A CONTINUUM OF COMPUTER PROCESSOR-SHARING QUEUEING MODELS[+]

Leornard Kleinrock
University of California
Los Angeles, California, USA

Jiunn Hsu
Bell Telephone Laboratories
Holmdel, New Jersey, USA

ABSTRACT

This paper is concerned with the response time provided to jobs in a time-shared computer systems environment. The main thrust of this paper is to extend all known results and any future results for such scheduling algorithms so as to create a continuum of response time functions by introducing a single degree of freedom into our model. Thus, given any scheduling algorithm, its performance may be made to vary continuously from its original form to that of the first-come-first-served scheduling algorithm by means of this degree of freedom. The solution for this continuum is simply expressed in terms of the solution for the original scheduling algorithm and the implementation of this degree of freedom in any computer system is extremely simple. The class of queueing systems considered is of the M/G/1 type.

INTRODUCTION

The problem of scheduling many job requests competing for service on the central processing unit (CPU) of a time-shared computer system has been studied for a number of years [1-16]. The major effort has been directed at solving for the expected response time $T(t)$ which is the average time a job spends in the system given that it requires t seconds of service from the CPU. Methods from queueing theory have been useful in studying these highly preemptive scheduling algorithms for allocating time on the CPU among the various competing jobs.

A number of important algorithms have emerged from these studies, notably the round robin (RR) [3,14], the generalized foreground-background (FB) [4,7], and, of course, the common first-come-first-served (FCFS). The object of the scheduling algorithm is to give rapid response times to short jobs (at the expense of the longer jobs) in some "equitable" fashion. The FCFS algorithm provides no discrimination on the basis of job length and therefore occupies one extreme in the set of possible behaviors available from time-sharing systems; this procedure is the simplest form of "batch-processing." The FB algorithm operates as follows: the CPU (service facility) is preempted by that job in the system which has so far received least service. This algorithm occupies the other extreme in its ultimate discrimination in favor of the very shortest jobs and thereby provides the most detrimental behavior to the very longest jobs. Between these two extremes lies the well-known RR system in which all jobs currently in the system share the CPU equally in the sense that when n jobs are competing for service, they each receive service at the rate of $1/n$ seconds per second of elapsed time. The RR system occupies a special place in the study of time-shared scheduling algorithms in that it is "eminently fair" [15]. This refers to the fact that the normalized response time $T(t)/t$ (normalized with respect to a job's required service time) is constant; taking this ratio as a "penalty rate" it is clear that customers are discouraged either from partitioning their jobs into smaller ones or from grouping theirs with others to create larger ones.

The queueing model used in these studies is that of an M/G/1 system with a preemptive scheduling algorithm for allocating time on the CPU (server). Recently [1] tight bounds have been placed on the response time $T(t)$ as a function of t; that is, functions $T_L(t)$ and $T_U(t)$ have been established for any M/G/1 system in the sense that $T_L(t) \leq T(t) \leq T_U(t)$ and, in fact, these are the tightest bounds possible. Thus, the range of possible values for $T(t)$ has been established. A remaining question is how one might synthesize a scheduling algorithm which yields a desired feasible response time in this permitted range. The three algorithms mentioned above, as well as numerous others, provide examples of achievable behavior but, as yet, the necessary and sufficient conditions for a function $T(t)$ to be a response time function are unknown.

Previous studies have been directed toward specific algorithms and their behavior. In this paper we present a method whereby it is possible to create a continuum of algorithms from any given one. The continuum is achieved by varying a single parameter in our model and permits a performance variation which ranges from the original algorithm to that of the FCFS algorithm in a continuous fashion. If the original algorithm's mean response time

[+] This work was supported by the Advanced Research Projects Agency of the Department of Defense (DAHC15-69-C-0285).

has already been solved for, then by our procedure we may immediately write down the mean response time for the continuum; on the other hand, if the distribution (or its transform) is given analytically, then so too is that for the continuum by the methods developed in this paper. Below we present the model, the analysis, and finally some examples and discussion of this class of continuous algorithms.

THE MODEL

The class of queueing systems we consider are of the form M/G/1. The Poisson arrivals are assumed to occur at an average rate of λ per second and the distribution of time required in the CPU (the service time distribution) will be denoted by $B(t)$ with first and second moments denoted by
denoted by $\overline{t}$ and $\overline{t^2}$, respectively.

