Overload control for SPC telephone exchanges

- refined models and stochastic control

R.K. Boel

Laboratorium voor Theoretische Elektriciteit, Rijksuniversiteit Gent
Grote Steenweg Noord 2, B9710 Gent-Zwijnaarde, Belgium

J.H. van Schuppen

Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

In telephone networks the switching and connecting operations are performed by the exchanges. The Stored Program Control (SPC) exchanges which are nowadays installed are computer controlled. One of the problems with these exchanges is the severe performance degradation during periods in which the demand for service exceeds the design capacity. The problem of overload control is then to maximize the number of successfully completed calls. In this paper two models for overload control of an SPC exchange are proposed that are refinements of an earlier model. A stochastic control problem for one of these models is shown to have a bang-bang type of optimal solution.

1. Introduction

The purpose of this paper is to present refined models for the operation of SPC telephone exchanges and to consider a stochastic control problem for overload control.

Telephone exchanges are the operational units at the nodes of telephone networks. In the last few years computer-controlled *Stored Program Control (SPC)* exchanges have been installed. In such an exchange the operations are executed by a processor according to a stored program. The operations of such an exchange may be summarized as follows. If a customer picks up the receiver this action generates a signal that will be detected by the exchange. After some delay, the exchange answers by sending a dial tone. After the customer has dialed the desired number, the exchange establishes, with some delay and depending on availability, a connection with the requested phone. All these different tasks have to be executed sequentially by the processor.

The performance of an SPC exchange can degrade considerably during periods in which the demand for service exceeds the design capacity [5]. The response time of the exchange during such periods is relatively long. This may cause impatient customers to dial prematurely, before a dial tone is given, after which an incompletely received telephone number takes up processor capacity and ends up as an unsuccessful call. Other requests for connections, that have been transmitted properly to the exchange, may encounter long processing delays. This then causes customers to abandon the call request and, possibly, to redial soon after. In this case capacity of the exchange is also wasted. That this performance degradation is a serious problem may be concluded from the data of [5].

The *problem of overload control* is then to maximize the number of successfully processed call requests. A call request may either be given access or be refused access. This decision represents the control action. References on this problem are [1,5,6,7,8,9,10,13]. The overload control problem also arises in mobile automatic telephony, in PBX business exchanges and other communication equipment.

A model for overload control has been proposed by one of the authors [13], based on an approach developed by F.C. Schoute [8,9,10]. This model consists of a hierarchical queueing system representing calls-in-build-up and tasks for the processor. Weak points in this model are: 1. there is no model

for the successfully processed call requests; 2. retrials are not modeled. The present paper gives two refined models of an SPC exchange and then considers a stochastic control problem for one of these models.

The terminology used for counting and jump processes may be found in [4]. For a survey of modeling, stochastic filtering and stochastic control of such processes see [3]. References on the control of queueing systems are [11,12].

The authors acknowledge useful discussions with F.C. Schoute of Philips Telecommunicatie Industrie on the problem of overload control. They also thank the governments of Belgium and The Netherlands which through their cultural exchange agreement have provided financial support for the cooperation of the two authors.

2. A HIERARCHICAL QUEUEING SYSTEM

In this section the model for an SPC telephone exchange of [13] is summarized and discussed.

The mathematical model

A brief description of the technical operation of an SPC exchange follows. A customer who picks up the receiver sends thus a signal to the telephone exchange, to be called a *call request.* Call requests, when detected by the exchange, are placed in a buffer by the central processor. These buffered requests will be termed *calls-in-build-up.* During its presence at the buffer a call-in-build-up generates *tasks* which are executed sequentially by the central processor. Examples of tasks are a request for a dial tone, detection and recognition of dialed digits, the establishment of a connection, and related actions.

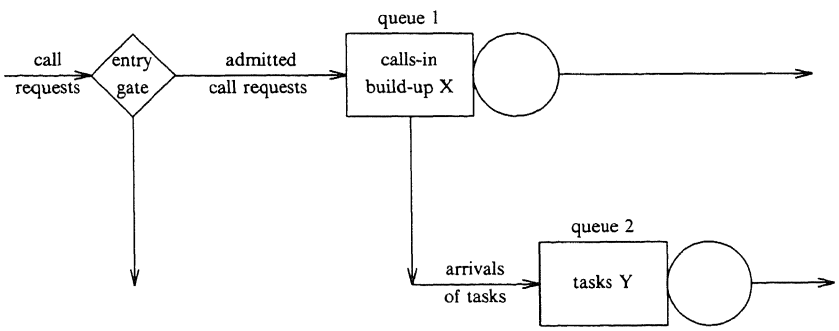

FIGURE 1. A hierarchical model for overload control.

