On Optimal Scheduling Algorithms for Time-Shared Systems

LEONARD KLEINROCK AND ARNE NILSSON

University of California, Los Angeles, California

ABSTRACT The problem of finding those optimum scheduling algorithms for time-shared systems that minimize a cost function that depends on waiting time and required service time is considered An optimality condition which sometimes leads to infeasible algorithms is established The procedure is improved upon by use of a mathematical programming technique but still does not always generate feasible algorithms. These results are used as upper bounds on the performance of known feasible algorithms so that it is possible to evaluate how close to optimal the present algorithms come.

KEY WORDS AND PHRASES time-sharing, scheduling, scheduling algorithms, optimal-scheduling algorithms

CR CATEGORIES 4 32, 4 35, 4 6

1. *Introduction*

During the last decade considerable effort has been put forth in the analysis of various scheduling algorithms for multiaccess time-sharing systems. The dominant performance measure has almost always been the mean waiting time (response time) conditioned on a customer's (job's) service-time requirement. However, little attention has been given to the important problem of defining some appropriate cost function and then of finding that algorithm which is *optimal* with respect to this cost function. Some effort must be made to solve this synthesis problem. The original model for these systems was to view them as *single*-resource systems [6, 15]. In the past few years significant understanding has been gained through *multiple*-resource models using results from queuing networks [16]. As a first attempt to solve the optimization problem referred to above, we study only the case of the single-resource models in this paper.

2. *Definitions*

We view a time-sharing system as a preemptive queuing system served by a single (server) resource—the CPU. The model we choose is that of an M/G/1 queuing system [14]; that is, we assume that a Poisson source generates arrivals to the system

This work was supported by the Advanced Research Projects Agency of the Department of Defense under Contract DAHC 15-73-C-0368

Authors' present addresses L Kleinrock, Computer Science Department, University of California, Los Angeles, CA 90024, A. Nilsson, North Carolina State University, Raleigh, NC 27650

at a rate λ and that the processing-time distribution $B(x)$ is arbitrary (with density-function $b(x)$):

$$P[\text{interarrival time} \leq t \text{ seconds}] = 1 - e^{-\lambda t}, \qquad t \geq 0,$$
$$P[\text{service-requirement} \leq x \text{ seconds}] = B(x).$$

The nth moment of service time is denoted by $\overline{x^n}$.

We define

$w(y|x)$ = the density function for the waiting time of a customer who requires x seconds of service;

$W(x)$ = mean (conditional) waiting time for a customer requiring x seconds of service.

Clearly,

$$W(x) = \int_0^\infty yw(y|x)\,dy. \tag{2.1}$$

Expressions for the density $w(y|x)$ are not so easily obtained except for some rather simple scheduling algorithms. The mean waiting time $W(x)$ has been obtained, however, for a large class of interesting scheduling algorithms [3, 6, 7, 10, 11, 15, 17].

Let us now consider a *cost function* for a time-sharing system. The form we have chosen for the cost function is reasonable but really quite arbitrary, and we stress that there may exist other cost functions that are more realistic for a given application. We have been forced into defining a cost function, since the user population has so far failed to come up with one or a set of such functions. Our assumption is that the expected cost incurred by customers with service time x is a function only of their expected delay and of x. Specifically, we define

$C(W, x)$ = cost incurred by a customer with service time x if the expected delay is W (we assume that $C(W, x)$ is convex in W);

$W = W(x, s)$ = expected delay for a customer with service time x if the scheduling algorithm is s;

$\text{COST}(s)$ = total expected cost of using algorithm s.

Obviously,

$$\text{COST}(s) = \int_0^\infty C(W, x)b(x)\,dx, \tag{2.2}$$

and our objective is to find that scheduling algorithm that minimizes the total expected cost. We restrict ourselves to algorithms that use "no a priori information" [15]. For these algorithms we can relax the problem and minimize over functions $W(x)$.