We refer to our algorithms as "selfish scheduling algorithms" (SSA), the concept for which was first introduced in [13]. The principal behind this model is that all customers in the computer system currently requesting service are divided into two groups: those in a "queue box" waiting for service; and those in a "service box" sharing the facility in a fashion which we shall refer to as the "raw" scheduling algorithm. This raw scheduling algorithm may be of any type, as for example any of those already discussed above (RR, FB, FCFS, etc.). An arrival always enters the queue box first, where his priority (a numerical value) increases from zero at a positive rate α; similarly, whenever a job is in the service box, his priority increases at a positive rate β (regardless if he is sharing the CPU or merely waiting in the service box for access to the CPU). See Figs. 1 and 2. All customers possess the same parameters α, β and we restrict our attention to the case $0 \leq \beta \leq \alpha$.

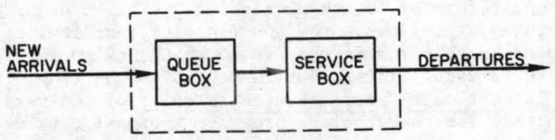

Figure 1. Decomposition of the SSA System

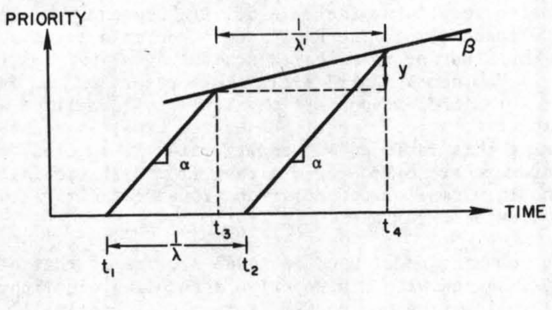

Figure 2. Calculation of the Conditional Arrival Rate to the Service Box

A customer will pass from the queue box to the service box when his priority climbs to a value equal to the priority of the customers in the service box (note that all customers in the service box have the same priority, which grows at a rate β). Since $0 \leq \beta \leq \alpha$, a customer who was originally placed in the queue box must sooner or later catch up with those customers in the service box and join them to share the service facility in a fashion dictated by the raw scheduling algorithm*; in particular, there is no feedback from the service box to the queue box. Since those customers in the service box are attempting to "run away" with the service facility, we choose to call these "selfish" scheduling algorithms. The parameters α and β will appear in our results as a ratio (β/α) whose value may be varied in a continuous way in order to achieve a variety of system behaviors as shown below.

*In any case when the service box empties completely, then that customer (if any) with the highest value of priority in the queue box will immediately pass to the service box, at which time his priority growth rate drops from α to β. A customer arriving to an empty system passes directly into the service box (and his priority then grows from zero at a rate β).

ANALYSIS

Of interest to us is the response time (time in system) conditioned on the required service time (t). Since the response time equals the sum of the waiting time and the service time, it is sufficient to solve for the waiting time itself. Below we obtain a result for the Laplace transform of the density associated with this conditional waiting time and which we denote by $W^*(s|t)$. The mean waiting time conditioned on this service time will be denoted by

$$W(t) \overset{\Delta}{=} E[\text{waiting time}|t \text{ seconds of service required on the CPU}] \tag{1}$$

Clearly, in terms of our earlier definition, we must have

$$T(t) = W(t) + t \tag{2}$$

Furthermore, let

$$B^*(s) \overset{\Delta}{=} \int_0^\infty e^{-st} dB(t) \tag{3}$$

be the transform associated with the service time density and

$$\rho \overset{\Delta}{=} \lambda \overline{t} \tag{4}$$

be the usual utilization factor for the system. Moreover, we define

$$\lambda' \overset{\Delta}{=} \lambda\left(1 - \frac{\beta}{\alpha}\right) \tag{5}$$

and

$$\rho' \overset{\Delta}{=} \rho\left(1 - \frac{\beta}{\alpha}\right) = \lambda' \overline{t} \tag{6}$$

Let us assume that we have solved for the behavior of the raw scheduling algorithm in an isolated M/G/1 time-sharing system (i.e., one with no queue box); assume that the transform for the conditional waiting time in this case is given by $\hat{W}^*_\lambda(s|t)$ where we have explicitly noted the dependence of this result upon the Poisson arrival parameter λ. In this isolated case the mean waiting time and mean response time will be denoted by $\hat{W}_\lambda(t)$ and $\hat{T}_\lambda(t)$, respectively.

