Processor Shadowing: Maximizing Expected Throughput in Fault-Tolerant Systems

J. L. Bruno,[1,4] *E. G. Coffman, Jr.,*[2] *J. C. Lagarias,*[3] *T. J. Richardson,*[2] and *P. W. Shor*[3]

[1]Computer Science Dept., University of California, Santa Barbara, CA 93106
[2] Bell Labs, Lucent Technologies, Murray Hill, NJ 07974
[3] AT&T Research, Murray Hill, NJ 07974

December 8, 1998

Abstract

This paper studies parallel processing as a device for increasing fault tolerance. In the first of two basic models, a single job with a given running time is to be run on a finite set of processors; each processor is subject to failure but only while running a job. If a job is running on only one processor, and that processor fails, then the job must be restarted on another processor, assuming not all processors have already failed. To avoid such losses in accrued running time when at least two processors are available, it can be decided at any time to run the job synchronously on two processors in parallel, a replication technique we call *shadowing*. Clearly, shadowing has its own downside: while two processors are running, the failure rate is doubled. We show how to resolve this trade-off optimally; we devise a policy that schedules shadowing in such a way as to maximize the probability that the job finishes before all processors fail. We prove that the policy is of threshold type. That is, depending on the number of processors and the duration of the job, there is an optimal time to begin shadowing; once started, shadowing continues so long as neither processor fails and the job does not complete. We also show that the thresholds are monotone in the number of processors, i.e., if more processors are initially available, then shadowing should be started sooner.

In the second of our two models, we have the same set-up except that we have an unbounded number of jobs, each having the same running time, and the objective is to maximize the expected number of jobs completed before all processors fail. We show that the optimal policy is again of threshold type, but that the thresholds are, surprisingly, *not* monotone in the number of processors. The optimal thresholds have a curious oscillatory behavior that we study in detail.

Variants of the above problems are also analyzed using the same methods; several other variants are left as interesting open problems.

Key words: fault tolerant scheduling, stochastic scheduling, reliable computation, stochastic optimization

1. Introduction

Assume we are given a single job with a known running time $\tau > 0$ which is to be run on a system of $m \geq 1$ faulty processors. The processor times-to-failure are independent random variables with the common exponential law $F(t) = 1 - e^{-t},\ t \geq 0.$ A processor can fail only while it is running a job. If

[4]The author was supported in part by a University of California MICRO grant and the Xerox Corporation.

a job is being run by only one processor and that processor fails, then the job must be restarted from the beginning. In order not to risk the loss of running time accumulated on a single processor, the running of the job can be replicated (done in parallel) on two processors, a technique we call *shadowing*, which we assume can be stopped and started with negligible delay; the processors work synchronously, always in the same state. The problem is to schedule shadowing so as to maximize the probability that the job finishes before all processors fail.

In practice, when a processor is running a job, the state of the computation is composed of two parts, one volatile and located in the processor, and one nonvolatile and stored in system memory; only the volatile part is lost when a failure occurs. If we are shadowing and one of the processors fails at time $t < \tau$, the non-failed processor continues uninterrupted from time t onward. If a job is being run by one processor and at time t, to commence shadowing the volatile state of the running processor is installed in the new processor and both continue from time t onward. We assume that the time to install the volatile part of a state in a new processor is negligible.

The downside of processor shadowing is that the failure rate is doubled, but the downside of running the job on only one processor is that the job must be restarted from the beginning when a failure occurs. In Section 2, we show that this trade-off is resolved optimally (in the sense of completion probability) by a threshold policy. For each τ and m there exists an optimal threshold τ_m such that a single processor runs the job in $[0, \tau_m)$, and if no failure has occurred, then shadowing is done throughout $[\tau_m, \tau]$. Note that there is no loss in our restriction to two-processor shadowing; running the job using 3 or more processors does not enhance the reliability (we just need two processors to avoid starting the job from the beginning if one processor fails) and increases the probability of processor failure.

In a direct generalization of the above problem, there are $n > 1$ jobs to be completed (in any order), and each requires τ time units. This more difficult problem is open for general n, but there is much that we can say about the large-n behavior of policies that attempt to complete as many jobs as possible before all processors fail. More precisely, Section 3 has the same set-up as Section 2 except that there are infinitely many jobs, each of duration τ, and the optimal policy must now maximize the expected number of jobs completed before all processors fail. We give such a policy in Section 3 and prove that it also is of threshold type.

