SELECTING A SCHEDULING RULE THAT MEETS PRE-SPECIFIED RESPONSE TIME DEMANDS

E. G. Coffman, Jr.
The Pennsylvania State University

I. Mitrani
University of Newcastle upon Tyne, England

In this paper we study the problem of designing scheduling strategies when the demand on the system is known and waiting time requirements are pre-specified. This important synthesis problem has received little attention in the literature, and contrasts with the common analytical approach to the study of computer service systems. This latter approach contributes only indirectly to the problem of finding satisfactory scheduling rules when the desired (or required) response-time performance is specifiable in advance.

Briefly, the model studied assumes a Poisson system with M (priority) classes of jobs. For each class a desired mean waiting time is assumed known in advance. Making use of a well-known conservation law, our main result is a constructive decision procedure for deciding the existence of a preemptive scheduling rule providing the desired waiting time performance, and if one exists, describing one such rule. Our assumptions are discussed and indications are made of how they can be weakened for particular cases.

Key words and phrases: queueing disciplines, synthesis of service disciplines, design of sequencing rules

CR categories: 4.32, 5.5, 8.3

I. Introduction

Broadly speaking, the performance of a computer system is determined by the hardware (its speed, capacity, etc.), the demand (job types, arrival patterns, etc.) and by the scheduling strategy employed (the order in which jobs are executed). A typical study of system performance can be summarized as trying to answer the question: Given the hardware and demand characteristics and the scheduling strategy, what are the values of certain measures of performance? However, while the hardware and demand characteristics can be regarded as fixed in the sense that the system management has little or no control over them, the scheduling strategy is variable. One can serve requests according to FIFO, LIFO, Round-Robin, Preemptive and Non-Preemptive priority, Processor-Sharing and innumerably many other scheduling rules. It is important, therefore, to be able to answer the question: Given the hardware and demand characteristics and the values of certain measures of performance, what scheduling strategy should be employed in order to achieve these values?

Posed like this, the problem has received little attention. A special case in which the choice of strategy was restricted to a family of foreground-background disciplines was studied by Michel and Coffman [1].

This paper deals with systems in which the demand contains jobs of several different types, or classes (all arriving in Poisson streams and having exponentially distributed lengths) and performance is measureo by the average response times for the various job types. The choice of scheduling strategy is, for all practical purposes, unrestricted. Our main result is a theorem giving a necessary and sufficient condition for the existence of a scheduling strategy which satisfies a given performance requirement. It turns out that there is a "universal' family of scheduling strategies such that if a performance requirement can be satisfied at all, it can be satisfied by a strategy from this family. We shall give an algorithm which, given a performance requirement, will either find a scheduling strategy to satisfy it, or will determine that no such strategy exists. Furthermore, the strategies so found are fairly simple and easy to implement.

II. Description of the Model

We think of the computer as a single server giving service to jobs of M different classes. Jobs of class i arrive in a Poisson stream with rate λ_i and have execution times distributed exponentially with parameter μ_i $(i=1,2,\dots,M)$. The traffic intensity for class i is $\rho_i = \lambda_i/\mu_i$ and the total traffic intensity is $\rho = \rho_1 + \dots + \rho_M$. Let the steady-state expected response time (time spent in the system) for jobs of class i be W_i $(i=1,2,\dots,M)$. The system performance will be measured by the vector $\underline{W} = (W_1, W_2, \ldots, W_M)$ whose value depends on the scheduling strategy. The problem, as stated in the introduction, is:

Given a performance requirement $\underline{W}$ and the parameters λ_i and μ_i ($i=1,2,\ldots,M$), determine whether $\underline{W}$ can be realized and if so, find a scheduling strategy that realizes it.