The dynamics of the processor load may be modeled by a hierarchical queueing system as in figure 1. Call requests represented by an arrival process may or may not be admitted to the exchange, possibly based on the outcome of a toss of a coin. The probability of admission represents the control action. Call requests that have been refused access are assumed not to return. A call request that has been admitted to the exchange is placed in a buffer with an infinite number of servers. Such calls-in-build-up have independent identically exponentially distributed service times.

During its presence at queue 1 a call-in-build-up generates tasks that have to be executed by the central processor. The intensity of the arrival process of tasks is assumed to be proportional to the number of calls-in-build-up in queue 1. The task execution process is modeled by a single server queue M/M/1 operating on a first-in-first-out rule and with an infinite buffer.

Assume given a complete probability space (Ω, F, P) and a time index set $T = R_+$. Let

$$Z_+ = \{1,2,...\}, \quad N = \{0,1,2,...\}.$$

The construction of the hierarchical queueing system proceeds via a measure transformation indexed by a class of control policies U. For each admissable control policy $U(.)$ one obtains the following dynamic representation for the hierarchical queueing system,

$$dX(t) = [\lambda_0 U(t) - \mu_1 X(t)]dt + dM_1(t), \quad X(0), \tag{2.1}$$

$$dY(t) = [\lambda_2 X(t) - I_{(Y(t)>0)}\mu_2]dt + dM_2(t), \quad Y(0), \tag{2.2}$$

where $X: \Omega \times T \to R_+$ represents the number of calls-in-build-up, $Y: \Omega \times T \to R_+$ the number of tasks waiting or being served and M_1, M_2 local martingales. For details on this model see [13, sections 2 and 3].

Criticism and comments on the hierarchical queueing system

1. How to represent a successfully processed call request? A call request will be termed *successful* if it reaches a ringing or busy signal at the requested phone. The goal of overload control is to maximize the number of successfully processed call requests. It is preferable to exhibit successful call requests explicitly in the model rather than only in the cost function.

   Clearly a call request will be processed successfully if the delay in giving a dial tone and in establishing a connection is smaller than the time a customer is willing to wait. Thus one needs to model the time delays. How to do this is discussed at point 3 below.
2. The criticism may be voiced that in the hierarchical queueing system there is no connection between the server in queue 1 and the server in queue 2. Thus a call-in-build-up may leave from queue 1 before the tasks it has generated have been processed by the central processor in queue 2.

   To counter this criticism recall that a call request in queue 1 represents the active task generation phase during which tasks, such as a request for a dial tone, for a connection and for routing are generated. Should there then be a connection between the departure processes of queue 1 and queue 2, in particular should the time a call request is in the active task generation phase depend on the processing of its tasks? A little thought leads one to conclude that one has to distinguish call requests that are actively generating tasks and those that are merely waiting for the processing of these tasks. Queue 1 should include the former, another queue could represent the latter. Furthermore, there should be a connection between the processing of tasks and the waiting call requests. This the leads to the question what is the delay in processing a call request compared with the patience of a customer?

   Remark that in general the active task generation phase is longer than the period during which the customer dials the telephone number.
3. What is the time necessary to process a call request and how can one model the patience of a customer? The customer notices two types of delay, one in waiting for a dial tone and one in waiting for the connection. In the hierarchical model these delays are not explicitly represented. On the other hand, the time necessary to process the call request is not explicitly represented either. This period could be inferred from: 1. the time a call request is present in queue 1 actively generating tasks; 2. after a call attempt has left queue 1, the time it takes the processor to process the tasks generated by that call attempt.

   Notice that because of the memory in queue 2 the second period is sensitive to overload conditions. Thus in situations close to or in overload, the intensity of the arrival process of tasks is momentarily larger than the intensity of the server process of queue 2. Then queue 2 will increase rapidly and cause the waiting time necessary to process the tasks of a customer to grow too.

   The question is then how to refine the model such that the above mentioned time periods are exhibited explicitly?

4. In the hierarchical queueing system it is assumed that customers that have been refused access will not return. This is unrealistic. A fraction of customers will attempt to redial after some time. Such repeated call requests will be termed *retrials*. Although it is hinted at in [13] that retrials may be modeled, this has not been done yet. Forys [5] argues that retrials can be a very important cause of performance degradation.
5. In the model of [13] it is assumed that the number of calls-in-build-up and the number of tasks can be measured and used for control. In most exchanges this is not possible. In general one can observe only the number of calls-in-build-up and the idle time of the processor. The last measurement is not relevant for overload conditions. The full information case, in which one assumes knowledge of the past of all processes, is useful for theoretical analysis only. The ultimate goal is the partial information case, in which only practically available measurements are used. Solution of that problem will involve the solution of a filtering problem.