Throughout the paper, the symbol π denotes a shadowing policy. A policy decision, $\pi(m, t) = \pi(m, t, \tau)$, is determined by the number m of processors, the accumulated processing time t of a job, and the job's original processing time τ; the decision value is 1 or 2, the number of processors that are to run the job. Thus, fixing τ, a shadowing policy is a map $\pi : \mathrm{N} \times [0, \tau] \to \{1, 2\}$ that determines the number of processors that are to run the job in the state defined by $t \in [0, \tau]$, the accumulated processing time, and $m \in \mathrm{N}$ the number of processors available. Trivially, $\pi(1, t) = 1,\ 0 \leq t \leq \tau$.

The key results on optimal policies are proofs of monotonicity properties, since such properties yield simplifications in computing the parameters of optimal policies. For example, the optimal threshold policies of Sections 2 and 3 are based on the fact that $\pi(m,t)$ is nondecreasing in the time parameter t. For the single-job problem of Section 2, we also prove that $\pi(m,t)$ in nondecreasing in m (the thresholds are decreasing functions of m). This ordering $\ldots < \tau_m < \tau_{m-1} < \ldots < \tau$ may seem unsurprising; it simply says that shadowing should begin earlier in systems with more available processors. However, we find that this monotonicity does *not* apply to the optimum threshold policy for the problem with infinitely many jobs. Indeed, τ_m exhibits an oscillatory behavior; Section 3 describes the asymptotics of this behavior in some detail. We show that, if $\tau > \ln 2$, then τ_m converges to $\ln 2$ as $m \to \infty$ and we derive the asymptotic form of the expected number of complete jobs under an optimal policy. This asymptotic form is, roughly speaking, a fixed point of the recursion (in m) satisfied by optimal solutions. To obtain more refined estimates we consider perturbing away from the fixed point and then continuing the recursion satisfied by the asymptotic solution. It turns out that oscillatory behavior is intrinsic to the recursion satisfied by the perturbations. To establish this we first show that a simplified recursion exhibits the same first order response to a perturbation. Then, in the appendix, for a particular τ, we prove the oscillatory behavior of solutions to this simplified recursion by showing that they are well approximated (asymptotically) by sinusoids.

Problems like our shadowing problem (e.g., checkpointing and rollback/recovery problems) make up the mathematical foundations of fault-tolerant scheduling. Many references to the literature can be found in $[2-12]$. Implicitly, our problem models situations where (i) the reliability of job completions must be significantly higher than the reliability of an individual processor, (ii) the time to repair a processor exceeds the time that can be devoted to the completion of a job (processor repair and re-use is not considered), and (iii) the failure rate of idling processors is negligible, at least by comparison to the failure rate of processors running jobs. In Section 4, the final section, we mention several open problems in which our model is extended to make it more practical.

2. The Single-Job Problem

For the single-job problem, we let $Q_m^\pi(t)$ denote the conditional probability of successfully completing the job given that the job has already accumulated t units of running time, a total of m processors is available, and the processors are assigned according to policy π. Observe the boundary conditions for all π

(1) $$Q_1^\pi(t) = e^{-(\tau-t)},\ 0 \le t \le \tau, \quad Q_m^\pi(\tau) = 1,\ m \ge 1.$$

Define

$$Q_m^\star(t) := \sup_\pi Q_m^\pi(t)\,.$$

and call a policy π *optimal* if $Q_m^\pi(t) = Q_m^\star(t)$.

Remark In defining the optimality of a policy we might have required only that $Q_m^\pi(0) = Q_m^\star(0)$. This is, however, equivalent to the definition above. ■

Below, we first develop the equations underlying the analysis of an optimal policy, $\pi^\star$. We then prove that $\pi^\star(m,t)$ is nondecreasing in t (i.e., $\pi^\star$ is a threshold policy), and nondecreasing in m (the threshold is decreasing in m). Computational issues are covered in Section 2.4, and the analysis in Section 2.5 of a useful variant concludes the section.

2.1. Optimality equations

Given continuous functions $f(t)$ and $g(t)$ on $[0,\tau]$, define

$$\varphi_1(f,g)(t) := f(t) - g(0), \qquad \text{and} \qquad \varphi_2(f,g)(t) := 2(f(t) - g(t)).$$

For a given policy π, we can compute $Q_m^\pi(t)$ for each $m \geq 2$, by recursively solving

$$\frac{d}{dt}Q_m^\pi(t) = \varphi_{\pi(m,t)}(Q_m^\pi, Q_{m-1}^\pi)(t), \;\; Q_m^\pi(\tau) = 1, \tag{2}$$

where $Q_1^\pi(t) = e^{-(\tau - t)}$. To see this, suppose $\pi(m,s) = 1$ for $s \in [t, t+\Delta t]$. We can write

$$Q_m^\pi(t) = (1 - e^{-\Delta t})Q_{m-1}^\pi(0) + e^{-\Delta t}Q_m^\pi(t + \Delta t)\,.$$