We wish to solve for $W^*(s|t)$ in terms of $\hat{W}^*_\lambda(s|t)$ and other system parameters; in addition, we wish to express $W(t)$ or $T(t)$ in terms of $\hat{W}_\lambda(t)$ or $\hat{T}_\lambda(t)$. The principal result of this paper is given in the following theorem.

Theorem

For any selfish scheduling algorithm (SSA) with parameters λ, ρ, β, and α, and for which $\rho < 1$, we have

$$W^*(s|t) = \left(\frac{1-\rho}{1-\rho'}\right)\left(\frac{s - \lambda' + \lambda' B^*(s)}{s - \lambda + \lambda B^*(s)}\right)\hat{W}^*_{\lambda'}(s|t) \tag{7}$$

where $\hat{W}^*_{\lambda'}(s|t)$ is the transform of the conditional waiting time density for the raw scheduling algorithm in isolation with a Poisson arrival rate λ' as given in Eq. (5). Moreover,

$$T(t) = \frac{\lambda \overline{t^2}}{2(1-\rho)} - \frac{\lambda' \overline{t^2}}{2(1-\rho')} + \hat{T}_{\lambda'}(t) \tag{8}$$

where, again, $\hat{T}_{\lambda'}(t)$ is the mean conditional response time* for the raw scheduling algorithm at an input rate λ'.

Proof

Let us follow a tagged customer through the system and condition his service requirement to be t seconds. In Fig. 2 we have portrayed the case in which our tagged customer arrives at an instant t_1. At time t_3 his priority has increased to that of those in the service box, at which point he enters the service box. The arrival rate of customers to the service box, conditioned on the presence of our tagged customer in that box, will be denoted by λ' (typically different from λ which is the arrival rate of jobs to the overall system) but will still be a Poisson arrival process [13]. Thus we see that the service box itself (conditioned on the presence of our tagged customer in that box) is also an M/G/1 system with average arrival rate λ' and with service distribution $B(t)$ which is the same as for the overall system. This turns out to be the key to analyzing the SSA systems.

Let us now calculate λ'. Referring once again to Fig. 2 we have indicated a second arrival to the system at time t_2 where an average interarrival time is of course $1/\lambda$. These two adjacent arrivals will arrive to the service box at a slightly slower rate, namely λ' where their interarrival time $(t_4 - t_3)$ is shown in the figure. We are sure that the service box does not empty during this interval since we are calculating the service box arrival rate conditioned on the presence of the tagged customer in that box. We may calculate the vertical offset y in two different ways as shown from the geometry of that diagram:

$$y = \frac{\beta}{\lambda'}$$

$$y = \left(\frac{1}{\lambda'} - \frac{1}{\lambda}\right)\alpha$$

which therefore gives

$$\lambda' = \lambda\left(1 - \frac{\beta}{\alpha}\right)$$

This is consistent with Eq. (5).

We have already defined $W^*(s|t)$ as the transform for the waiting time density conditioned on t seconds of service, whose mean is given by $W(t)$. Let us further define: $S^*(s|t)$ as the transform of the conditional response time whose mean is $T(t)$; $Q^*(s|t)$ as the transform of the conditional density of the total time a customer spends in the queue box; $Y^*(s|t)$ as the transform of the conditional density of the time a customer spends in the service box; and $V^*(s|t)$ as the transform of the conditional density of waiting time in the service box with mean $V(t)$. It is intuitively clear (and can be shown rigorously [16]) that the time our tagged customer spends in the queue box is independent of the time he waits in the service box. As a result $Q^*(s|t)$ will be independent of t and may therefore be written as $Q^*(s|t) = Q^*(s)$. Furthermore, the total waiting time in the system for our tagged customer will be the sum of the time he waits in the queue box and the time he waits in the service box; since these are independent, we have immediately

$$W^*(s|t) = Q^*(s)V^*(s|t) \tag{9}$$

Moreover, the time spent in the queue box will be independent of the raw scheduling algorithm (so long as it is a conservative** algorithm, the case of interest here) and will depend only upon α and β. In particular, then the flow of customers from the queue box to the service box will occur at a rate λ' if the service box is not idle (and at an infinite rate if the service box goes idle and if the queue box is not empty). Thus, whereas the functions $W^*(s\ t)$ and $V^*(s\ t)$ will depend upon the scheduling algorithm, their ratio $Q^*(s)$ will be independent of that algorithm.

