2, 3, 4, 5, 6, *large*, 7, *large*, 8, 9, 10, 11, 12, *large*). Table II demonstrates that the system performance is significantly affected by the visit pattern employed. We may observe that a rule which selects the visit frequency of a station in proportion to the square root of its utilization (ρ_i) works very well in minimizing the mean customer waiting time of this system. The analysis of this optimization issue and the derivation of the square root rule can be found in Boxma, Levy, and Weststrate [11]. We may further learn from the table that unevenly spaced visit patterns (last row) usually perform worse than evenly spaced patterns.

2) Dynamic Policies: As an alternative to using a polling table,

TABLE II
MEAN CUSTOMER WAITING TIME

Visit frequency of large station	Utilization of large station				
	0.02	0.08	0.18	0.32	0.72
1 (even)	**9.11**	10.13	12.54	18.23	263.05
2 (even)	9.42	**9.50**	10.45	13.59	169.50
3 (even)	9.96	9.65	**10.07**	12.37	140.43
4 (even)	10.54	10.03	10.14	**12.02**	127.50
6 (even)	11.77	10.97	10.71	12.14	**117.75**
12 (even)	15.54	14.12	13.81	14.26	120.75
4 (uneven)	10.61	10.18	10.40	12.30	128.25

which dictates a *static* visit order, one may use a *dynamic order* in which the service order is changing dynamically and is determined according to the system state during its operation. For example, a possible policy is to observe the contents of the different queues and to decide to serve next the queue which is mostly loaded. The advantage of dynamic orders is that they are very sensitive to the actual system state and thus can be used to improve its performance. The disadvantages of such orders is that they require information gathering during operation and that they are hard to analyze. As a result, very little is known today about systems with dynamic orders.

For systems *without* switchover periods, in which there is full information about the buffer contents and one is allowed to choose the next queue to be served after every service completion, the optimal operation rule is simple. It has been shown that the weighted sum of the mean waiting times, $\sum_{i=1}^{N}c_i\lambda_i E[W_i]/\lambda$ (where the c_i's are constants and $\lambda=\sum_{i=1}^{N}\lambda_i$), can be minimized by using a simple rule, named the μc rule. With this rule, the queues are ranked according to the ratio c_i/b_i (in the literature $1/b_i$ is usually denoted by μ_i, from which the μc rule gets its name) and ordered according to this ratio (highest ratio—highest priority), and the server always serves acustomer from the highest ranked queue which is nonempty. This rule applies both for systems in which the service times are generally distributed and no preemption is allowed (Meilijson and Yechiali [54]) and for systems in which the service times possess the memoryless property and preemption is allowed (Baras, Dorsey, and Makowski [4], Baras, Ma, and Makowski [5], and Buyukkoc, Varaiya, and Walrand [16]). The optimal operation rule (within the class of all nonpreemptive policies) has also been provided for a more general system which allows customer routing and general service times. The optimal rule in this framework is expressed in an algorithmic way (rather than a closed-form expression) and was provided by Klimov [39], [40]. Once a fixed priority order is determined, the delay analysis of the priority system can be carried out as well (see, e.g., Kleinrock [37] for systems with no customer routing and Simon [62] and Sidi and Segall [61] for systems with customer routing).

When switchover times are not identically zero, the problem of the optimal control of a polling system is much more difficult and is still open. Preliminary results in this area are available in Hofri and Ross [32] that considered the case of two stations. Based on some conjectures, it is shown that the solution to this problem is of a threshold type, i.e., there exist thresholds that determine when the server switches from one queue to the other.

Another result regarding the optimal operation of systems with nonzero switchover periods deals with *semidynamic* policies for selecting the visit order (Browne and Yechiali [14]). Browne and Yechiali [14] consider policies which commit to visit every queue exactly once in each cycle and which determine the visit order (of the next cycle) at the beginning of the cycle. The analysis of these policies does not provide derivation of any measure of the customer delay. Rather, what is provided is a simple and optimal rule for selecting the visit order when the goal is to minimize the mean duration of the next cycle. It was found that without switching

times, it is optimal to rank the stations by an increasing order of N_i/λ_i (where N_i is the number of customers present at queue i at the beginning of the cycle) for *both* gated and exhaustive service. Note that this rule is independent of the service time b_i. When switchover times are included, a more general index type rule applies.