The first term is the probability that the processor fails in the interval $[t, t+\Delta t]$ and thereafter the job, starting with zero accumulated processing time and $m-1$ processors, is successfully completed. The second term is the probability that the processor does not fail in the interval $[t, t+\Delta t]$ and thereafter the job, starting with $t + \Delta t$ accumulated processing time and m processors, is successfully completed. Solving for $[Q_m^\pi(t+\Delta t) - Q_m^\pi(t)]/(1 - e^{-\Delta t})$ and taking the limit as Δt goes to zero, we get (2) with $\pi(m,t) = 1$. A similar analysis assuming $\pi(m,s) = 2$ for $s \in [t, t+\Delta t]$ results in (2) with $\pi(m,t) = 2$. We can generalize to arbitrary measurable π either by approximating and taking limits or by refining the argument above to Lebesgue points of π.

Equation (2) suggests that, for each $m \geq 2$, we should have $Q_m^\star(t) = q_m(t)$ for $0 \leq t \leq \tau$, where the $q_m,\; m \geq 2$, solve the following recursive system of differential equations:

$$\frac{d}{dt}q_m(t) = \min\{\varphi_1(q_m, q_{m-1})(t),\; \varphi_2(q_m, q_{m-1})(t)\}, \tag{3}$$

$$q_m(\tau) = 1 \tag{4}$$

with $q_1(t) = e^{-(\tau-t)}$. To see this, we note first, from standard results on o.d.e.'s [1], that q_m is a uniquely defined C^1 function. Next, if we define a policy $\pi^\star$ according to

$$(5) \qquad \pi^\star(m,t) = \begin{cases} 1, & \varphi_1(q_m, q_{m-1}) < \varphi_2(q_m, q_{m-1}), \\ 2, & \varphi_1(q_m, q_{m-1}) \geq \varphi_2(q_m, q_{m-1}), \end{cases}$$

it follows inductively that $Q_m^{\pi^\star}(t) = q_m(t)$ solves (2), as verified below.

Lemma 1. *For each m, we have $q_m(t) = Q_m^\star(t)$ and hence, $\pi^\star$ is optimal.*

Proof: The proof is by induction. We have $q_1(t) = Q_1^\star(t)$ trivially, so assume $q_n(t) = Q_n^\star(t)$ for $n \leq m-1$, and let π be an arbitrary policy. For almost all $t \in [0,\tau]$ we have

$$\frac{d}{dt}(q_m - Q_m^\pi)(t) = \min\{\varphi_1(q_m, q_{m-1})(t),\ \varphi_2(q_m, q_{m-1})(t)\} - \varphi_{\pi(m,t)}(Q_m^\pi, Q_{m-1}^\pi)(t)\,.$$

By the inductive hypothesis, $Q_{m-1}^\pi(t) \leq q_{m-1}(t)$, we have

$$\begin{aligned} \frac{d}{dt}(q_m - Q_m^\pi)(t) &\leq \min\{\varphi_1(q_m, Q_{m-1}^\pi)(t),\ \varphi_2(q_m, Q_{m-1}^\pi)(t)\} - \varphi_{\pi(m,t)}(Q_m^\pi, Q_{m-1}^\pi)(t) \\ &\leq \varphi_{\pi(m,t)}(q_m, Q_{m-1}^\pi)(t) - \varphi_{\pi(m,t)}(Q_m^\pi, Q_{m-1}^\pi)(t) \\ &= \pi(m,t)(q_m(t) - Q_m^\pi(t)) \end{aligned}$$

almost everywhere. It is now easy to see that $\frac{d}{dt}(q_m - Q_m^\pi)(t) < 0$ almost everywhere on the set $\{t : q_m(t) < Q_m^\pi(t)\}$. It follows that if $q_m(t) - Q_m^\pi(t) < 0$ for some $t \leq \tau$ then $0 = q_m(\tau) - Q_m^\pi(\tau) \leq q_m(t) - Q_m^\pi(t)$, which is a contradiction. This completes the proof. ■

Remark. If π is an optimal policy then $\pi = \pi^\star$, up to sets of measure 0. Thus, from a practical point of view, there is a unique optimal policy. ■

Example 1: For $m = 2$ we can compute directly the policy $\pi^\star$. It is given by $\pi^\star(2,t) = 1$ if $0 \leq t < \tau/2$ and $\pi^\star(2,t) = 2$ if $\tau/2 \leq t \leq \tau$, i.e., $\pi^\star$ calls for shadowing if both processors are available and $t \in [\tau/2, \tau]$. Solving (2) we obtain

$$(6) \qquad Q_2^{\pi^\star}(t) = \begin{cases} e^{-\tau}[1 + 2(1 - e^{-\tau/2})e^t], & 0 \leq t < \tau/2, \\ 1 - (1 - e^{-(\tau-t)})^2, & \tau/2 \leq t \leq \tau. \end{cases}$$

It can be verified directly that $Q_2^{\pi^\star}$ solves (3), i.e., that $Q_2^{\pi^\star} = q_2$.