We shall narrow slightly the scope of this problem by excluding from consideration (a), strategies which allow the server to be idle while there are jobs in the system and (b), strategies which make use of, and influence, the remaining processing time of jobs in the system. (For example, the Shortest-Remaining-Processing-Time discipline will not be considered as a candidate for realizing $\underline{W}$). The reasons for (a) are obvious; those for (b) will become apparent shortly. We can say at this point, however, that a real-life strategy is very unlikely to be of the type mentioned in (b) because job execution times are not usually known in advance.

For any strategy which is not of type (a) or (b), the following relation holds:

$$\sum_{i=1}^{M} \rho_i W_i = \frac{V}{1-\rho} \quad \text{where} \quad V = \frac{1}{2} \sum_{i=1}^{M} (\lambda_i/\mu_i^2). \qquad (1)$$

This is known as Kleinrock's conservation law. Its proof is not difficult and we shall outline it here for completeness (see [2]).

When jobs arrive in Poisson streams, the expected load on the system (the sum of the remaining processing times of all jobs in the system) at a random point of time in the steady-state is a constant which does not depend on the scheduling strategy, provided that strategies of type (a) are not allowed. The value of that constant must be $V/(1-\rho)$ as can be verified from the FIFO strategy. On the other hand, if the strategy is not of type (b), the expected remaining processing time of any class i job in the system is $1/\mu_i$ regardless of how much service it has received already (the memoryless property of the exponential distribution). The average number of class i jobs in the system is, according to Little's theorem, $\lambda_i W_i$. Hence the expected load on the system is equal to $\rho_1 W_1 + \rho_2 W_2 + \cdots + \rho_M W_M$. Comparing the two expressions gives (1).

Hereafter we shall talk simply of 'scheduling strategies' meaning strategies not of type (a) or (b). Performance vectors whose elements satisfy (1) will be called 'feasible'. Being feasible is clearly a necessary but not a sufficient condition for a performance vector to be achievable by some scheduling strategy.

Note that the conservation law implies, for instance, that a scheduling strategy which realizes M-1 of the elements of a feasible performance vector must also realize the Mth element. If M=1 (just one job class) then all scheduling strategies yield the same average response time.

III. Results

Our aim now is twofold: first, to find a necessary and sufficient condition for a performance vector to be achievable by some scheduling strategy and second, to devise an effective algorithm which will check the condition and determine the strategy. Before proceeding with the general case, however, we shall illustrate our ideas on the special case when there are only two job classes.

Special case: M=2. Consider the average response time of class 1 jobs, W_1, in the steady-state. The lowest value that W_1 can possibly take, call it W_1^{min}, is achieved when class 1 jobs do not suffer any delays due to the presence in the system of class 2 jobs, i.e., when class 1 jobs have preemptive priority over class 2 jobs. If they do have preemptive priority, the scheduling rule for class 1 jobs among themselves does not matter, so one might as well choose FIFO. Thus the classic preemptive priority discipline results in W_1 reaching its lowest possible value, W_1^{min}. Because of the conservation law, it also results in W_2 reaching its highest possible value, W_2^{max}.

Similarly, the discipline which gives class 2 jobs preemptive priority over class 1 jobs (and FIFO among themselves) results in W_2 reaching its lowest possible value, W_2^{min}, and W_1 reaching its highest possible value, W_1^{max}.

If we think of performance vectors as points on the $W_1 \times W_2$ plane, the feasible vectors lie on a straight line through the two points (W_1^{min}, W_2^{max}) and (W_1^{max}, W_2^{min}). Denote these points by $P(1,2)$ and $P(2,1)$ respectively ('1,2' and '2,1' refer to the priority ordering). Let H be the line segment between $P(1,2)$ and $P(2,1)$: $\underline{W} \in H$ if, and only if, it can be expressed as

$$\underline{W} = \alpha P(1,2) + (1-\alpha)P(2,1); \; 0 \le \alpha \le 1. \qquad (2)$$

In order that a performance vector $\underline{W}$ be realizable by some scheduling strategy it is necessary and sufficient that $\underline{W} \in H$. The necessity of the condition is evident: if $\underline{W} \notin H$ then either $\underline{W}$ is not feasible or it is feasible but at least one of the inequalities

$$W_i^{min} \le W_i \le W_i^{max} \; ; \; i=1,2$$

is violated. In neither case can there be a strategy realizing $\underline{W}$. To show the sufficiency, consider the following family of scheduling strategies.