B. Control of Service Duration

The service policy within a station determines the service duration in that station. A very effective way to control the service duration is via the limited policies. By assigning proper limits on the maximal number of customers that can be served in a station during one visit of the server, it should be possible to optimize the performance of the system.

Limited policies can be either deterministic or probabilistic. In a deterministic limited policy, a limit L_i is fixed for station i and no more than L_i customers are served per one visit of the station. There are gated and exhaustive versions of limited policies. In a gated-limited policy, the number of customers served during a visit is the minimum between L_i and the number of customers present in the station when it was polled. An exhaustive-limited policy is essentially an exhaustive policy, except that the server switches from station i if he already served L_i customers.

In probabilistic limited policies, the limit is not fixed but determined randomly. One method to choose such a random limit is known as a *Bernoulli* policy (Keilson and Servi [36] and Servi [59]). According to this method when the service of a type-i customer is completed, a type-i coin is tossed to determine whether the limit has been achieved (and therefore to switch to the next station) or not. Again, there are gated and exhaustive versions of Bernoulli policies. In a Bernoulli-gated policy the server will deterministically switch to the next station once all the customers which were present at the served queue at the polling instant are served. In a Bernoulli-exhaustive policy such deterministic switching occurs only when the served queue becomes empty.

The advantage of both the deterministic limited policies and the Bernoulli policies is that they form effective means for affecting the system performance. Their disadvantage, however, is that they are too complicated to be analyzed exactly for general systems. As a result they do not lend themselves for efficient system optimization.

An alternative to these policies is the family of fractional service policies (Levy [49], [50]). In a fractional service policy, each queue is assigned a parameter p_i and the philosophy behind the policy is to serve at each visit to queue i a "fraction" p_i of the work present at that queue. The policies available in the literature concentrate on serving a fraction p_i of the customers present at the queue. More precisely, the number of customer served is a random variable whose mean is p_i times the number of customers present at the queue. Here too, there are gated variations (*binomial-gated* policy, in which the number of customers served is a function of the number of customers present at the polling instant) and exhaustive variations (*fractional-exhaustive*, in which the number of customers served depends also on the number of customers arriving to the queue during its service). The advantage of these methods is that they can affect the system performance and that they lend themselves to exact analysis. However, it seems that they affect the system performance less than the limited policies do.

Finally, the server can limit the *time* it spends in a queue in order to reduce the starvation of other nonempty queues. Such a policy introduces fairness into the system but is very difficult to analyze. Only the case of two queues with exponential service times and no switchover times have been analyzed so far (see Coffman, Fayolle, and Mitrani [20]).

C. Pseudoconservation Laws

It is well known that work conservation in a system gives rise to a conservation law for mean waiting times (Kleinrock [37]), i.e., a linear relation between the mean waiting times that does not depend on the order of service of customers (as long as they are not preempted). Polling systems are not work conserving systems since the server remains idle during switchover periods, although work might be present in the system. Nevertheless, pseudoconservation laws for many polling systems have been discovered in recent years (Watson [73] and Ferguson and Aminetzah [27]). Recent derivations of these laws are based on a decomposition theorem of Boxma and Groenendijk [9] (which is motivated by previous decomposition results of Fuhrmann [28] and Fuhrmann and Cooper [29]) that proves that the total amount of work in the system is composed of two independent components; one is the amount of work in the corresponding system without switchover times; the other is the amount of work at an arbitrary epoch during a switchover period.

Conservation laws for mean waiting times serve several useful purposes. In many complex sysems (limited-1 polling, for instance) they are the only meaningful exact relations that can be obtained. Thus, they provide important qualitative insight into the behavior of such systems. They can also serve as a test for the quality of suggested approximations, and be useful in constructing approximations for individual mean delays in the various stations for complex systems. Even when exact results for the individual mean delays are available, conservation laws are excellent tools for checking the correctness of the (usually complex) numerical computations.

The basic form of a conservation law (when applied to the basic system) is

$$\sum_{i=1}^{N} \rho_i E[W_i] = \rho \frac{\sum_{i=1}^{N} \lambda_i b_i^{(2)}}{2(1-\rho)} + \rho \frac{s^{(2)}}{2s} + \frac{s}{2(1-\rho)}\left[\rho^2 - \sum_{i=1}^{N} \rho_i^2\right] + \sum_{i=1}^{N} Z_i$$

where W_i is the waiting time at station i and Z_i are quantities that depend on the service discipline that is in use. For instance, $Z_i = 0$ and $Z_i = s\rho_i^2/(1-\rho)$ for stations in which the exhaustive discipline and the gated discipline are employed, respectively.