3. *The Optimization Problem*

Our objective is to minimize the total cost with respect to $W(x)$ or, equivalently, the scheduling algorithm; that is,

$$\min_{W(x)} \int_0^\infty C(W, x)b(x)\,dx. \tag{3.1}$$

$W(x)$ is determined by the scheduling algorithm, which can be any "no a priori information" algorithm; the service time distribution $B(x)$ also affects $W(x)$, but we assume $B(x)$ to be given and not a function which we are permitted to vary in the optimization problem.

It has been shown [10, 15] that the function $W(x)$ (= W for convenience) must obey at least four different constraints, namely,

(i) $\dfrac{dW}{dx} \geq 0$,

(ii) $W \leq W_U$ (upper bound),

(iii) $W \geq W_L$ (lower bound),

(iv) $\displaystyle\int_0^\infty W(x)(1 - B(x))\,dx$ = constant (independent of the scheduling algorithm)

$$= \frac{\rho \overline{x^2}}{2(1-\rho)} \triangleq L,$$

where $W_U \geq W_L \geq 0$.

Condition (i) simply states that $W(x)$ cannot decrease with x. Conditions (ii) and (iii) are tight upper and lower bounds, where

$$W_U(x) = \frac{\lambda \overline{x^2}}{2(1-\rho)(1-\rho_x)} + \frac{x\rho_x}{1-\rho_x},$$

$$W_L(x) = \frac{\lambda \overline{x_x^2}}{2(1-\rho_x)},$$

and

$$\overline{x_x^n} = \int_0^x y^n b(y)\,dy + x^n(1 - B(x)),$$

$$\rho_x = \lambda \overline{x_x}.$$

Condition (iv) is a conservative law, which must be valid for any work-conserving scheduling algorithm (we restrict ourselves to such algorithms).

The optimization problem may now be formulated as

$$\min_W \int_0^\infty C(W, x) b(x)\,dx \tag{3.2}$$

under the conditions (i)–(iv) (where $W(x)$ is varied by varying the scheduling algorithm). We recognize this problem as belonging to the class of calculus-of-variations problems. Unfortunately, it has been impossible to find a computationally efficient mathematical procedure which solves such a problem, excluding mathematical programming, which we report on in Section 5. However, using just one of the conditions (the conservation law), it *is* possible to obtain a relatively simple expression for W that optimizes C. For simplicity we assume that W is a continuous function; unfortunately (for the purists), this means that the large and interesting class of multilevel systems [9, 11] are not included among the algorithms we consider. (However, the selfish scheduling algorithms (SSA) [4, 8, 15], which may use these multilevel systems, will smooth out their discontinuities and are included among our algorithms.) Of greater importance is the observation that since we have kept just one of the four conditions, we may end up with a solution for $W(x)$ which violates some (or all) of the other three conditions.

Yet more annoying is the fact that even if we do find a W which falls into the class of functions defined by (i)–(iv), it *still* may not be feasible, since conditions (i)–(iv) are necessary but not sufficient; further, it may be that we have no idea how to *implement* such a scheduling algorithm even if it can be shown to be feasible.

These problems arise since there may exist other constraints on W unknown to us at this time. That is, we can make use of some known *necessary* conditions on $W(x)$ but are currently unable to state the *necessary and sufficient* conditions on W. However, we make use of the following important observation.

If we optimize over a constraint space which includes some, but not necessarily all, of the constraints, then any solution which we obtain and which is also *realizable* by means of a known algorithm *must* be the true optimum solution (obviously it must satisfy all the feasibility constraints if it is realizable).

Thus the optimization problem may be formulated as

$$\min_{W} \int_0^\infty C(W, x) b(x)\, dx, \tag{3.3}$$

under the constraint

$$\int_0^\infty W(x)(1 - B(x))\, dx = L. \tag{3.4}$$

A straightforward approach (using the Lagrange multiplier technique) gives us the following necessary condition for optimality (note that we have assumed that $C(W, x)$ is convex in $W(x)$):

$$\frac{\partial}{\partial W}\{C(W, x)\} b(x) = k(1 - B(x)), \tag{3.5}$$

where k is a Lagrange multiplier. An optimal $W(x)$, which we denote by W for simplicity, can be derived from this relation, although it may possibly be infeasible, as we have said. We conclude that the result of this optimization procedure is a class of optimal W only some of which are feasible.