Based on the preceding comments, two new models are introduced in the next section. The aim is to represent all phenomena which cause the performance degradation under overload, while keeping the model analytically tractible.

3. Refined models

In this section two new models are proposed for the processor load in an SPC exchange. They differ from the hierarchical queueing system of section 2 in that a call request may be in an active or in a passive phase. In addition, there is an equation for the process of successfully processed call requests. In the first model retrials will be modeled.

Model 1

See figure 2. for the interconnections of the network of model 1.

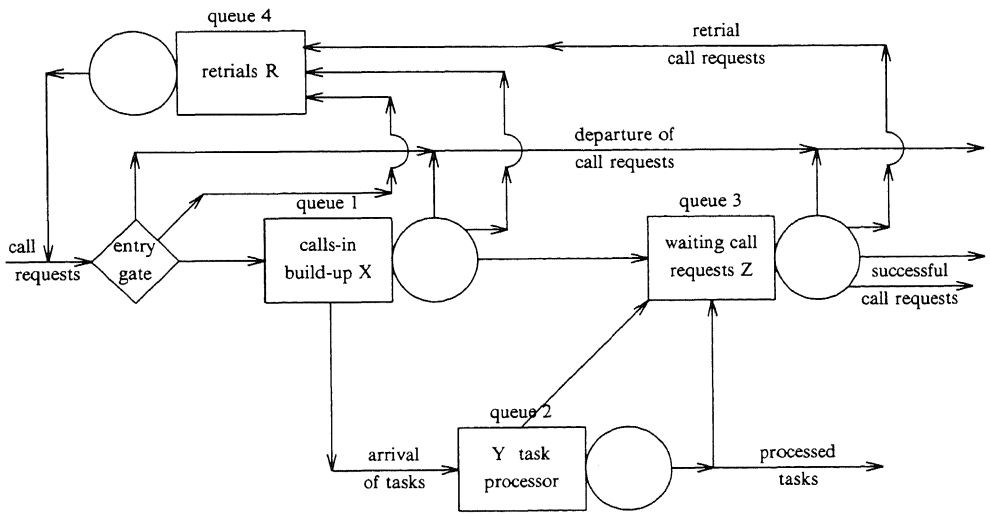

FIGURE 2. A refined model for overload control (model 1).

Assume given a complete probability space and a time index set $T = R_+$. An arrival process is denoted by A and a departure process by D. Super indices will be used to relate such processes to a queue. From now on all stochastic processes denoted by M with a certain index will denote local martingales.

The entry gate. The arrival process of original call requests is assumed to be a Poisson process with intensity λ_0 and with representation,

$$dA(t) = \lambda_0\, dt + dM_1(t), \quad A(0). \tag{3.1}$$

As in the hierarchical queueing system of section 2 access to the exchange is controlled. The arrival process of queue 1 with the calls-in-build-up will be assumed to have the representation,

$$A^X(t) = \sum_{k=1}^{\infty} \bar{U}(k) I_{(\tau_k(A+D^R) \leq t)}, \tag{3.2}$$

where $\tau_k(A+D^R)$ is the stopping time of the k-th arrival of the process $(A+D^R)$ and $\bar{U}:\Omega \times N \to \{0,1\}$ is a random sequence that represents whether a call request is admitted or not. As in [13] one can then show existence of a stochastic process $U:\Omega \times T \to [0,1]$ such that A^X has the representation

$$dA^X(t) = [\lambda_0 + \mu_4 R(t)]\, U(t)\, dt + dM_2(t), \quad A^X(0). \tag{3.3}$$

Furthermore, as in [13], one can reformulate the model such that one only works with a control process U from some class $\underline{U}$ to be specified later. From such a control process U one can deduce the sequence $\bar{U}$ of (3.2).

Thus a fraction $U(.)$ of customers is admitted, and a fraction $(1-U(.))$ is not admitted. Of the latter a fraction r_0 is assumed to redial after some time. This process will be modeled in the retrial queue, see below. There also the variable R will be defined.