Conservation laws are available for most polling models discussed in this paper. For the basic model with cyclic service time such laws are available in Ferguson and Aminetzah [27], Watson [73], Boxma and Groenendijk [9]. For the Bernoulli service discipline in Tedijanto [72], for binomial-gated and binomial-exhaustive in Levy [49] and [50], for systems with polling tables in Boxma, Groenendijk, and Weststrate [10], for random polling in Boxma and Weststrate [13], for polling systems with probabilistic routing in Sidi and Levy [60] and for polling systems with simultaneous arrivals to the queues in Levy and Sidi [51]. For a survey of the conservation law results see Boxma [8].

VII. Summary and Future Work

We reviewed in this paper the state of the art in the area of polling system. We focused the discussion on the capabilities and limitations of polling system models and on their use in modeling various applications.

Two subjects seem to be of great importance for future work. The first is the analysis of systems with limited service; the analysis of these systems is available today via approximation methods which are not very accurate. Improved techniques are required for better performance evaluation of many existing systems. The second area, in which research has just started, is that of optimization. The development of effective optimization techniques will contribute to improve the performance of polling systems and many of their applications.

Appendix

A. The Buffer-Occupancy Method

The solution of the mean sojourn time in the cyclic system with exhaustive service, using the buffer occupancy approach, is done as follows. We define $f_i(j,k) \triangleq E[X_i^j X_i^k]$, for $j \neq k$ and $E[(X_i^j)^2 - X_i^j]$ for $j = k$. Then the variables $\{f_i(j,k)\}$ form a set of N^3

linear equations:

$$f_{i+1}(j,k) = \lambda_j\lambda_k s_i^{(2)} + s_i\lambda_k f_i(j) + s_i\lambda_j f_i(k) + f_i(i)\lambda_j\lambda_k\left[\frac{2b_i s_i}{1-\rho_i} + \frac{b_i^{(2)}}{(1-\rho_i)^3}\right] + \frac{b_i}{1-\rho_i}\left[f_i(i,j)\lambda_k + f_i(i,k)\lambda_j\right] + f_i(j,k) + f_i(i,i)\lambda_j\lambda_k\left(\frac{b_i}{1-\rho_i}\right)^2 \quad i \neq j,\ i \neq k \quad \text{(A.1)}$$

$$f_{i+1}(i,j) = \lambda_i\lambda_j s_i^{(2)} + s_i\lambda_i\left[f_i(j) + \frac{f_i(i)\lambda_j b_i}{1-\rho_i}\right] \quad i \neq j \quad \text{(A.2)}$$

$$f_{i+1}(i,i) = \lambda_i^2 s_i^{(2)}. \quad \text{(A.3)}$$

Using the values of $f_i(i,i)$ and $f_i(i)$ [defined to be $E[X_i^i] = s\lambda_i/(1-\rho)$] the mean sojourn time at queue i is computed

$$E[T_i] = b_i + \frac{\lambda_i b_i^{(2)}}{2(1-\rho_i)} + \frac{f_i(i,i)}{2\lambda_i f_i(i)}. \quad \text{(A.4)}$$

B. The Station-Time Method

The solution of the mean sojourn time in the cyclic system with exhaustive service, using the station-time approach, is done as follows. Let g_{ij} be defined as the covariance of U_i and U_j when station i is visited prior to station j, namely,

$$g_{ij} = \begin{cases} E\left[\left(U_i - E[U_i]\right)\left(U_j - E[U_j]\right)\right] & j \leq i \\ E\left[\left(U_{i+N} - E[U_{i+N}]\right)\left(U_j - E[U_j]\right)\right] & j > i \end{cases}.$$