Divide time into consecutive intervals labeled I_1 and I_2 alternately and let class 1 jobs have preemptive priority during I_1-intervals, while class 2 jobs are given preemptive priority during I_2-intervals. Suppose that the average lengths L_1 and L_2 of the I_1 and I_2-intervals are allowed to increase indefinitely, keeping their ratio constant.

Then, in the long run, the system will be able to reach steady-state within each interval, so that the value of $\underline{W}$ will be $P(1,2)$ during I_1-intervals and $P(2,1)$ during I_2-intervals. Hence the performance vector of this 'mixed' strategy is given by $\underline{W} = \alpha P(1,2) + (1-\alpha)P(2,1)$, where α is the proportion of time occupied by I_1-intervals: $\alpha = L_1/(L_1 + L_2)$. Clearly, every $\underline{W} \in H$ can be realized by a strategy of this type.

Thus, given a performance vector $\underline{W}$, we attempt to solve (2) with respect to α. If there is no solution, or if the solution lies outside the range $0 \le \alpha \le 1$, $\underline{W}$ is impossible to realize. Otherwise the obtained value of α can be used to construct a mixed strategy realizing $\underline{W}$.

It should be pointed out that the above scheduling strategies, with their ever increasing swings from one discipline to another, are not very suitable for practical applications. In practice one would choose some finite average lengths for the I_1 and I_2-intervals. For every L_1 it is possible to find L_2 so as to realize any $\underline{W} \in H$. Solving (2) and using $L_2 = (1-\alpha)L_1/\alpha$ gives a good approximation if L_1 is large compared to the average interarrival and execution times. An exact solution is much more difficult to obtain. The model with finite values for L_1 and L_2 (assuming that L_1 and L_2 are distributed exponentially) is analyzed in [4].

At the other extreme, when L_1 and L_2 are allowed to shrink to zero (keeping their ratio constant), we have a family of scheduling strategies where the processor is shared between the top class 1 job and the top class 2 job in the system in a given proportion. Again every $\underline{W} \in H$ can be realized by a strategy from the family.

In choosing L_1 and L_2 one should consider the trade-off between switching overhead (when they are small) and high variance of $\underline{W}$ (when they are large). This is a difficult problem and we have no ready solution for it.

General case: $M > 2$. Here too, the preemptive priority disciplines play a special role. There are $M!$ such disciplines, one for each assignment of priorities to the M job classes. Denote by $P(1,2,\dots,M),\dots,P(M,M-1,\dots,1)$ the performance vectors of these strategies (listing the priority permutations in lexicographical order, say).

The feasible performance vectors lie on an M-dimensional hyperplane which passes through the points $P(1,2,\dots,M),\dots,P(M,M-1,\dots,1)$. Let H be the convex hull defined by these points: $\underline{W} \in H$ if, and only if, there exist M points $P_1, P_2, \dots, P_M$ from the set $P(1,2,\dots,M),\dots,P(M,M-1,\dots,1)$ and M numbers $\alpha_1,\alpha_2,\dots,\alpha_M$ satisfying $\alpha_i \ge 0$ $(i=1,2,\dots,M)$ and $\alpha_1 + \alpha_2 + \dots + \alpha_M = 1$, such that

$$\underline{W} = \sum_{i=1}^{M} \alpha_i P_i. \quad (3)$$

Denote by H^* the set of all $\underline{W}$ which are the performance vectors of scheduling strategies. We have the following result.

Theorem $H^* \equiv H$. In other words, a performance vector $\underline{W}$ is achievable by some scheduling strategy if and only if $\underline{W} \in H$.