2.2. $\pi^\star(m,t)$ is monotone in t

We prove below that, whenever an optimal policy begins shadowing, it remains committed to shadowing so long as neither processor fails and the job does not complete. This means that optimal policies are of threshold type: for each $m \geq 2$, there exists a threshold $\tau_m,\ 0 < \tau_m < \tau$, such that $\pi^\star(m,t) = 1,\ 0 \leq t < \tau_m$, and $\pi^\star(m,t) = 2,\ \tau_m \leq t \leq \tau$. We begin with two preliminary results.

Lemma 2. *For every $m \geq 2$, there exist τ', τ'' with $0 < \tau' \leq \tau'' < \tau$ such that $\pi^\star(m,t) = 1,\ t \in [0,\tau')$, and $\pi^\star(m,t) = 2,\ t \in (\tau'', \tau]$.*

Proof: From the definition of $Q_m^{\pi^\star}(0)$ (a completion probability) we have $q_m(0) > q_{m-1}(0)$ and $q_m(0) < 1$ for all $m \geq 2$. Now consider (3). Since

$$\varphi_2(q_m, q_{m-1})(0) = 2\varphi_1(q_m, q_{m-1})(0) > \varphi_1(q_m, q_{m-1})(0),$$

we conclude by the continuity of φ_1 and φ_2 in t that $\pi^\star(m,t) = 1$ in some neighborhood of 0. Similarly, since $\varphi_1(q_m, q_{m-1})(\tau) > \varphi_2(q_m, q_{m-1})(\tau) = 0$, we have $\pi^\star(m,t) = 2$ in some neighborhood of τ. ■

The following technical lemma is a key ingredient in our analysis of (3). Hereafter, we often adopt the more compact notation $\dot{z}(t) := \frac{d}{dt}z(t)$.

Lemma 3. *Let $f(t),\ t \geq 0$, be a given nondecreasing C^1 function on $[t_0, t_1]$, and let $g(t)$ satisfy*

$$\dot{g}(t) \quad = \quad a(t)[g(t) - f(t)] \tag{7}$$

on the interval $[t_0, t_1)$, where $a(t)$ is a real continuous positive function. If $\dot{g}(t) < 0$ for some $t \in (t_0, t_1)$ then $\dot{g}(t') < 0$ for all $t' \in [t, t_1)$.

Proof: Assume $\dot{g}(t) < 0$ and assume contrary to the lemma that there exists $t' \in [t, t_1)$ such that $\dot{g}(t') \geq 0$. By the continuity of $\dot{g}$ we can assume that t' is the minimal such value. However, we have $g(t') \geq f(t') \geq f(t) > g(t)$, which contradicts the minimality of t'. ■

Remark If the given function $f(t)$ is strictly increasing on $[t_0, t_1)$ then we can replace the hypothesis $\dot{g}(t) < 0$ with $\dot{g}(t) \leq 0$, since $\dot{g}(t) = 0$ then implies $\dot{g}(t') < 0$ for all $t' > t$ sufficiently close to t. ■

By (5), we have $\pi^*(m,t) = 2$ if $q_m(t) - q_{m-1}(0) \geq 2q_m(t) - 2q_{m-1}(t)$, or equivalently, if

$$2q_{m-1}(t) - q_m(t) - q_{m-1}(0) \geq 0.$$

Now $q_{m-1}(0)$ is just a constant, so if we can show that

$$h_m(t) := 2q_{m-1}(t) - q_m(t) \tag{8}$$

is strictly increasing on $[0,\tau]$ for each $m \geq 2$, then by Lemma 2, $\pi^\star$ will be a threshold policy.