Then, the variables g_{ij} form a set of N^2 equations

$$g_{ij} = \frac{\rho_i}{1-\rho_i}\left(\sum_{m=i+1}^{N} g_{jm} + \sum_{m=1}^{j-1} g_{jm} + \sum_{m=j}^{i-1} g_{mj}\right) \quad j < i \quad \text{(A.5)}$$

$$g_{ij} = \frac{\rho_i}{1-\rho_i}\left(\sum_{m=i+1}^{j-1} g_{jm} + \sum_{m=j}^{N} g_{mj} + \sum_{m=1}^{i-1} g_{mj}\right) \quad j > i \quad \text{(A.6)}$$

$$g_{ii} = \frac{r_{i-1}^{(2)} - r_{i-1}^2}{(1-\rho_i)^2} + \frac{\lambda_i b_i^{(2)}}{(1-\rho_i)^3}\cdot\frac{(1-\rho_i)\sum_{j=1}^{N} r_j}{1-\rho} + \frac{\rho_i}{1-\rho_i}\sum_{\substack{j=1 \\ j\neq i}}^{N} g_{ij}. \quad \text{(A.7)}$$

From the variables g_{ij} $(1 \leq j \leq N)$ and the system parameters, $E[T_i]$ can directly be computed (see, e.g., Ferguson and Aminetzah [27]).

C. Limited-1 System

The closed-form expression for the mean sojourn time in the fully symmetric cyclic system with limited-1 service is given by

$$E[T_i] = b_i + \frac{s_i^{(2)} - s_i^2}{2s_i} + \frac{N\lambda_i b_i^{(2)} + s_i(N+\rho) + N\lambda_i\left(s_i^{(2)} - s_i^2\right)}{2\left(1-\rho-N\lambda_i s_i\right)}.$$

Acknowledgment

We would like to thank O.J. Boxma, R.B. Cooper, and U. Yechiali for their helpful comments and suggestions.

References

[1] N. Abramson, "The ALOHA system—Another alternative for computer communications," in *Proc. 1970 Fall Joint Comput. Conf.*, Houston, TX, Nov. 1970, pp. 17-19.

[2] M. Alford and R. R. Muntz, "Queueing models for polled multi-queues," in *4th Texas Conf. Comp. Syst.*, pp. 5B-2.1-2.5, 1975.

[3] J. E. Baker and I. Rubin, "Polling with a general-service order table," *IEEE Trans. Commun.*, vol. COM-35, pp. 283-288, Mar. 1987.

[4] J. S. Baras, A. J. Dorsey, and A. M. Makowski, "Two competing queues with geometric service requirements and linear costs: The μc rule is often optimal," *Adv. Appl. Prob.*, vol. 17, pp. 186-209, Mar. 1985.

[5] J. S. Baras, D.-J. Ma, and A. M. Makowski, "K competing queues with geometric service requirements and linear costs: The μc rule is always optimal," *Syst. Cont. Lett.*, vol. 6, no. 3, pp. 173-180, Aug. 1985.

[6] J. P. C. Blanc, "A numerical approach to cyclic-service queueing models," Rep. Dep. Economics, Tilburg Univ., Tilburg, The Netherlands; also to appear in *Queueing Syst.*, vol. 6, no. 1, 1990.

[7] O. J. Boxma, "Two symmetric queues with alternating service and switching time," in *Performance '84*, E. Gelenbe, Ed. North-Holland, Amsterdam, pp. 409-431.

[8] O. J. Boxma, "Workloads and waiting times in single-server systems with multiple customer classes," *Queueing Syst.*, vol. 5, pp. 185-214, 1989.

[9] O. J. Boxma and W. P. Groenendijk, "Pseudo-conservation laws in cyclic queues," *J. Appl. Prob.*, vol. 24, pp. 949-964, 1987.

[10] O. J. Boxma, W. P. Groenendijk, and J. A. Weststrate, "A pseudo-conservation law for service systems with a polling table," *IEEE Trans. Commun.*, to be published.

[11] O. J. Boxma, H. Levy, and J. A. Weststrate, "Optimization of polling systems," Rep., Cent. Math. and Comp. Sci., Amsterdam, Nov. 1989; also in *Performance'90*, P. J. B. King, I. Mitrani, and R. J. Pooley, Eds. Amsterdam, The Netherlands: North-Holland, 1990, pp. 349-361.

queue systems with cyclic service," *Perform. Eval.*, vol. 7, no. 4, pp. 299-308, Nov. 1987.

[13] O. J. Boxma and J. A. Weststrate, "Waiting times in polling systems with Markovian server routing," preprint 1989.

[14] S. Browne and U. Yechiali, "Dynamic priority rules for cyclic-type queues," *Adv. Appl. Prob.*, vol. 21, no. 2, pp. 432-450, June 1989.