Proof: We shall demonstrate that (1) H^* is a convex set, (2) the performance vectors of all preemptive priority disciplines are extreme points in H^* and (3) H^* has no other extreme points. Since every convex set is the convex hull of its extreme points, (1), (2) and (3) imply that H^* is the convex hull defined by $P(1,2,\dots,M),\dots,P(M,M-1,\dots,1)$; hence H^* and H are the same set.

Claim (1). We have to show that if $\underline{W}_1 \in H^*$ and $\underline{W}_2 \in H^*$ then $\alpha\underline{W}_1 + (1-\alpha)W_2 \in H^*$ for all values of α satisfying $0 \le \alpha \le 1$.

Proof: Since $\underline{W}_1$ and $\underline{W}_2$ belong to H^*, there exist strategies S_1 and S_2 which realize them. Then, using the procedure described in the last section, we can construct a mixed strategy S (consisting of operating S_1 and S_2 alternately) whose performance vector is $\alpha\underline{W}_1 + (1-\alpha)\underline{W}_2$, where α is the proportion of time that S_1 is operated. Hence $\alpha W_1 + (1-\alpha)W_2 \in H^*$ for all $0 \le \alpha \le 1$. □

Claim (2). This property says that none of the points $P(1,2,\dots,M),\dots,P(M,M-1,\dots,1)$ can be expressed in the form $\alpha\underline{W}_1 + (1-\alpha)\underline{W}_2$, where $0 < \alpha < 1$, $\underline{W}_1 \in H^*$, $\underline{W}_2 \in H^*$ and $\underline{W}_1 \ne \underline{W}_2$.

Proof: Suppose that $P(1,2,\dots,M)$, for example (the argument applies equally well to all other points), can be expressed in the above form. Let S_1 and S_2 be the strategies which realize $\underline{W}_1$ and $\underline{W}_2$. There exists a mixed strategy S which uses S_1 and S_2 alternately and whose performance vector is $\alpha\underline{W}_1 + (1-\alpha)\underline{W}_2$, i.e., exactly the same as $P(1,2,\dots,M)$. S, being equivalent to the preemptive priority discipline which gives class 1 jobs top priority, produces the lowest average response time for class 1 jobs that they can possibly have. This means that both S_1 and S_2 give class 1 jobs this lowest possible average response time, i.e., they both give class 1 jobs top preemptive priority. Given that this is the case, S gives class 2 jobs the lowest average response time they can possibly have, which means that both S_1 and S_2 give class 2 jobs second top preemptive priority. Repeating this argument $M-2$ more times we see that $\underline{W}_1 = \underline{W}_2 = P(1,2,\dots,M)$. □

Claim (3). Here we have to prove that any point $\underline{W}$ which is not one of $P(1,2,\dots,M),\dots,P(M,M-1,\dots,1)$ can be expressed in the form $\underline{W} = \alpha\underline{W}_1 + (1-\alpha)\underline{W}_2$ where $0 \le \alpha \le 1$, $\underline{W}_1 \in H^*$, $\underline{W}_2 \in H^*$ and $\underline{W}_1 \ne \underline{W}_2$.

Proof: Let S be the strategy which realizes $\underline{W} = (W_1, W_2, \dots, W_M)$. Since S is not one of the preemptive priority disciplines, there is at least one pair of job classes, say class 1 and class 2, such that class 1 jobs may be present in the system when a class 2 job is being served and class 2 jobs may be present in the system when a class 1 job is being served. Construct a scheduling strategy S_1 which follows the actions of S exactly, except for the instances of time when S would have served a class 1 or a class 2 job. At those instances S_1 works according to the following algorithm: if S would have served a class 1 job and there are class 2 jobs in the system, serve a class 2 job instead; if S would have served a class 2 job and there is not one in the system then serve one of the class 1 jobs that S would have served earlier; otherwise do as S would have done. It is easy to see that S_1 devotes precisely the same periods of time to the service of class 1 and class 2 jobs as S; only within those periods S_1 gives class 2 jobs preemptive priority over class 1 jobs, whereas S does not. Since the actions of S_1 with respect to jobs of classes $3,4,\ldots,M$ are the same as those of S and occur at exactly the same moments of time, the average response times for classes $3,4,\ldots,M$ are the same under both strategies. However, class 1 jobs wait more, on the average, under S_1 than under S, while class 2 jobs wait less. Thus if $\underline{W}_1 = (W_1^{(1)}, W_2^{(1)}, \ldots, W_M^{(1)})$ is the performance vector of S_1, we have $W_1^{(1)} > W_1$, $W_2^{(1)} < W_2$, $W_i^{(1)} = W_i$ $(i=3,4,\ldots,M)$.