The calls-in-build-up. A call request present in queue 1 will be termed a call-in-build-up. Its presence there signifies that it is actively generating tasks that have to be executed by the processor. It will be assumed that a call request is present at queue 1 for an exponentially distributed time with mean μ_1^{-1}. As in the hierarchical queueing system queue 1 will be taken to be a $./M/\infty$ queue, thus with an infinite number of servers. The departure process from queue 1 is then represented by,

$$dD^X(t) = \mu_1 X(t) dt + dM_3(t), \quad D^X(0). \tag{3.4}$$

The equation for the number of calls-in-build-up $X:\Omega \times T \to R_+$ is,

$$X(t) = X(0) + A^X(t) - D^X(t), \tag{3.5}$$

$$dX(t) = [(\lambda_0 + \mu_4 R(t))\, U(t) - \mu_1 X(t)] dt + dM_4(t), \quad X(0). \tag{3.6}$$

The tasks. During the presence of a call request in queue 1 it is assumed to generate tasks. The process of task generation will be assumed to be a Poisson process for each customer. The arrival rate at queue 2 is thus proportional to the number of calls-in-build-up. The representation of the arrival process of queue 2 is,

$$dA^Y(t) = \lambda_1 X(t) dt + dM_5(t), \quad A^Y(0). \tag{3.7}$$

The service times at queue 2 are independent and exponentially distributed with mean μ_2^{-1}. Tasks are executed in the order of their arrival.

$$dD^Y(t) = \mu_2 I_{(Y(t)>0)} dt + dM_6(t), \quad D^Y(0), \tag{3.8}$$

$$Y(t) = Y(0) + A^Y(t) - D^Y(t), \tag{3.9}$$

$$dY(t) = [\lambda_1 X(t) - \mu_2 I_{(Y(t)>0)}]dt + dM_7(t),\ Y(0). \tag{3.10}$$

Call requests waiting for the processing of their tasks. As mentioned in section 2, the presence of a call request in queue 1 represents the active task generation phase of the call request. However, after the active task generation phase there will be a period in which the customer has to wait for the processing of his tasks. This waiting time will be modeled by queue 3.

Only a fraction w_1 of the customers are assumed to be still waiting after their last task has been generated. The remaining $(1-w_1)$ fraction of customers is assumed to have departed. Of this a fraction r_1 goes to the retrial queue.

Let for $k \in Z_+$, $Z(k,.):\Omega\times T\to R_+$ be the number of tasks that have to be processed before the last task of the k-th customer leaving queue 1 is completed. The arrival process of $Z(k,.)$ is then taken to be,

$$A^Z(k,t) = W_1(k) I_{(\tau_k(D^X)\leq t)} Y_{\tau_k(D^X)-}, \tag{3.11}$$

where $\tau_k(D^X)$ is the stopping time at which the k-th customer departs from queue 1 and $W_1:\Omega\times N\to\{0,1\}$ is a sequence of independent random variables that determines whether a customer is still waiting or not. Assume that $P(W_1(k)=1)=w_1$ and that W_1 is independent of all other processes. The expression (3.11) is an approximation of the true waiting time for several reasons. For example, because it starts when the k-th customer leaves queue 1 rather than at the time this customer generates his last task. The departure process for $Z(k,.)$ must then be,

$$D^Z(k,t) = \sum_{s\leq t} I_{(Z(k,s-)>0)} \Delta D^Y(s), \tag{3.12}$$

$$Z(k,t) = Z(k,0) + A^Z(k,t) - D^Z(k,t), \tag{3.13}$$

$$dZ(k,t) = [\mu_1 w_1 X(t) Y(t) I_{(D^X(t)=k-1)} - \mu_2 I_{(Y(t)>0)} I_{(Z(k,t)>0)}]dt + dM_8(t),\ Z(k,0). \tag{3.14}$$

Summarizing, $Z(k,.)$ jumps to the value Y_{τ_k-} at $\tau_k(D^X)$, and subsequently jumps by -1 each time D^Y jumps by $+1$ until it becomes zero.

The patience of customers. Queue 3 will also model the patience of customers in waiting for the processing of their tasks. The total processing time of a customer consists of the time his call request generates tasks, which includes his dial time, and his waiting time after the generation of the last task. The task generation time is exponentially distributed by the assumptions for queue 1. This time does not depend on the state of the network, in particular not on overload conditions. It will be assumed that the waiting time of the customer after the last task in generated, is also exponentially distributed with mean μ_3^{-1}.

In accordance with the assumptions stated above concerning the waiting time of a customer after having left queue 1, one has the following representation. Here $P(k,t)=1$ represents that a customer is waiting and $P(k,t)=0$ that he is not waiting. For the k-th customer leaving queue 1,

$$P(k,t) = P(k,0) + A^P(k,t) - D^P(k,t), \tag{3.15}$$

$$A^P(k,t) = W_1(k) I_{(\tau_k(D^X)\leq t)}, \tag{3.16}$$

where $\tau_k(D^X)$ and W_1 are as defined below (3.11),

$$dA^P(k,t) = w_1\mu_1 X(t) I_{(D^X(t)=k-1)}dt + dM_9(t),\ A^P(k,0), \tag{3.17}$$

$$dD^P(k,t) = \mu_3 I_{(P(k,t)>0)}dt + dM_{10}(t),\ D^P(k,0), \tag{3.18}$$

$$dP(k,t) = [w_1\mu_1 X(t) I_{(D^X(t)=k-1)} - \mu_3 I_{(P(k,t)>0)}]dt + dM_{11}(t),\ P(k,0). \tag{3.19}$$