In order to proceed with some examples and special cases, we require the specification of the cost function $C(W, x)$. It is difficult to find a generally agreed-upon function of this type, and so we are left in the position of having to invent some. This we do in the following sections as we study some examples and extensions.

4. *Simple Examples*

In this section we demonstrate the above method through some simple, yet important, examples and also show that it is possible to end up in the feasible or infeasible regions.

Example 1. Let us choose

$$C(W, x) = W,$$
$$b(x) = \mu e^{-\mu x}.$$

This is a reasonable but simplistic cost function.

From eq. (3.5) we find that

$$W = \frac{k}{2\mu},$$

and therefore $W(x)$ is independent of x.

One feasible scheduling algorithm which gives us a W independent of x is FCFS (first come first served), and so FCFS is one optimal choice (from among many, in fact, from among all nonpreemptive algorithms which operate independent of the service time).

Example 2. Here we choose

$$C(W, x) = \frac{W}{x + a}, \qquad \text{where} \quad a \geq 0, \quad a \text{ constant,}$$
$$b(x) = \mu e^{-\mu x}.$$

This cost function is fairly reasonable in that it behaves sensibly with regard to W and x.

The optimal W must be (eq. (3.5))

$$W = \frac{k}{2\mu}(x + a).$$

A feasible algorithm which gives us this optimal W is one picked from the SSA family, namely, SRR (selfish round robin) [4, 8, 15].

If, however, $b(x)$ is chosen as

$$b(x) = 2\mu(2\mu x)e^{-2\mu x},$$

the optimal W is

$$W(x) = \frac{k}{4\mu}(x + a)\left(1 + \frac{1}{2\mu x}\right).$$

This $W(x)$ clearly violates constraint (i), $dW/dx > 0$, for small x. This solution is infeasible.

Example 3. Now consider

$$C(W, x) = \frac{W^2}{x},$$
$$b(x) = 2\mu(2\mu x)e^{-2\mu x}.$$

This combination once again gives us that SRR is optimal! On the other hand, with $b(x) = \mu e^{-\mu x}$ we find that the processor-shared round robin (RR) algorithm [2, 6, 15] is optimal. Both of these are feasible, of course.

Example 4. If we let

$$C(W, x) = \frac{W}{x^2 + a},$$
$$b(x) = \mu e^{-\mu x},$$

we get that

$$W(x) = \frac{k}{2\mu}(x^2 + a).$$

Unfortunately, this W violates the upper bound W_U, which requires that $W_U \sim \rho x/(1 - \rho)$ when x is large, and therefore the optimal W we have found is infeasible, since it is proportional to x^2.

Example 5. Finally, consider

$$C(W, x) = W,$$
$$b(x) = 2\mu(2\mu x)e^{-2\mu x}.$$

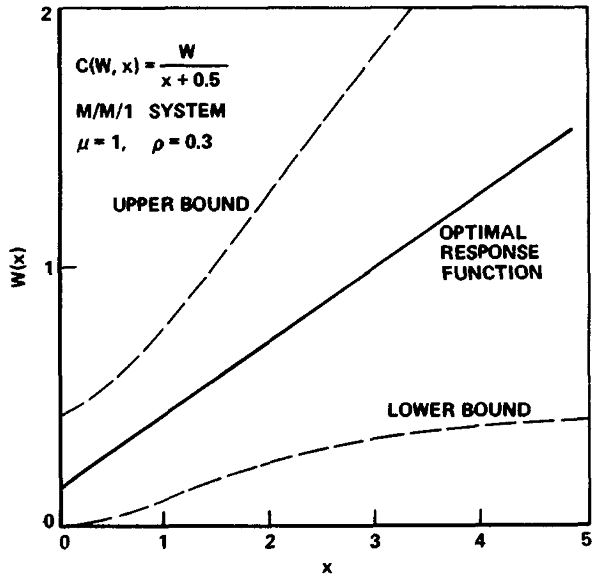

FIG 1 Optimal $W(x)$ for $C(W, x) = W^2/(x + 0.5)$ M/M/1 system, $\mu = 1$, $\rho = 0.3$

This gives us a W of the form

$$W = \frac{k}{4\mu}\left(1 + \frac{1}{2\mu x}\right).$$

The solution is such that $dW/dx < 0$, and since this violates constraint (i), we see that this solution is also infeasible.