In a similar fashion, a strategy S_2 with performance vector $\underline{W}_2 = (W_1^{(2)}, W_2^{(2)}, \ldots, W_M^{(2)})$ can be constructed, such that $W_1^{(2)} < W_1$, $W_2^{(2)} > W_2$, $W_i^{(2)} = W_i$ $(i=3,4,\ldots,M)$.

Let S' be a mixed strategy consisting of operating S_1 and S_2 alternately for long intervals of time. The performance vector $\underline{W}' = (W_1', W_2', \ldots, W_M')$ of S' is given by $\underline{W}' = \alpha\underline{W}_1 + (1-\alpha)\underline{W}_2$ where α is the proportion of time that S_1 is operated. Clearly $W_i' = W_i$ $(i=3,4,\ldots,M)$ for all values of α. Furthermore, $W_1' < W_1$ when $\alpha = 0$ and $W_1' > W_1$ when $\alpha = 1$; therefore $W_1' = W_1$ for some $0 < \alpha < 1$. For that value of α we must have also $W_2' = W_2$ because of the conservation law. $\underline{W}$ has now been expressed in the desired form, with $\underline{W}_1 \in H^*$, $\underline{W}_2 \in H^*$ and $\underline{W}_1 \neq \underline{W}_2$. □

This completes the proof of the theorem. We have established that every achievable performance vector is of the type (3) and can therefore be realized by a mixed scheduling strategy consisting of operating in rotation the M preemptive priority disciplines whose performance vectors appear in the right-hand side of (3). The problem now is, given the performance vector, how do we find these disciplines? Tyring all combinations of M out of the $M!$ points $P(1,2,\ldots,M),\ldots,P(M,M-1,\ldots,1)$ in order to discover whether one of these combinations satisfies (3) is obviously undesirable for arbitrary $m > 3$.

Let us rename the performance vectors of the preemptive priority disciplines $P_1, P_2, \ldots, P_{M!}$ and reformulate the problem as follows: Find $M!$ non-negative numbers $\alpha_1, \alpha_2, \ldots, \alpha_{M!}$, all but M of which are equal to zero, such that

$$\sum_{i=1}^{M!} \alpha_i = 1 \quad \text{and} \quad \sum_{i=1}^{M!} \alpha_i P_i = \underline{W}$$

We have here $M+1$ linear constraints, M of which are independent (the vectors P_i and $\underline{W}$ have only $M-1$ independent elements because of the conservation law), to which we wish to find a non-negative solution such that at most M of the variables are non-zero. This is the well-known 'initial basis' problem in linear programming. It can be solved by introducing M artificial variables β_0 and $\underline{\beta} = (\beta_1, \beta_2, \ldots, \beta_{M-1})$ and solving the linear program

$$\min \left(\beta_0 + \sum_{i=1}^{M-1} \beta_i\right) \tag{4}$$

subject to the constraints

$$\beta_0 + \sum_{i=1}^{M!} \alpha_i = 1 \quad \text{and} \quad \underline{\beta} + \sum_{i=1}^{M!} \alpha_i P_i = \underline{W}$$