The successfully processed call requests. The call request of the k-th customer leaving queue 1 is successful if the processing of his last task is finished before his patience has run out. The successfully processed call requests may then be modeled by,

$$D^S(k,t) = \sum_{s \leq t} I_{(P(k,s-)>0)} I_{(Z(k,s-)=1)} \Delta D^Z(k,s), \tag{3.20}$$

$$= \sum_{s \leq t} I_{(P(k,s-)>0)} I_{((Z(k,s-)=1)} \Delta D^Y(s),$$

$$dD^S(t) = \sum_{k=1}^{\infty} dD^S(k,t) \tag{3.21}$$

$$= [\sum_{k=1}^{\infty} I_{(P(k,t)>0)} I_{(Y(t)>0)} I_{(Z(k,t)=1)}] \mu_2 dt + dM_{12}(t), \quad D^S(0).$$

Retrials. Customers with a call request may be turned away by the exchange or loose their patience and terminate the call request. In the model these cases are represented by: 1. the call requests that have been refused access to the exchange by the entry gate; 2. the call requests that have been terminated by customers that are in the active task generation phase of queue 1; 3. the call requests that are unsuccessful because the customer's patience has run out before his last task has been processed. It is assumed that of the customers that have been turned away or that lost their patience, a fraction attempts to redial after an exponentially distributed time with mean μ_4^{-1}. In the model this will be represented by queue 4 that is in principle $./M/\infty$, with an infinite number of servers. A call request present in queue 4 will be termed to be in the *retrial mode.* The variable R represents the number of call requests that are in the retrial mode. The independent random sequences $Q_1, Q_3 : \Omega \times N \to \{0,1\}$ represent whether a call request goes to the retrial mode, if $Q_1(k)=1$, or not, if $Q_1(k)=0$. Assume that $P(\{Q_1(k)=1\})=r_1$, and $P(\{Q_3(k)=1\})=r_3$ and that the sequences Q_1, Q_2 are independent and independent of all other processes.

The process of retrials can then be modeled as,

$$dA^{R0}(t) = [\lambda_0 + \mu_4 R(t)] r_0 (1-U(t)) dt + dM_{13}(t), \quad A^{R0}(0), \tag{3.22}$$

$$A^{R1}(t) = \sum_{k=1}^{\infty} Q_1(k)(1-W_1(k)) I_{(\tau_k(D^X) \leq t)}, \tag{3.23}$$

$$dA^{R1}(t) = r_1(1-w_1)\mu_1 X(t) dt + dM_{14}(t), \quad A^{R1}(0), \tag{3.24}$$

$$A^{R3}(t) = \sum_{k=1}^{\infty} Q_3(k) I_{(\tau_k(D^{SN}) \leq t)}, \tag{3.25}$$

where $\tau_k(D^{SN})$ is the stopping time of the k-th jump of the process D^{SN}, which process counts the number of call requests that leave queue 3 unsuccessfully,

$$D^{SN}(t) = \sum_{s \leq t} I_{(P(k,s-)=0)} I_{(Z(k,s-)=1)} \Delta D^Z(k,s),$$

$$dA^{R3}(t) = \sum_{k=1}^{\infty} r_3 \mu_2 I_{(P(k,t)=0)} I_{(Z(k,t)=1)} I_{(Y(t)>0)} dt + dM_{15}(t), \quad A^{R3}(0), \tag{3.26}$$

$$A^R(t) = A^{R0}(t) + A^{R1}(t) + A^{R3}(t), \tag{3.27}$$

$$dA^R(t) = [r_0(1-U(t))(\lambda_0 + \mu_4 R(t)) + r_1(1-w_1)\mu_1 X(t), \tag{3.28}$$

$$+ \sum_{k=1}^{\infty} r_3 \mu_2 I_{(P(k,t)=0)} I_{(Z(k,t)=1)} I_{(Y(t)>0)}] dt + dM_{16}(t), \quad A^R(0),$$

$$dD^R(t) = \mu_4 R(t) dt + dM_{17}(t), \quad D^R(0), \tag{3.29}$$

$$R(t) = R(0) + A^R(t) - D^R(t), \tag{3.30}$$

$$dR(t) = [r_0(1-U(t))(\lambda_0 + \mu_4 R(t)) + r_1(1-w_1)\mu_1 X(t) + \sum_{k=1}^{\infty}(\mu_2 r_3 I_{(P(k,t)=0)} I_{(Z(k,t)=1)} I_{(Y(t)>0)}) - \mu_4 R(t)]dt + dM_{18}(t), \quad R(0). \tag{3.31}$$

The final stochastic dynamic system consists then of the formula's (3.6,3.10,3.11,3.14,3.16,3.19,3.31) with as controlled variable the successful departure process specified by (3.21). The specification of the stochastic control system is then completed by the definition of a class of admissable controls.