5. *Solution by Mathematical Programming*

In Section 3 we showed one way of obtaining an optimal algorithm. The optimization problem formulated in eq. (3.2) under the conditions (i)–(iv) may also be solved by mathematical programming. We have carried out this method; it is a straightforward numerical optimization [5], and we do not describe the specific method here. Instead, we report on the results obtained for some interesting combinations of $C(W, x)$ and $B(x)$.

(1) Let us choose

$$C(W, x) = \frac{W}{x + a},$$
$$b(x) = \mu e^{-\mu x}.$$

This is Example 2 of Section 4, and there we showed that the optimal algorithm is SRR.

In Figure 1 we show the optimal $W(x)$ obtained from the numerical procedure and the upper and lower bounds. Clearly, we recognize this $W(x)$ as the response curve for an SRR algorithm. We knew this beforehand from our analytical optimization method; nevertheless, it serves as a verification of the numerical procedure.

(2) Now choose

$$C(W(x), x) = \frac{W}{x^2 + a},$$
$$b(x) = \mu e^{-\mu x}.$$

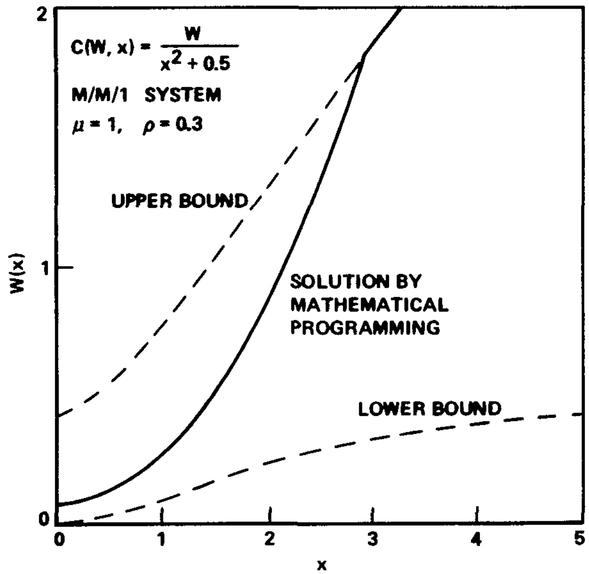

FIG 2 Optimal $W(x)$ for $C(W, x) = W^2/(x^2 + 0.5)$ M/M/1 system, $\mu = 1$, $\rho = 0.3$.

From Section 4 we know that the analytical method produces an infeasible solution. The mathematical programming solution is shown in Figure 2; note that the optimal $W(x)$ has been forced to stay within the permissible region. Unfortunately, this solution is still not feasible, since we know that a realizable $W(x)$ cannot coincide with $W_U(x)$ over a measurable interval. Moreover, we have no direct procedure for implementing that part of $W(x)$ which does lie in the permissible region. Thus we conclude that there must exist other constraints on $W(x)$ which are unknown to us at this time. We can, however, learn quite a lot from the optimal infeasible $W(x)$ as given by the numerical procedure. A good compromise for a scheduling algorithm is an SFB algorithm, and our numerical investigations show that the optimal cost as obtained from the numerical method does not differ much from the cost obtained when an SFB algorithm with suitable parameters is chosen.

(3) Now consider

$$C(w, x) = \frac{W}{x + a},$$
$$b(x) = 2\mu(2\mu x)e^{-2\mu x}.$$

We know that the analytical method generates a $W(x)$ that violates constraint (i), that is, $dW/dx \geq 0$. The optimal solution from the mathematical programming approach is shown in Figure 3 (of course, it will *not* violate the conditions (i)–(iv)); we note that this is a plausible response function, but alas we do not know how to implement it. Once again, however, we may take advantage of the optimal algorithm (yet unknown) and attempt a suitable approximation; indeed, let us choose a multilevel system with two levels, namely, FCFS up to $x = 1.3$, followed by RR [9]. The numerical investigation shows that the optimal cost is 0.214 units and the cost for the multilevel system is 0.218 units. Since the mathematical programming approach yields a solution whose cost must be a *lower bound* on the optimal cost, and since we have guessed a solution which is almost as good as the lower bound, we are encouraged that we must be extremely close to the optimal feasible algorithm.