[15] W. Bux, "Local-area subnetworks: A performance comparison," *IEEE Trans. Commun.*, vol. COM-29, pp. 1465-1473, Oct. 1981.

[16] C. Buyukkoc, P. Varaiya, and J. Walrand, "The $c\mu$ rule revisited," *Adv. Appl. Prob.*, vol. 17, pp. 237-238, Mar. 1985.

[17] I. Capetanakis, "Tree algorithms for packet broadcast channels," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 505-515, Sept. 1979.

[18] R. T. Carsten, E. E. Newhall, and M. J. M. Posner, "A simplified analysis of scan times in an asymmetrical newhall loop with exhaustive service," *IEEE Trans. Commun.*, vol. COM-25, no. 9, pp. 951-957, Sept. 1977.

[19] J. W. Cohen and O. J. Boxma, *Boundary Value Problems in Queueing System Analysis*. Amsterdam, The Netherlands: North-Holland, 1983.

[20] E. G. Coffman, G. Fayolle, and I. Mitrani, "Two queues with alternating service periods," *Performance '87*, pp. 227-237, Dec. 1987.

[21] R. B. Cooper, "Queues served in cyclic order: Waiting times," *Bell Syst. Tech. J.*, vol. 49, pp. 399-413, Mar. 1970.

[22] R. B. Cooper and G. Murray, "Queues served in cyclic order," *Bell Syst. Tech. J.*, vol. 48, pp. 675-689, Mar. 1969.

[23] B. T. Doshi, "Queueing systems with vacations—A survey," *Queueing Syst.*, vol. 1, pp. 29-66, 1986.

[24] M. Eisenberg, "Two queues with alternating service," *Siam J. Appl. Math.*, vol. 36, no. 2, pp. 287-303, Apr. 1979.

[25] M. Eisenberg, "Queues with periodic service and changeover time," *Oper. Res.*, vol. 20, pp. 440-451, 1972.

[26] M. J. Ferguson, "Mean waiting time for a token ring with station dependent overheads," in *Advances in Telecommunication Networks Series*, R. L. Pickholtz, Ed. 1986, pp. 43-68.

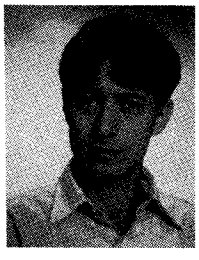
[27] M. J. Ferguson and Y. J. Aminetzah, "Exact results for nonsymmetric token ring systems," *IEEE Trans. Commun.*, vol. COM-33, pp. 223-231, Mar. 1985.

[28] S. W. Fuhrmann, "Symmetric queues served in cyclic order," *Oper. Res. Lett.*, vol. 4, no. 3, pp. 139-144, Oct. 1985.