Let us assume that we have solved for the behavior of the raw scheduling algorithm in an isolated M/G/1 time-sharing system. In particular, let us assume that we have calculated the transform of the conditional wasted time $\hat{W}^*_{\lambda}(s|t)$. In this case then it is clear that for the service box of the SSA system we must have, using this raw scheduling algorithm in that box,

$$V^*(s|t) = \hat{W}^*_{\lambda'}(s|t) \tag{10}$$

We must now solve for $Q^*(s)$ in order to complete our solution. This task is simple since the time spent in the queue box is independent of the raw scheduling algorithm and also is independent of the tagged customer's service time. Therefore, let us choose a specific raw scheduling algorithm, the FCFS algorithm, in order to solve for $Q^*(s)$. In this case, the overall SSA system really becomes one large FCFS system since the oldest customer in this system will receive the full attention of the server. Furthermore, the service box itself behaves as an FCFS M/G/1 system with an arrival rate λ'. From these two observations and from the well-known Pollaczek-Khinchin transform equation for the waiting time in a FCFS M/G/1 system [17], we may write down immediately

$$W^*(s|t) = \frac{s(1 - \rho)}{s - \lambda + \lambda B^*(s)}$$

and

$$V^*(s|t) = \frac{s(1 - \rho')}{s - \lambda' + \lambda' B^*(s)}$$

These last two equations hold only when the raw scheduling algorithm is FCFS, but we have already established that the ratio $Q^*(s)$ will be independent of this algorithm and so we have immediately from Eq. (9) that

$$Q^*(s) = \left(\frac{1 - \rho}{1 - \rho'}\right)\left(\frac{s - \lambda' + \lambda' B^*(s)}{s - \lambda + \lambda B^*(s)}\right)$$

Using this last general result and Eq. (10) we may then substitute back into Eq. (9) to yield the final solution for the transform of the conditional waiting time density in an arbitrary SSA system, thus establishing Eq. (7).

Using the usual moment generating properties of Laplace transforms, that is $-W(t) = \lim_{s\to 0} dW^*(s|t)/ds$, we immediately obtain the mean conditional waiting time and from this simply obtain the mean conditional response time as given in Eq. (8). This completes the proof of our main theorem.

Let us now consider a cascade of SSA systems. In particular, consider Fig. 1 with parameters α_1 and β_1 corresponding to the original α and β parameters described earlier. Assume now that the service box itself is an SSA system with its own queue box and its own service box with parameters α_2 and β_2. Let us continue this iterated structure in which each service box contains its own queue box and service box down to say N levels. Effectively we have now defined N raw scheduling algorithms, the nth of which has a response function which we shall denote by $\hat{W}^*_{\lambda}(s|t,n)$ (for a system Poisson input rate of λ). We assume that the innermost raw scheduling algorithm (the Nth) has a known performance function $\hat{W}^*_{\lambda}(s|t,N)$. We may easily analyze the overall performance of this cascaded system by applying our main theorem iteratively. For this purpose let us define $\lambda_n = \lambda_{n-1}[1 - (\beta_n/\alpha_n)]$, $\rho_n = \lambda_n\bar{t}$

*If we replace T by W in this equation, then we have the corresponding result for the mean conditional waiting time.

**A conservative algorithm is one which neither creates nor destroys "work"; that is, all arriving customers must be served to completion with no overhead and the server must never go idle when customers are awaiting service [17].

for $n = 1, 2, \ldots, N$ (and $\lambda_0 = \lambda$, $\rho_0 = \rho$) and

$$g(n,m) = \left(\frac{1-\rho_n}{1-\rho_m}\right)\left(\frac{s-\lambda_m+\lambda_m B^*(s)}{s-\lambda_n+\lambda_n B^*(s)}\right)$$