(using only the first $M-1$ elements of P_i and $\underline{W}$). An initial basis for (4) is obtained by setting $\alpha_i = 0$ $(i=1,2,\ldots,M!)$, $\beta_0 = 1$, $\underline{\beta} = \underline{W}$. If in the solution to (4) we have $\beta_0 = 0$ and $\underline{\beta} = 0$ (which means that M of the α_i are non-zero) then an expression of $\underline{W}$ in the form (3) has been found. Otherwise we conclude that $\underline{W} \notin H$ and therefore it cannot be realized at all. The procedure for solving (4) should be slightly different from the normal simplex method because of the 'very long and narrow' shape of the constraint matrix. Instead of keeping the whole matrix in storage, one would prefer to generate its columns one at the time and, one would hope to find a solution without needing all $M!+1$ of them. For example, when trying to decide which column should be replaced in the basis, one can stop searching as soon as the first column which reduces the object function has been found, rather than look for the column which yields the biggest reduction. If such a policy is adopted, the order in which the priority disciplines are listed becomes important. It seems a good idea to put $P(1,2,\ldots,M)$, $P(2,3,\ldots,M,1),\ldots,P(M,1,2,\ldots,M-1)$ at the beginning of the list because these performance vectors correspond to strategies where every job class is given every priority, and it is not unlikely that they alone will provide a solution.

The remarks made at the end of the previous section apply here too. Having determined the M priority disciplines from which the mixed strategy realizing $\underline{W}$ is to be constructed, these will in practice be operated in rotation for finite intervals of time rather than infinitely long ones. A good approximation will be obtained if the lengths of the intervals (calculated from the values of α_i) are large compared with the interarrival and execution times.

Priority Processor-Sharing strategies (Kleinrock [3]) giving jobs of different classes different fractions of the processor can also be used to realize any $\underline{W} \in H$ (this is because the preemptive priority disciplines are limiting cases of these strategies). Unfortunately there are no reliable analytical results for their performance vectors.

IV. Discussion

We have shown that a performance requirement stated in terms of the average response times of M job classes can be satisfied (if it can be satisfied at all) by a mixed scheduling strategy involving not more than M preemptive priority disciplines. These mixed strategies are conceptually very simple and would pose no implementation problems. The difficulty lies in deciding for how long to operate each preemptive priority discipline within the mix so as (a), to avoid excessive switching overheads and large variations in the response times and (b), to achieve good approximation to the required performance. Bearing in mind the complexity of the analytical problem, it seems that the answer in a practical situation would be best obtained by simulation.

Finally, a few words about the assumptions of the model. The most restrictive of these were the Poisson input and the exponential service times assumptions. Both were needed for the sole purpose of ensuring the validity of Kleinrock's conservation law. The results presented here will apply to any situation where some general conservation law of the form

$$f(W_1, W_2, \ldots, W_M) = \text{const} \qquad (1')$$

is in force. In particular, they apply in the case when inputs are Poisson, service times have general distributions and all jobs are served to completion: (1) holds in this case, with a different value for V. Thus if we are prepared to disallow strategies which interrupt the execution of jobs, we can drop the exponential service times restriction. The role of the preemptive priority disciplines will be played by the non-preemptive priority disciplines.

Acknowledgement

We wish to thank H. I. Scoins of Newcastle University for some constructive discussions about the problem. In particular, the linear programming formulation (4) is his.

References

[1] J. Michel and E. G. Coffman, Jr., Synthesis of a Feedback Queueing Discipline for Computer Operation, JACM, 21, 2 (1974), 329-349.

[2] L. Kleinrock, A Conservation Law for a Wide Class of Queueing Disciplines, Nav. Res. Log. Quart., 12, pp. 181-192, 1965.

[3] L. Kleinrock, Time-Shared Systems: A Theoretical Treatment, JACM, 14, pp. 242-261, 1967.

[4] E. G. Coffman, Jr. and I. Mitrani, "The Design of Sequencing Rules to Meet Pre-Specified Response-Time Demands," Technical Report, Computer Science Dept., Pennsylvania State University, 1975.