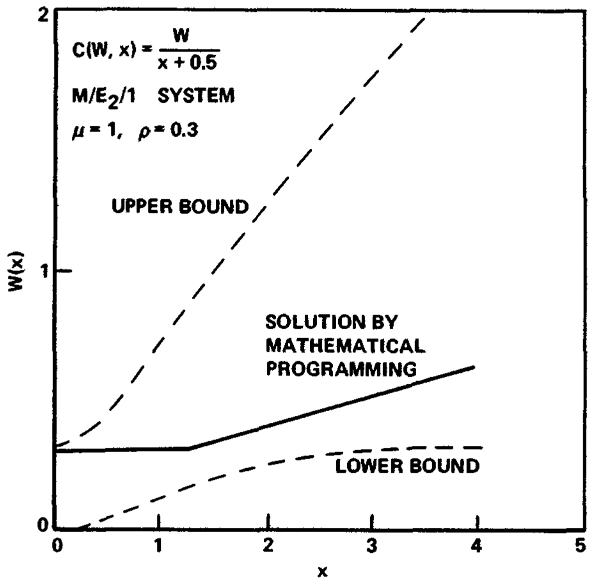

FIG. 3. Optimal $W(x)$ for $C(W, x) = W^2/(x + 0.5)$ $M/E_2/1$ system, $\mu = 1$, $\rho = 0.3$.

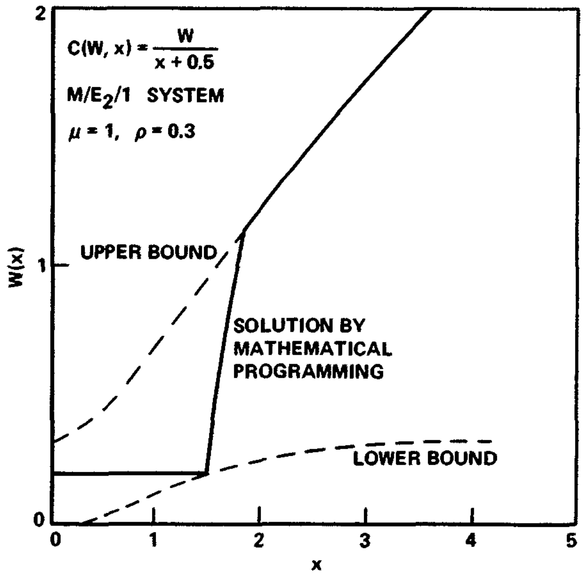

FIG 4. Optimal $W(x)$ for $C(W, x) = W/(x + 0.5)$ $M/E_2/1$ system; $\mu = 1$, $\rho = 0.3$

(4) Finally, consider

$$C(W, x) = \frac{W}{x + a},$$
$$b(x) = 2\mu(2\mu x)e^{-2\mu x}.$$

Once again we get an optimal $W(x)$ which we do not know how to implement (see Figure 4). We also have the same kind of behavior as in Example 2, namely, that the optimal $W(x)$ hits the upper bound and stays there. A good approximation for this optimal algorithm is once again a two-level system with FCFS up to $x = 1.5$, followed by FB.

In summary then, we find that the mathematical programming technique provides us with a means for finding (possibly infeasible) solutions for $W(x)$. These solutions, however, do provide lower bounds on the cost C and do suggest what the optimal feasible solution may be. Moreover, the suggested feasible solutions have costs which often come extremely close to the lower bounds!