With these definitions we have immediately that

$$W^*(s|t) = g(0,1)\, \hat{W}^*_{\lambda_1}(s|t,1)$$

This last corresponds exactly to Eq. 7 where $\lambda' = \lambda_1$ and $\rho' = \rho_1$. Continuing, we note that the first service box is itself an SSA system and so we may apply Eq. 7 again to yield

$$\hat{W}^*_{\lambda_1}(s|t,1) = g(1,2)\, \hat{W}^*_{\lambda_2}(s|t,2)$$

and in general

$$\hat{W}^*_{\lambda_n}(s|t,n) = g(n, n+1)\, \hat{W}^*_{\lambda_{n+1}}(s|t, n+1) \qquad n = 1,2,\ldots,N-1$$

This immediately leads to the following solution for our cascaded SSA system

$$W^*(s|t) = g(0,1)g(1,2) \ldots g(n-1, N)\, \hat{W}^*_{\lambda_N}(s|t,N)$$

The product of the g functions simplifies through successive cancellations, yielding finally

$$W^*(s|t) = \left(\frac{1-\rho}{1-\rho_N}\right)\left(\frac{s-\lambda_N+\lambda_N B^*(s)}{s-\lambda+\lambda B^*(s)}\right)\hat{W}^*_{\lambda_N}(s|t,N)$$

We further note that $\lambda_N = \lambda[1-(\beta_1/\alpha_1)][1-(\beta_2/\alpha_2)] \ldots [1-(\beta_N/\alpha_N)]$ and so for fixed $\{\beta_n/\alpha_n\}$ we may then find a ratio β/α such that $\lambda_N = \lambda[1-(\beta/\alpha)]$. If we now use these values of α and β in a single stage SSA system, we have:

Corollary

The iterated SSA system provides no generality beyond that of the single stage SSA system.

DISCUSSION AND EXAMPLES

Let us now examine the range of possible behavior for the SSA systems as we vary the ratio β/α over its values. From the basic structure of the selfish scheduling algorithms we see when $0 < \alpha = \beta$ that the system behaves in a pure FCFS fashion since those in the queue box can never catch those in the service box and therefore the oldest arrival will be served to completion. In this case ($\beta/\alpha = 1$) we see from Eqs. (5) and (6) that $\lambda' = \rho' = 0$; in this case the waiting time will be zero with probability 1 for any raw scheduling algorithm and so $\hat{W}^*_0(s|t) = 1$. Thus we are immediately led to

$$\lim_{\beta=\alpha} W^*(s|t) = \frac{s(1-\rho)}{s-\lambda+\lambda B^*(s)}$$

This is the classical Pollaczek-Khinchin transform equation for the waiting time density in an FCFS system, which confirms our earlier reasoning that this limiting case behaves in a pure FCFS fashion.

At the other extreme we note that when $0 = \beta$ then customers enter with and always maintain a priority value of zero; therefore, all customers in the system will always be in the service box and our SSA algorithm reduces to the raw scheduling algorithm. We may also establish this result by noting from Eqs. (5) and (6) that in this limit ($\beta = 0$) then $\lambda' = \lambda$ and $\rho' = \rho$. We have from Eq. (7) then

$$\lim_{\beta=0} W^*(s|t) = \hat{W}^*_\lambda(s|t)$$

which shows that the performance is identical to that of the raw scheduling algorithm.

Let us demonstrate the behavior of these SSA systems by means of some examples. First, we consider the selfish round robin (SRR) system in which the raw scheduling algorithm is RR. From Eq. (8) we have immediately that

$$T(t) = \frac{\lambda\overline{t^2}}{2(1-\rho)} - \frac{\lambda'\overline{t^2}}{2(1-\rho')} + \frac{t}{1-\rho'} \qquad (11)$$

which is easily converted to the form

$$T(t) = \frac{\beta/\alpha}{1-\rho'} T_{FCFS}(t) + \left[1 - \frac{\beta/\alpha}{1-\rho'}\right] T_{RR}(t)$$

where $T_{FCFS}(t)$ is the mean conditional response time in a FCFS M/G/1 system and $T_{RR}(t)$ is the mean conditional response time in a RR M/G/1 system ($T_{RR}(t) = t/(1-\rho)$); this exposes the nature of the linear mixing of the SRR system. It is well known that $T_{RR}(t_0) = T_{FCFS}(t_0)$ for $t_0 = \overline{t^2}/2\overline{t}$. Below in Fig. 3 we give an example for the mean conditional waiting time in the SRR system for the case of exponential service time. In Fig. 4 we give the standard deviation of the waiting time for the same system.