Lemma 4. *Let the q_m, $m \geq 2$, solve (3) with $q_1(t) \in C^1[0,\tau]$ and let the boundary conditions $q_1(t)$ and $q_m(\tau)$ be chosen so that $h_2(t)$ is increasing and $h_m(\tau) \geq h_{m-1}(\tau)$. Then h_m is increasing for each m.*

Proof: The proof is by induction on m. Assume that $h_n(t)$ is increasing for $n \leq m-1$, where $m > 2$. We know that h_m satisfies the following equation,

$$(9)\,\dot{h}_m(t) = \begin{cases} \max\{h_m(t) - 2h_{m-1}(t) + q_{m-1}(0), 2(h_m(t) - h_{m-1}(t))\}, & h_{m-1}(t) \geq q_{m-2}(0), \\ \max\{h_m(t) - h_{m-1}(0), 2(h_m(t) - q_{m-2}(0))\}, & h_{m-1}(t) < q_{m-2}(0)\,. \end{cases}$$

From (9) and the inductive hypothesis we can easily conclude that if $\dot{h}_m(t) \leq 0$ then $h_m(t) \leq h_{m-1}(t)$ for any $t < \tau$. We will show how to apply Lemma 3 to conclude that, if $\dot{h}_m(t') \leq 0$, then $\dot{h}_m(t) < 0$ for $t \in (t', \tau)$ and hence, $h_m(\tau) < h_{m-1}(\tau)$. Since we have assumed that this is not the case, we conclude that $\dot{h}_m(t) > 0$ for all $t < \tau$.

The application of Lemma 3 is piecewise. Assume that $\dot{h}_m(t') \leq 0$. By the inductive hypothesis there can be at most one $t = T_1 \leq \tau$ where $h_{m-1}(t) = q_{m-2}(0)$. We claim that there can be at most one $t = T_2 \in [t', \tau]$ where the two choices under the operative $\max$ in (9) are equal. This being the case, we see that the interval $[t', \tau]$ can be subdivided into at most three subintervals such that on each subinterval we have $\dot{h}_m(t) = a[h_m(t) - f(t)]$, where $a = 1$ or $a = 2$ and f is C^1 and increasing. We can apply Lemma 3 on each subinterval and use the fact that h_m is C^1 to observe that $\dot{h}_m < 0$ at the left endpoint of each subinterval. We then conclude that h_m is decreasing on $[t', \tau]$.

To see that the claim holds, observe that if h_m is decreasing on $[t', \tau]$ then indeed there can be at most one $t = T_2 \in [t', \tau]$ where the two choices under the operative $\max$ in (9) are equal. This consistency of the claim enables us to conclude that it holds, since h_m is uniquely determined by (9). ∎

To illustrate Lemma 4, consider $m = 2$. The boundary conditions $q_1(t) = e^{t-\tau}$ and $h_m(\tau) = 1$, and substitution into (8) of $q_1(t)$ and $q_2(t)$ gives

$$h_2(t) = \begin{cases} e^{-\tau/2}(2e^{t-\tau} - e^{-\tau/2}), & t \in [0, \tau/2), \\ e^{2(t-\tau)}, & t \in [\tau/2, \tau]. \end{cases}$$

One sees by inspection that $h_2(t)$ is strictly increasing on $[0, \tau]$.

In view of Lemma 2 and Lemma 4, we have now proved the desired result:

Theorem 1. *The optimal policy π^* is of threshold type.*

2.3. $\pi^\star(m, t)$ is monotone in m

We want to prove that if for some $a,\ 0 \leq a < \tau$, and some $m > 2$, we have $\pi^\star(m-1, t) = 2,\ a \leq t < \tau$, then $\pi^\star(m, t) = 2,\ a \leq t < \tau$. Coupled with Theorem 1, this result is equivalent to the threshold ordering $\ldots < \tau_m < \tau_{m-1} < \ldots < \tau$.

Theorem 2. *For all $m \geq 2$ we have $\tau_m < \tau_{m-1}$.*

Example 2. We illustrate the hand calculation of τ_3. We gave $q_2(t)$ in (6). Solving $\dot{q}_3(t) = 2(q_3(t) - q_2(t))$ over $[\tau/2, \tau]$, we obtain

$$q_3(t) = 4e^{-(\tau-t)} - [3 + 2(\tau - t)]e^{-2(\tau-t)}, \ \ \tau/2 \le t \le \tau$$

We verify $h_3(\tau/2) = 2q_2(\tau/2) - q_3(\tau/2) > q_2(0)$, so $\tau_3 < \tau_2 = \tau/2$. Extending $q_3(t)$ down from $\tau/2$ and solving $h_3(\tau_3) = q_2(0)$ for τ_3 gives

(10) $$\tau_3 = \frac{\tau}{4} + \frac{1}{2}\ln\frac{e^{\tau/2} - 1}{\tau/2} .$$

Continuing the calculation of $q_3(t)$ on $[0, \tau_3]$, one obtains the full solution