6. *Conclusion*

We have presented a new method which optimizes the scheduling algorithm in a time-sharing system. The importance of this work is that it provides some insight into the vast and difficult problem of synthesizing optimal queuing systems and, specifically, single-resource models of time-sharing systems. Unfortunately, we have not been able to formulate a complete optimization problem, since all the *necessary and sufficient* conditions on $W(x)$ are not known to us at this time. Furthermore, some of the conditions known to us are such that they do not easily lend themselves to mathematical optimization. The mathematical programming numerical procedure allows us to overcome some of these difficulties. Unfortunately, it does not settle the problem of finding feasible scheduling algorithms, since, as was shown by the examples in Section 5, we often found an optimal $W(x)$ that we could not implement. However, as has been shown, we managed to derive some useful and interesting results. Our point of view in Section 5 was to use the numerical procedure to suggest an appropriate feasible algorithm whose closeness to optimality we could measure.

Another approach to the problem of infeasible solutions is to "turn the problem around," as follows. Let us begin with a known feasible W (i.e., one which we know how to implement) and find that class of cost functions $C(W, x)$ which are minimized by that W (rather than seeking the W which minimizes a given $C(W, x)$). Examples of this approach may be found in [13].

Other approaches and extensions to this problem would be welcome contributions to the field.

REFERENCES

(Note Reference [12] is not cited in the text)

1 COFFMAN, E.G JR., AND KLEINROCK, L Feedback queueing models for time-shared systems *J ACM* 15, 4 (Oct 1968), 549–576

2 COFFMAN, E.G JR, MUNTZ, R R, AND TROTTER, H Waiting time distribution for processor-sharing systems. *J ACM 17*, 1 (Jan 1970), 123–130

3 ESTRIN, G, AND KLEINROCK, L Measures, models and measurements for time-shared computer utilities. Proc 22nd Nat. Conf of the ACM, Washington, D C, Aug 1967, pp 85–96

4 HSU, J Analysis of a continuum of processor-sharing models for time-shared computer systems Tech Rep UCLA-ENG-7166, University of California, Los Angeles, Calif., Oct. 1971.

5 KARLSSON, J On optimal time-sharing algorithms Master Th, Lund Institute of Technology, Lund, Sweden, 1977.

6 KLEINROCK, L Analysis of a time-shared processor *Nav Res Log Q 11* (March 1964), 59–73

7 KLEINROCK, L Time-shared systems. A theoretical treatment *J ACM 14*, 2 (April 1967), 242–261

8. KLEINROCK, L A continuum of time-sharing scheduling algorithms AFIPS 1970 Spring JCC, AFIPS Press, Arlington, Va., pp. 453–458

9 KLEINROCK, L, AND MUNTZ, R R Multilevel processor-sharing queuing models for time-shared models Proc 6th International Teletraffic Congress, Munich, W Germany, Aug 1970, pp. 341/1–8

10 KLEINROCK, L, MUNTZ, R R, AND HSU, J Tight bounds on the average response time for processor-sharing models of time-shared computer systems Proc IFIP Congress 71, North-Holland Publishing Co., Amsterdam, 1977, pp 124–133

11 KLEINROCK, L A selected menu of analytical results for time-shared computer systems. In *Systemprogrammierung*, R Oldenburg Verlag, Munich, W Germany, 1972, pp. 45–73

12 KLEINROCK, L., AND MUNTZ, R.R. Processor-sharing queueing models of mixed scheduling disciplines for time-shared systems. *J. ACM 19*, 3 (July 1972), 464–482
13 KLEINROCK, L , AND NILSSON, A On optimal scheduling algorithms for time-shared systems. UCLA Computer Systems Modeling and Analysis Note, Univ. of California, Los Angeles, Calif., June 1974.
14. KLEINROCK, L. *Queueing Systems, Volume I: Theory.* John Wiley and Sons, New York, 1975.
15. KLEINROCK, L. *Queueing Systems, Volume II: Computer Applications.* John Wiley and Sons, New York, 1976.
16. REISER, M. Interactive modeling of computer systems. *IBM Syst. J. 15*, 4 (1976), 309–327.
17. SCHRAGE, L E. The queue M/G/1 with feedback to lower priority queues. *Management Sci.* 13 (1967), 466–471.

RECEIVED DECEMBER 1977, REVISED OCTOBER 1979; ACCEPTED MARCH 1980