[29] S. W. Fuhrmann and R. B. Cooper, "Stochastic decompositions in the

$M/G/1$ queue with generalized vacations," *Oper. Res.*, vol. 33, 1117-1129, 1985.
[30] S. W. Fuhrmann and Y. T. Wang, "Mean waiting time approximations of cyclic service systems with limited service," *Performance '87*, P. J. Courtois and G. Latouche, Eds. pp. 253-264.
[31] O. Hashida, "Analysis of multiqueue," *Rev. Elec. Commun. Lab.*, vol. 20 (3, 4), pp. 189-199, 1972.
[32] M. Hofri and K. W. Ross, "On the optimal control of two queues with server set-up times and its analysis," *Siam J. Comp.*, vol. 16, no. 2, pp. 399-419, Apr. 1987.
[33] P. Humblet, "Source coding for communication concentrators," *Electron. Syst. Lab.*, Massachusetts Inst. Technol., Cambridge, ESL-R-798, Jan. 1978.
[34] O. C. Ibe and X. Cheng, "Approximate analysis of asymmetric single-server token-passing systems," *IEEE Trans. Commun.*, vol. COM-37, pp. 572-577, June 1989.
[35] T. Katayama, "A cyclic service tandem queueing model," Rep. NTT Commun. Switch. Lab., Tokyo; also to appear in *Queueing Syst.*
[36] J. Keilson and L. D. Servi, "Oscillating random walk models for $GI/G/1$ vacation systems with Bernoulii schedules," *J. Appl. Prob.*, vol. 23, pp. 790-802, Sept. 1986.
[37] L. Kleinrock, *Queueing Systems, Volume 2: Applications*. New York: Wiley-Interscience, 1976.
[38] L. Kleinrock and H. Levy, "The analysis of random polling systems," *Oper. Res.*, vol. 36, no. 5, pp. 716-732, Sept.-Oct. 1988.
[39] G. P. Klimov, "Time-sharing service systems I," *Theory Prob. Appl.*, vol. 19, pp. 532-551, 1974.
[40] ____, "Time-sharing service systems II," *Theory Prob. Appl.*, vol. 23, pp. 314-321, 1978.
[41] A. G. Konheim and B. Meister, "Waiting lines and times in a system with polling," *J. Assoc. Comput. Mach.*, vol. 21, pp. 470-490, 1974.
[42] J. B. Kruskal, "Work-schedule algorithms: A non-probabilistic queueing study (with possible application to No. 1A ESS)," *Bell Syst. Tech. J.*, vol. 48, no. 9, pp. 2963-2974, Nov. 1969.
[43] P. J. Kuehn, "Multiqueue systems with nonexhaustive cyclic service," *Bell Syst. Tech. J.*, vol. 58, no. 3, pp. 671-698, Mar. 1979.
[44] S. S. Lam, "Packet broadcast networks—A performance analysis of the *R*-ALOHA protocol," *IEEE Trans. Comput.*, vol. C-29, pp. 596-603, July 1980.
[45] K. K. Leung, "Waiting time distribution for token-passing systems with limited-one service via discrete Fourier transforms," *IEEE INFOCOM '90*.
[46] H. Levy, Ph.D. dissertation, Dept. Comp. Sci., UCLA.
[47] ____, "Delay computation and dynamic behavior of non-symmetric polling systems," *Perform. Eval.*, vol. 10, pp. 35-51, 1989.
[48] ——, "On the complexity of Swartz's method for calculating the expected delay in non-symmetric polling systems," preprint; submitted.
[49] ____, "Analysis of cyclic-polling systems with binomial-gated service," *Proc. Int. Sem. Perform. Distributed Parallel Syst.*, Kyoto, Japan, Dec. 1988, T. Hasegawa, H. Takagi, and Y. Takahashi, Eds.
[50] ____, "Optimization of polling systems: The fractional exhaustive service method," Tech. Rep. Dep. Comput. Sci., Tel-Aviv Univ., Nov. 1988.
[51] H. Levy and M. Sidi, "Correlated arrivals in polling systems," Tech. Rep. EE-676, Dept. of Elec. Eng., Technion—Israel Institute of Technology, June 1988; also in *INFOCOM '89*, 1989, pp. 907-913.
[52] H. Levy, M. Sidi, and O. Boxma, "Dominance relations in polling systems," *Queueing Systems, Theory and Applications*, vol. 6, pp. 155-172, 1989.
[53] D. R. Manfield, "Analysis of a polling system for two-way traffic," *IEEE Trans. Commun.*, vol. COM-33, pp. 1001-1006, Sept. 1985.
[54] I. Meilijson and U. Yechiali, "On optimal right-of-way policies at a single-server station when insertion of idle times is permitted," *Stochast. Processes Appl.*, vol. 6, pp. 23-52, 1977.
[55] M. Nomura and K. Tsukamoto, "Traffic analysis on polling systems," *Trans. Inst. Elec. Commun. Eng. Japan*, vol. J61-B, no. 7, pp. 600-607, July 1978.
[56] I. Rubin and L. F. De Moraes, "Message delay analysis for polling and token multiple-access schemes for local communication networks," *IEEE J. Select. Areas Commun.*, vol. SAC-1, pp. 935-947, Nov. 1983.
[57] D. Sarkar and W.I. Zangwill, "Expected waiting time for nonsymmetric cyclic queueing systems—Exact results and applications," Tech. Rep.
[58] L. D. Servi, "Capacity estimation of cyclic queues," *IEEE Trans. Commun.*, vol. COM-33, pp. 279-282, Mar. 1985.
[59] ____, "Average delay approximation of $M/G/1$ cyclic service queues with Bernoulli schedules," *IEEE J. Select. Areas Commun.*, vol. SAC-4, pp. 813-822, Sept. 1986.
[60] M. Sidi and H. Levy, "A queueing network with a single cyclically roving server," Tech. Rep., Dep. Elec. Eng., Technion—Israel Inst. Technol., submitted; also in *Performance '90*, P. J. B. King, I. Mitrani, and R. J. Pooley, Eds. Amsterdam, The Netherlands: North-Holland, 1990, pp. 319-331.
[61] M. Sidi and A. Segall, "Structured priority queueing systems with applications to packet-radio networks," *Perform. Eval.*, vol. 3, pp. 264-275, 1983.
[62] B. Simon, "Priority queues with feedback," *J. Assoc. Comp. Mach.*, vol. 31, pp. 134-149, 1984.
[63] M. M. Srinivasan, "An approximation for mean waiting times in cyclic server systems with non-exhaustive service," *Perform. Eval.*, vol. 9, pp. 17-33, 1988.
[64] G. B. Swartz, "Polling in a loop system," *J. Assoc. Comp. Mach.*, vol. 27, pp. 42-59, 1980.
[65] H. Takagi, "Mean message waiting times in symmetric multi-queue systems with cyclic service," *Perform. Eval.*, vol. 5, no. 4, pp. 271-277, Nov. 1985.
[66] ____, *Analysis of Polling Systems*. Cambridge, MA: M.I.T. Press, Apr. 1986.
[67] ____, "Queueing analysis of polling systems with zero switch-over times," IBM TRL Tech. Rep. TR87-0034.
[68] ____, "Queueing analysis of polling models," *ACM Comp. Surv.*, vol. 20, no. 1, pp. 5-28, Mar. 1988.
[69] H. Takagi and M. Murata, "Queueing analysis of scan-type TDM and polling systems," *Comput. Network Perform. Eval.*, T. Hasegawa, H. Takagi, and Y. Takahashi, Eds. New York: Elsevier Science, 1986, pp. 199-211.
[70] T. Takine, Y. Takahashi, and T. Hasegawa, "Performance analysis of a polling system with single buffers and its applications to interconnected neworks," *IEEE J. Select. Areas Commun.*, vol. SAC-4, pp. 802-812, Sept. 1986.
[71] ____, "Exact analysis of asymmetric polling systems with single buffers," *IEEE Trans. Commun.*, vol. COM-36, pp. 1119-1127, Oct. 1988.
[72] Tedijanto, "Exact analysis for the cyclic-service queue with a Bernoulli schedule," preprint, 1988.
[73] K. S. Watson, *Performance Evaluation of Cyclic Service Strategies —A Survey*, in *Perform. '84*, E. Gelenbe, Ed. New York: North-Holland, 1985, pp. 521-533.
[74] W. P. Groenendijk, "A conservation-law based approximation algorithm for waiting times in polling systems," Rep. OS-R8816, 1988.