(11) $$q_3(t) = \begin{cases} 4e^{-(\tau-t)} - [3 + 2(\tau - t)]e^{-2(\tau-t)}, & \tau_2 \le t \le \tau \\ e^{-\tau} - 4[1 - e^{-\tau/2}]e^{-(\tau-t)} - \tau e^{-2(\tau-t)}, & \tau_3 \le t < \tau_2, \\ e^{-\tau}\left[1 + 2(1 - e^{-\tau/2})(1 + 2e^t(1 - e^{-\tau_3}))\right] & 0 \le t < \tau_3 \ . \end{cases}$$

■

Our proof of Theorem 2 will be based on a careful analysis of the function

$$r_m(t) := \frac{\dot{q}_m(t)}{\dot{q}_{m-1}(t)}, \ \ m \ge 2,$$

which we note is continuous and piecewise C^1 on $[0, \tau)$, the only exceptional points being τ_m and τ_{m-1}. We begin the proof of Theorem 2 once we have proved the properties of $r_m(t)$ given in the next two lemmas.

Lemma 5. $\lim_{t\uparrow\tau} r_m(t) = 0, \ \ m \ge 2.$

Proof: First we show that in a neighborhood of τ, the difference $d_m(t) := q_m(t) - q_{m-1}(t), \ m \ge 2$ decreases monotonically as t increases to τ. For $t \ge \tau_2$, we have $d_2(t) = e^{(t-\tau)}(1 - e^{(t-\tau)})$. For $m \ge 3$ and t in the neighborhood of τ, we have

(12) $$\dot{d}_m(t) = 2(d_m(t) - d_{m-1}(t)) .$$

Noting that $d_m(\tau) = d_{m-1}(\tau) = 0$, we apply Lemma 3 inductively over m to $-d_m$ to conclude that d_m is strictly decreasing in a neighborhood of τ.

To complete the proof of the lemma, observe that (12) and the monotonicity of $d_{m-1}(t)$ imply that, in a neighborhood of τ,

$$d_m(t) = \int_t^\tau 2e^{2(t-s)}d_{m-1}(s)ds \le \int_t^\tau 2e^{2(t-s)}d_{m-1}(t)ds = d_{m-1}(t)(1 - e^{2(t-\tau)}),$$

from which we get

$$r_m(t) \le (1 - e^{2(t-\tau)}), \ \ m \ge 3.$$

For $m = 2$, we have directly $\lim_{t\uparrow\tau} r_2(t) = 0$. ■

Lemma 6. *For all $m \ge 2$ we have $r_m(0) < 2$.*

Proof: We have $2\dot{q}_{m-1}(0) - \dot{q}_m(0) = \dot{h}_m(0) > 0$ which is equivalent to $r_m(0) < 2$. ■

Proof of Theorem 2: We prove the theorem by induction on two monotonicity properties:

$i)$ $\tau_{m-1} < \tau_{m-2} < \ldots < \tau_2$
$ii)$ $r_n(t)$ is decreasing on $[\tau_n, \tau)$ and constant on $[0, \tau_n]$, $2 \le n \le m-1$.

The proof is divided into two steps. The first step proves that $i)$ and $ii)$ imply $\tau_m < \tau_{m-1}$, and the second step proves that $i),\ ii)$, and $\tau_m < \tau_{m-1}$ imply that $r_m(t)$ is constant on $[0, \tau_m)$ and decreasing on $[\tau_m, \tau)$.

Basis ($m = 2$**)** We have already seen that $\tau_2 = \tau/2 < \tau_1 = \tau$. Direct calculation using (6) shows that $\dot{r}_2 = 0$ on $[0, \tau_2)$ and $\dot{r}_2 = -2e^{t-\tau} < 0$ on $[\tau_2, \tau]$.

Induction Step ($m \ge 3$)

We first verify, as follows, that the monotonicity of τ_m is equivalent to the monotonicity of $r_m(0)$. It is easy to see that

(13) $$q_n(t) = (q_n(0) - q_{n-1}(0))e^t + q_{n-1}(0)$$

satisfies

$$\dot{q}_n(t) = \varphi_1(q_n, q_{n-1})(t)$$

for $n \ge 1$ and $t \in [0, \tau_n]$, where $\tau_1 = \tau$ and $q_0(0) \equiv 0$. We solve the equation $h_n(\tau_n) = q_{n-1}(0)$ using (13) for $q_n(t)$ and $q_{n-1}(t)$ to obtain

(14) $$r_n(0) = 2(1 - e^{-\tau_n})$$

for $n \ge 2$. We can conclude that, for $m \ge 3$, $\tau_m < \tau_{m-1}$ if and only if $r_m(0) < r_{m-1}(0)$.