Model 2

Although model 1 answers the criticism of and comments on the hierarchical queueing model of section 2, it is rather complicated. Therefore a simplified model will be proposed below. Model 2 differs from model 1 in that the queues for the waiting call requests are aggregated to just one queue in which the distinction between customers disappears. Moreover, retrials are not modeled. See figure 3 for the network of model 2.

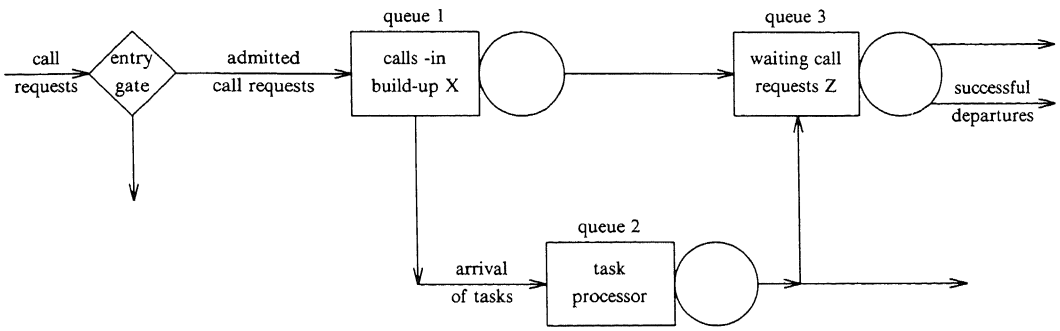

FIGURE 3. Another refined model for overload control (model 2).

Because part of model 2 is identical to model 1, those equations are not duplicated here. This concerns the entry gate, the buffer with calls-in-build-up as modeled by queue 1 and the task processor as modeled by queue 2, with the equations (3.1,3.3,3.4,3.6,3.7,3.8,3.10) with $R=0$.

The process of last tasks. It will be assumed that of every task finished by the processor, thus of $D^Y(t)$, it is a last task of some customer with a certain probability. This is modeled by a random variable $Q(t)$, with $P(\{Q(t)=1\})=c_2$ taken to be the proportion of last tasks over the total number of tasks, here $c_2=\lambda_2/\mu_1$. A disadvantage of this model is that it does not follow the short term fluctuations of the number of calls-in-build-up. The advantage of this model is that it is simple.

$$D^{YL}(t) = \sum_{k=1}^{\infty} Q_1(k) I_{(\tau_k(D^Y)\leq t)}, \tag{3.32}$$

$$dD^{YL}(t) = c_2\mu_2 I_{(Y(t)>0)}dt + dM_{19}(t), \quad D^{YL}(0). \tag{3.33}$$

Call requests waiting for processing of their tasks. The period which a call request has to wait for the processing of its tasks will be represented by queue 3. If there is a departure from queue 1, then there is an arrival at queue 3. The waiting time of each customer at queue 3 is in principle exponentially distributed with mean μ_3^{-1} and assumed to be independent of those of other customers. There is a departure from queue 3 if the patience of a customer runs out or if the last task of a customer is processed.

$$dA^Z(t) = dD^X(t) = \mu_1 X(t)dt + dM_{20}(t), \quad A^Z(0), \tag{3.34}$$

$$dD^{ZP}(t) = \mu_3 Z(t)dt + dM_{21}(t), \quad D^{ZP}(0), \tag{3.35}$$

$$D^{ZC} = \sum_{s \leqslant t} I_{(D^{YL}(s-) \geqslant D^Z(s-))} I_{(Z(s-)>0)} \Delta D^{YL}(s), \tag{3.36}$$

$$Z(t) = Z(0) + A^Z(t) - D^{ZP}(t) - D^{ZC}(t), \tag{3.37}$$

$$dZ(t) = [\mu_1 X(t) - \mu_3 Z(t) - c_2\mu_2 I_{(D^{YL}(t) \geqslant D^Z(t))} I_{(Y(t)>0)} I_{(Z(t)>0)}]dt + dM_{22}(t), \tag{3.38}$$
$$D^Z(0).$$