As a second example we consider the SFB system which has an FB raw scheduling algorithm. This gives

$$T(t) = \frac{\lambda\overline{t^2}}{2(1-\rho)} - \frac{\lambda'\overline{t^2}}{2(1-\rho')} + \frac{\lambda'\overline{t_t^2}}{2(1-\rho'_t)^2} + \frac{t}{1-\rho'_t}$$

where

$$\overline{t_t^k} = \int_0^t x^k\, dB(x) + t^k[1 - B(t)]$$

and $\rho_t = \lambda\overline{t}_t$ and $\rho'_t = \lambda'\overline{t}_t$. For the same system as in the two previous figures we find that the SFB system produces Figs. 5 and 6 below.

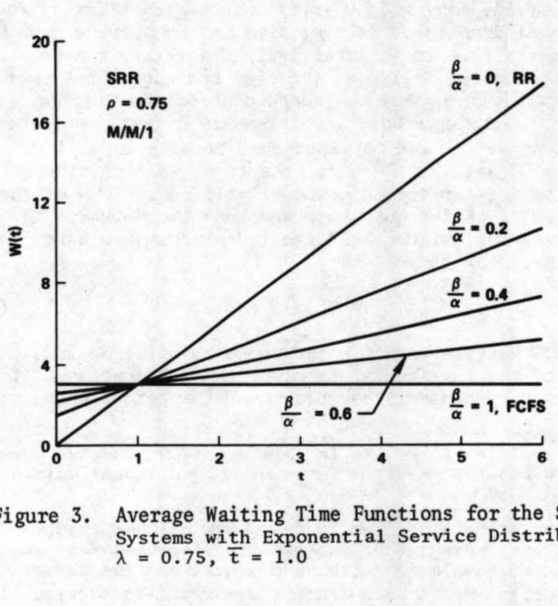

Figure 3. Average Waiting Time Functions for the SRR Systems with Exponential Service Distribution. $\lambda = 0.75$, $\overline{t} = 1.0$

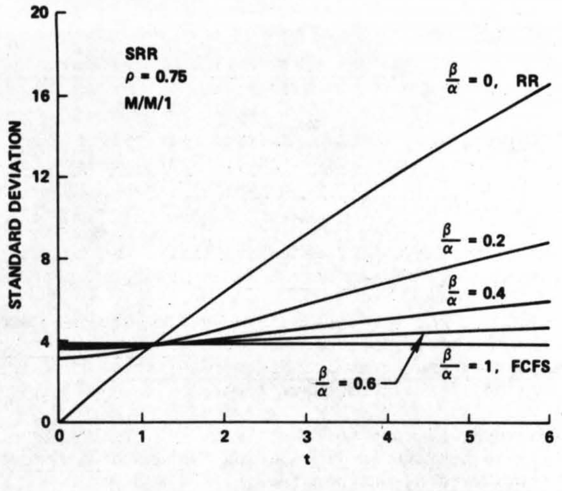

Figure 4. Standard Deviation of the Waiting Time for the SRR Systems with Exponential Service Distribution. $\lambda = 0.75$, $\bar{t} = 1.0$

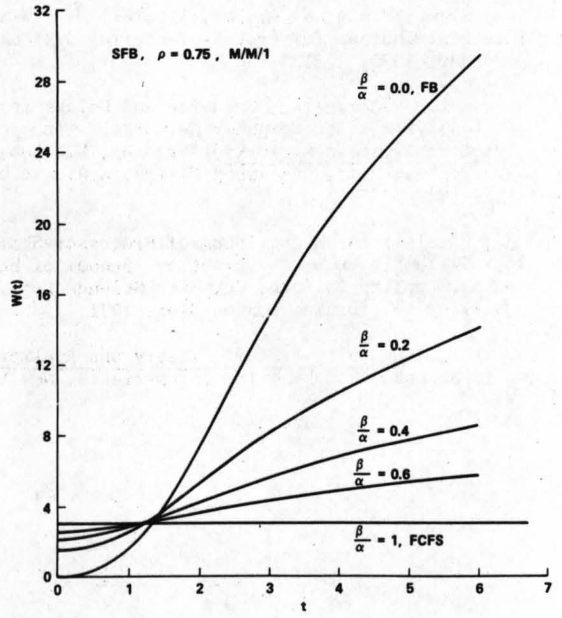

Figure 5. Average Waiting Time Functions for the SFB Systems with Exponential Service Distribution. $\lambda = 0.75$, $\bar{t} = 1.0$