The remainder of the proof relies on the behavior of $\dot{r}_m(t)$.

Step 1. We prove $r_m(0) < r_{m-1}(0)$, and hence $\tau_m < \tau_{m-1}$, by contradiction. Assume $r_m(0) \ge r_{m-1}(0)$. We show that $r_m(t)$ increases over $[0, \tau)$, thus contradicting Lemma 5, since $r_m(0) > 0$.

Consider first the interval $[0, \tau_{m-1}]$. Using (13), we have directly $\dot{r}_m(t) = 0$ in this interval. Next, consider the interval $[\tau_{m-1}, \tau_m]$, where we have

$$\dot{r}_m(t) = r_m(t)\left(\frac{2}{r_{m-1}(t)} - 1\right).$$

Since $r_{m-1}(t) < 2$ by the inductive hypothesis and Lemma 6, we have that $r_m(t)$ is increasing in $[\tau_{m-1}, \tau_m]$.

Finally, in the interval $[\tau_m, \tau)$ we have

$$\dot{r}_m(t) = \frac{2}{r_{m-1}(t)}[r_m(t) - r_{m-1}(t)]\,.$$

But $r_m(\tau_m) \geq r_m(0) \geq r_{m-1}(0) \geq r_{m-1}(\tau_m)$ so by Lemma 3 we conclude $r_m(t)$ is increasing in $[\tau_m, \tau]$, and we arrive at our contradiction. We have now proved $r_m(0) < r_{m-1}(0)$, and hence, $\tau_m < \tau_{m-1}$.

Step 2. On the interval $[0, \tau_m)$ we have $\dot{r}_m(t) = 0$, and on the interval (τ_m, τ_{m-1}), we have $\dot{r}_m(t) = r_m - 2$. Since $r_m(\tau_m) = r_m(0) < 2$, we conclude that r_m is strictly decreasing on $[\tau_m, \tau_{m-1}]$.

Finally, on the interval (τ_{m-1}, τ), we have

$$\dot{r}_m(t) = \frac{2}{r_{m-1}(t)}[r_m(t) - r_{m-1}(t)] \,.$$

Since $r_{m-1}(t)$ is strictly decreasing on (τ_{m-1}, τ), we can apply Lemma 3 to conclude that $\dot{r}_m(t) < 0$ on this interval, for otherwise, we would again obtain a contradiction to Lemma 5. ∎

2.4. Computations

The formulas for $q_m(t)$ and the thresholds τ_m of $\pi^\star$ become progressively more awkward as m increases. However, a relatively simple formula applies to $q_m(t)$ in the interval $[\tau/2, \tau]$ where shadowing is required by $\pi^\star$ when two or more processors are available. According to (3), we derive this formula by solving $\dot{q}_m(t) = 2[q_m(t) - q_{m-1}(t)]$ with $q_m(\tau) = 1,\ m \geq 1$, and $q_1(t) = e^{-(\tau - t)},\ 0 \leq t < \tau$. Rewriting in terms of $v_m(t) = q_m(t)e^{-2t}$ and then integrating gives the simple expression

$$v_m(t) = -\int 2v_{m-1}(t)dt + c,$$

where c is a constant to be determined and where $v_m(\tau) = e^{-2\tau}$ for all $m \geq 1$, and $v_1(t) = e^{-(\tau+t)}$, $0 \leq t \leq \tau$. A simple induction proves that $v_m(t)$ is given by e^{-2t} times the right-hand side of

$$q_m(t) = 2^{m-1}e^{-(\tau-t)} - e^{-2(\tau-t)} \sum_{0 \leq i \leq n-2} \frac{2^{m-1} - 2^i}{i!}(\tau - t)^i \tag{15}$$

for $t \in [\tau/2, \tau]$.

In the remainder of this subsection we give numerical methods for computing the functions $q_m(t)$ and $n_m(t)$ and their corresponding thresholds τ_m. Let $q_{jm}(t)$ denote the expression for the optimal completion probability in $\tau_j \leq t \leq \tau_{j-1},\ 1 \leq j \leq m$. Using (3), it not difficult to show that $q_{jm}(t)$ is of the form

$$q_{jm}(t) = a_0 + a_1 e^t + (b_0 + b_1 t + \cdots + b_{m-1-j}t^{m-1-j})e^{2t} \,, \tag{16}$$

with all coefficients a_i, b_i being functions only of τ. For the computation to follow, define the coefficient vectors $a_{jm} = (a_0, a_1)$ and $b_{jm} = (b_0, \ldots b_{m-1-j})$, $1 \leq j \leq m$, where $b_{mm} = ()$ is empty (the coefficient of e^{2t} is 0).