Successfully processed call requests. Finally one has to model the process of successfully processed call requests D^S. In the model it is assumed that a processed call request is successful if the number of completed last tasks is larger than or equal to the number of customers that have departed from queue 3,

$$D^S(t) = D^{ZC}(t) = \sum_{s \leqslant t} I_{(D^{YL}(s-) \geqslant D^Z(s-))} I_{(Z(s-)>0)} \Delta D^{YL}(s), \tag{3.39}$$

$$dD^S(t) = c_2\mu_2 I_{(D^{YL}(t) \geqslant D^Z(t))} I_{(Z(t)>0)} I_{(Y(t)>0)} dt + dM_{23}(t), \quad D^S(0). \tag{3.40}$$

$$d(D^{YL}(t) - D^Z)(t) = [c_2\mu_2 I_{(Y(t)>0)} - \mu_3 Z(t) \tag{3.41}$$
$$- c_2\mu_2 I_{(D^{YL}(t) \geqslant D^Z(t))} I_{(Y(t)>0)} I_{(Z(t)>0)}]dt + dM_{24}(t), \quad D^{YL}(0) - D^Z(0).$$

The stochastic control system of model 2 consists then of (3.6,3.10,3.38,3.41) with as controlled variable the process D^S of (3.40). Let $\underline{U}$ be the class of admissable control policies that are measurable functions of the past of the processes in the model. This completes the specification of model 2.

4. Stochastic control

In this section the overload control problem is formulated as a stochastic control problem for model 2.

Problem 4.1. *Given the stochastic dynamic system described by model 2 of section 3 with the time index set* $T = [t_0, t_1]$, *the class of input processes* $\underline{U}$ *and the cost function*

$$J(u) = -E_u[D^S(t_1) - D^S(t_0)] \tag{4.1}$$
$$= -E_u[\int_{t_0}^{t_1} c_2\mu_2 I_{(D^{YL}(t) \geqslant D^Z(t))} I_{(Z(t)>0)} I_{(Y(t)>0)} dt].$$

Determine an optimal control $u^* \in \underline{U}$ *such that* $J(u^*) \leqslant J(u)$, *for all* $u \in \underline{U}$.

Theorem 4.2. *Assume there exists a function* $v: T \times N^4 \to R$ *satisfying the following differential equation,*

$$v(t_1, k_1, \ldots, k_4) = 0,$$
$$dv(t,k)/dt - c_2\mu_2 I_{(k_4 \geqslant 0)}(k) I_{(k_1>0)}(k) I_{k_2>0)}(k) \tag{4.2}$$
$$+ [v(t, k_1 + 1, .) - v(t,k)]\lambda_0 I_{(v(t,k_1+1,.) - v(t,k) < 0)}$$

$$
\begin{aligned}
&+ [v(t,k_1-1,.,k_3+1,.)-v(t,k)]\mu_1 k_1 \\
&+ [v(t,.,k_2+1,.)-v(t,k)]\lambda_1 k_1 \\
&+ [v(t,.,k_2-1,.)-v(t,k)](1-c_2)\mu_2 I_{(k_2>0)}(k) \\
&+ [v(t,.,k_2-1,k_3-1,.)-v(t,k)]c_2\mu_2 I_{(k_4\geq 0)}(k) I_{(k_2>0)} \\
&+ [v(t,.,k_2-1,.,k_4+1)-v(t,k)]\mu_2 c_2 I_{(k_4<0)}(k) I_{(k_2>0)}(k) \\
&+ [v(t,.,k_3-1,k_4-1)-v(t,k)]\mu_3 k_3 \\
&= 0,
\end{aligned}
$$

where

$$
\bar{X}(t) = (X(t), Y(t), Z(t), (D^{YL}(t)-D^{Z}(t))), \tag{4.3}
$$

denotes the state, where $k^T=(k_1,k_2,k_3,k_4)\in N^4$ *denotes values of the state and a dot denotes components of* k *that remain unchanged. Then*

$$
U^*(t) = I_{R^-}(v(t,X(t-)+1,.)-v(t,X(t-),.)) \tag{4.4}
$$

is an optimal control for problem 4.2.

The interpretation of the optimal control law (4.3) is simple. Here $v(t,\bar{X}(t))$ is the estimate of the future cost at time $t\in T$ given the current state $\bar{X}(t)$. Then,

$$
v(t,X(t-)+1,.)-v(t,X(t-),.), \tag{4.5}
$$

is the change in the estimate of the future cost if a customer is admitted. Thus the control law (4.3) is such that a customer is admitted if in doing so the estimate of the future cost is decreased. The optimal control law is of bang-bang type, it takes only the extremal values 0 or 1. A similar result can be obtained for the stochastic control problem for model 1 although the equivalent of (4.2) is more complex.

The proof of 4.2 is a standard application of dynamic programming and therefore omitted. It is analogous to the proof of theorem 4.1 in [13]. In fact the proof is a special case of the following proposition.