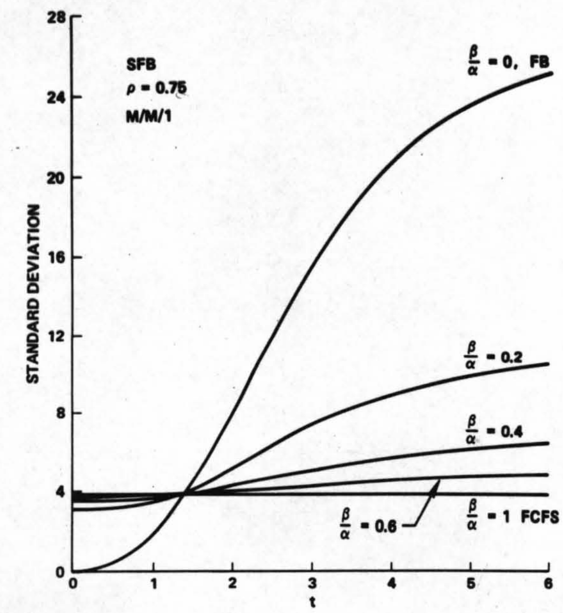

Figure 6. Standard Deviation of the Waiting Time for the SFB Systems with Exponential Service Distribution. $\lambda = 0.75$, $\bar{t} = 1.0$

Let us now demonstrate the behavior of the mean conditional time for different service time distributions. In Fig. 7 we show this function for the SFB algorithm with a two-stage Erlangian service time distribution for the same value of $\rho = 0.75$ as in our earlier examples. Similarly, in Fig. 8 we show the SFB case with hyperexponential service time; again we have $\rho = 0.75$ with the mean service time of the two stages of the hyperexponential system equal to 0.2 and 1.8 seconds, respectively.

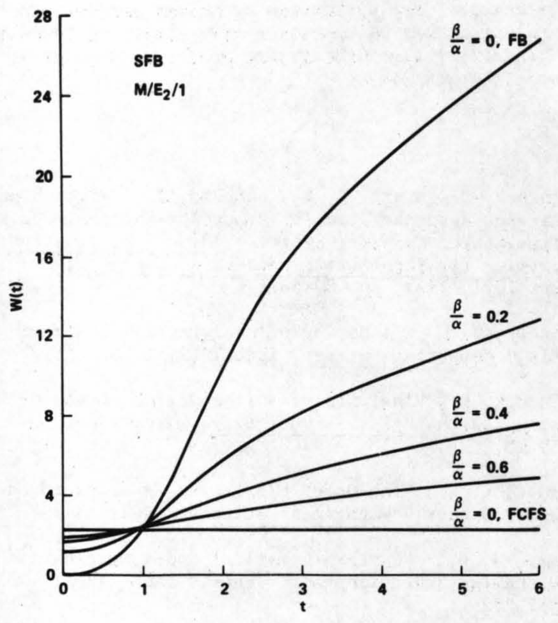

Figure 7. Average Waiting Time Functions for the SFB Systems with 2-Stage Erlangian Service Distribution. $\lambda = 0.75$, $\bar{t} = 1.0$

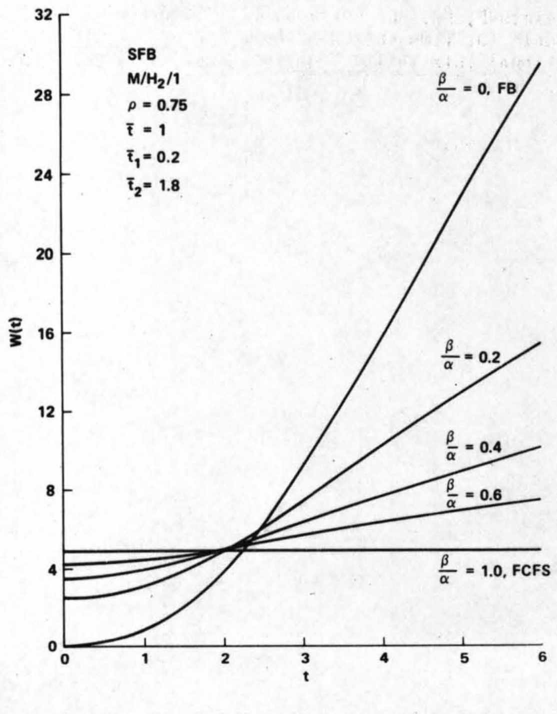

Figure 8. Average Waiting Time Functions for the SFB Systems with Hyperexponential Service Distribution. $\lambda = 0.75$, $\bar{t} = 1.0$