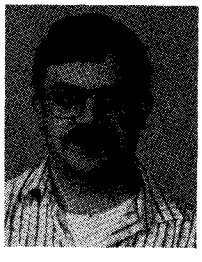
★



Hanoch Levy (S'83-M'86) received the B.A. degree in computer science with distinction from the Technion—Israel Institute of Technology in 1980, and the M.Sc. and the Ph.D. degrees in computer science from the University of California at Los Angeles, in 1982 and 1984, respectively.

From 1984 to 1987 he was a member of the technical staff in the Department of Teletraffic Theory at AT&T Bell Laboratories. Currently he is with the Department of Computer Science, Tel-Aviv University, Tel-Aviv, Israel. His current interests include computer communication networks, performance evaluation of computer systems, and queueing theory.

★



Moshe Sidi (S'77-M'82-SM'87) received the B.Sc., M.Sc., and the D.Sc. degrees from the Technion—Israel Institute of Technology, Haifa, Israel, in 1975, 1979, and 1982, respectively, all in electrical engineering.

From 1975 to 1981, he was a Teaching Assistant and a Teaching Instructor at the Technion in communication and data networks courses. In 1982 he joined the faculty of Electrical Engineering Department at the Technion. During the academic year 1983-1984 he was a Post-Doctoral Associate at the Electrical Engineering and Computer Science Department at the Massachusetts Institute of Technology, Cambridge, MA. During 1986-1987 he was a visiting scientist at IBM, Thomas J. Watson Research Center, Yorktown Heights, NY.

Dr. Sidi received the New England Academic Award in 1989. Currently, he serves as the Editor for Communication Networks in the IEEE TRANSACTIONS ON COMMUNICATIONS. His current research interests are in queueing systems and in the area of computer communication networks.