PROPOSITION 4.3. *Assume given a stochastic control system with as state process a pure jump process* $X:\Omega\times T\to R^n$. *The jumps can take only a finite number of values, say* $r_1,\ldots,r_m\in R^n$. *Let* X_i *represent the process that consists of the jumps of* X *of height* r_i *only. The intensities of these jumps are assumed to be linear in the control process* U,

$$
dX_i(t) = [\lambda_{1i}(X(t))+\lambda_{2i}(X(t))U(t)]dt + dM(t), \quad X_i(0). \tag{4.6}
$$

Given further a cost function,

$$
J(U) = E_U\Big[\int_{t_0}^{t_1} (c_1(X(s)) + c_2(X(s))U(s))\, ds\Big]. \tag{4.7}
$$

Then the Bellman-Hamilton-Jacobi equation is linear in the control U,

$$
\begin{aligned}
\min_{U(t)\in[0,1]}\Big[& dv(t,X(t))/dt + c_1(X(t)) + c_2(X(t))U(t) \\
& + \sum_{i=1}^{i=m} [v(t,X(t)+r_i)-v(t,X(t))][\lambda_{1i}(X(t))+\lambda_{2i}(X(t))U(t)]\Big],
\end{aligned} \tag{4.8}
$$

and the optimal control law is of bang-bang type,

$$U(t) = I_{R-}(c_2(X(t-) + \sum_{i=1}^{i=m} [v(t,X(t-)+r_i) - v(t,X(t-))]\,\lambda_{2i}(X(t-))). \qquad (4.9)$$

Comments

1. Instead of the stochastic control problem on a finite horizon one may also consider the infinite horizon problem, either for a discounted cost or for an average cost criterion. As in [13], there exists under certain conditions a time-invariant control law. Although this has not yet been worked out in detail for model 2 it seems that the control law is again of bang-bang type.
2. The stochastic control problem with partial observations still has to be considered. A realistic assumption is that the number of calls-in-build-up, the waiting call requests and the idle time of the processor and can be observed. This partially observed stochastic control problem leads to a stochastic filtering problem for the state of the control system given the observations. This filtering problem has been solved for the hierarchical queueing system of section 2. There it turns out that the resulting stochastic control system with the filter system is again linear in the control. By proposition 4.3 the optimal control law is thus again of bang-bang type.
3. For the application of control algorithms based on the suggested models and stochastic control, more research is necessary. The authors' research program includes an investigation of time-invariant stochastic control laws for average and discounted cost functions, development of algorithms for the numerical approximation of such control laws and of a performance analysis.

REFERENCES

1. B. BENGTSSON (1982). *On some control problems for queues,* Ph.D. thesis, Linköping University, Linköping.
2. R.K. BOEL, P. VARAIYA (1977). Optimal control of jump processes. *SIAM J. Control Optim. 15,* 92-119.
3. R.K. BOEL (to appear). Modelling, estimation and prediction for jump processes. *Advances in statistical signal processing, volume 1,* JAI Press.
4. P. BRÉMAUD (1981). *Point processes and queues - Martingale dynamics,* Springer-Verlag, Berlin.
5. L.J. FORYS (1983). Performance analysis of a new overload strategy. *10th International Teletraffic Congress (ITC).*
6. R.L. FRANKS, R.W. RISHEL (1973). Overload model of telephone network operation. *Bell System Tech. J. 52,* 1589-1615.
7. B. KARLANDER (1973). Control of central processor load in an SPC system. *Ericsson Technics 30,* 221-243.
8. F.C. SCHOUTE (1981). Optimal control and call acceptance in a SPC exchange. *9th International Teletraffic Congres.*
9. F.C. SCHOUTE (1983). *The technical queue: A model for definition and estimation for processor loading,* Report SR2200-83-3743, Philips Telecommunicatie Industrie, Dept. SAS, Hilversum.
10. F.C. SCHOUTE (1983). Adaptive overload control of an SPC exchange. *10th International Teletraffic Congress.*
11. M. SOBEL (1974). Optimal operation of queues. A.B. CLARKE (ed.). *Mathematical methods in queueing theory,* Lecture Notes in Economics and Mathematical Systems, volume 98, Springer-Verlag, Berlin, 231-261.
12. S. STIDHAM JR., N.U. PRABHU (1974). Optimal control of queueing systems. A.B. CLARKE (ed.). *Mathematical methods in queueing theory,* Lecture Notes in Economics and Mathematical Systems, volume 98, Springer-Verlag, Berlin, 263-294.
13. J.H. VAN SCHUPPEN (1984). *Overload control for an SPC telephone exchange - An optimal stochastic control approach,* Report OS-R8404, Centre for Mathematics and Computer Science, Amsterdam.