CONCLUSIONS

In this paper we have extended the class of selfish scheduling algorithms beyond that discussed in [13]. Our principal result was to show that the SSA algorithm permits us to take any solved system and immediately display the results for an SSA system in which the original algorithm appears as the raw scheduling algorithm. This provides us with a continuum of systems which depend upon the parameter β/α and which will give a performance which ranges from the FCFS system all the way through to the raw scheduling algorithm itself.

This generality may be obtained with almost no cost of implementation in a computer system. The implementation merely consists of counting at a rate α for units in the queue box and at a rate β for all those in the service box. Furthermore, the evaluation of system performance for these systems may be expressed very simply in terms of the raw scheduling algorithm system performance as given in the main theorem above.

REFERENCES

1. Kleinrock, L., Muntz, R. R., and Hsu, J., "Tight Bounds on Average Response Time for Processor-Sharing Models of Time-Shared Computer Systems," Proc. International Federation for Information Processing Congress, August 1971, TA-2, pp. 50-58.
2. McKinney, J. M., "A Survey of Analytical Time-Sharing Models," Computing Surveys, June 1969, 1:105-116.
3. Kleinrock, L., "Analysis of a Time-Shared Processor," Naval Research Logistics Quarterly, March 1964, 11:59-73.
4. Schrage, L. E., "The Queue M/G/1 with Feedback to Lower Priority Queues," Management Science, 1967, 13:466-471.
5. Shemer, J. E., "Some Mathematical Considerations of Time-Sharing Scheduling Algorithms," JACM, April 1967, 14:262-272.
6. Kleinrock, L. and Muntz, R. R., "Multilevel Processor-Sharing Queueing Models for Time-Shared Models," Proc. Sixth International Telegraphic Congress, August 1970, 341/1-341/8.
7. Kleinrock, L., and Coffman, E., "Some Feedback Queueing Models for Time-Shared Systems," Proc. Fifth International Teletraffic Congress, June 1967, pp. 91-92.
8. Scherr, A. L., An Analysis of Time-Shared Computer Systems, MIT Press, Cambridge, Mass., 1967.
9. Kleinrock, L., "Certain Analytic Results for Time-Shared Processors," Proc. International Federation for Information Processing Congress, August 1968, pp. 838-845.
10. Greenberger, M., "The Priority Problem and Computer Time-Sharing," Management Science, 1966, 12:888-906.
11. Adiri, I., and Avi-Itzhak, B., "A Time-Sharing Queue with a Finite Number of Customers," Operations Research Statistics and Economics Mimeograph Series, No. 10, Technion, Israel, February 1968.
12. Kleinrock, L., and Coffman, E. G., "Distribution of Attained Service in Time-Shared Systems," J. Computer System Science, October 1967, 1:287-298.
13. Kleinrock, L., "A Continuum of Time-Sharing Scheduling Algorithms," AFIPS Conference Proc., Spring Joint Computer Conference, May 1970, pp. 453-458.
14. Coffman, E. G., Muntz, R. R., and Trotter, H., "Waiting Time Distribution for Processor-Sharing Systems," JACM, January 1970, 17:123-130.
15. Kleinrock, L., "Scheduling, Queueing and Delays in Time-Shared Systems and Computer Networks." In Computer-Communication Networks, Abramson, N., Kuo, F. (eds.), Prentice-Hall, Englewood Cliffs, N.J., to be published, May 1973.
16. Hsu, J., "Analysis of a Continuum of Processor-Sharing Computer Systems," Ph.D. dissertation, School of Engineering and Applied Science, Computer Science Dept., University of California, Los Angeles, 1971.
17. Kleinrock, L., Queueing Systems: Theory and Applications, to be published by Wiley Interscience, New York, 1973.