The basis of the calculation is $m = 1$, namely,

$$\begin{aligned} \tau_1 &= \tau, \\ a_{1,1} &= (0, e^{-\tau}), \\ b_{1,1} &= (\,), \\ q_{1,1}(t) &= e^{-\tau} e^t, \;\; 0 \le t \le \tau_1 \,. \end{aligned}$$

Now let $m \ge 2$ and assume that we have calculated $a_{j,m-1}$, b_{j-m-1}, and τ_j for $j = 1, \ldots, m-1$. We apply $\dot{q}_{jm}(t) = 2(q_{jm}(t) - q_{j,m-1}(t))$ to calculate a_{jm} and b_{jm} for $j = 1, \ldots, m-1$. With $q_{m-1,m}(t)$ and $q_{m-1,m-1}(t)$ in hand, we obtain τ_m as the solution to

$$2q_{m-1,m-1}(\tau_m) - q_{m-1,m}(\tau_m) = q_{m-1,m-1}(0)\,.$$

We get

$$\tau_m = \frac{1}{2} \ln \left(-\frac{a_1}{b_0} \right) \;,$$

where $a_{m-1,m-1} = (a_0, a_1)$ and $b_{m-1,m} = (b_0)$.

Once we have τ_m we can compute a_{mm}, b_{mm}, and $q_{mm}(t)$ using $\dot{q}_{mm}(t) = q_{mm}(t) - q_{m-1,m-1}(0)$. Figure **??** gives the formulas for computing a_{jm} and b_{jm} for $j = 1, \ldots, m$ and τ_m, assuming we have previously computed $a_{j,m-1}$, $b_{j,m-1}$, and τ_j for $j = 1, \ldots, m-1$.

The method was implemented using Mathematica. Figures **??** and **??** show the values of $q_{mm}(0)$ and τ_m, respectively, versus m with $\tau = 5$.

2.5. A Variant

Suppose the number of available processors is unlimited; the job-completion probability is 1, but it is of interest to determine the expected number of processors lost prior to job completion under a policy that minimizes this objective function.

Let $M(t)$ denote the expected number of processors which will fail prior to completing the current job given that t units of processing time have elapsed and an optimal policy is used. By an argument similar to that used in Section 2 we find that $M(t)$ solves

$$\frac{d}{dt} M(t) = \max\{M(t) - M(0) - 1, -2\}$$

where $M(\tau) = 0$. The solution is found to be $M(t) = e^\tau - e^t$ if $\tau \le \ln 2$ (no shadowing occurs); otherwise, we have

$$M(t) = \begin{cases} 2 + 2(\tau - \ln 2) - e^t, & 0 \le t \le \ln 2, \\ 2(\tau - \ln 2) - 2(t - \ln 2), & \ln 2 \le t \le \tau. \end{cases}$$

(shadowing begins at $t = \ln 2$). As we will see, these results also help describe the asymptotic behavior of the solution to the problem of the next section.

3. The Problem with Infinitely Many Jobs

Recall that, in this problem, we have m processors and infinitely many jobs, each of duration τ. The problem is to find a policy that maximizes the expected number of jobs that are completed before all processors fail. Below, we first develop an optimal policy and then prove that it is of threshold type. In Section 3.3, we note that, in a surprising contrast with the single-job problem, the policy function $\pi^\star(m,t)$ is *not* monotone in m; an asymptotic analysis is given along with an accounting for the non-monotone behavior of τ_m. The section is rounded out by the analysis of an interesting, perhaps more realistic variant.

3.1. Optimality equations

Let $N_m^\pi(t)$ denote the expected number of jobs completed given that the current job has accumulated t units of running time, a total of m processors are available, and the processors are assigned according to policy π. The finishing time is not part of the objective, so there is no need to process more than one job at a time.

Since $\pi(1,t)=1$, we see that $N_1^\pi(t)$ is equal to $p(t)(1+N_1^\pi(0))$ where $p(t)=e^{t-\tau}$ is the probability that the current job will be completed. Solving for $N_1^\pi(0)$, we obtain the complete solution

(17) $$N_1^\pi(t)=e^t/(e^\tau-1)\ .$$

By an argument similar to that given in Section 2.1, we find that $N_m^\pi(t), m\ge 2$, can be obtained by recursively solving

(18) $$\frac{d}{dt}N_m^\pi(t)=\varphi_{\pi(m,t)}(N_m^\pi,N_{m-1}^\pi)(t)\ ,$$

with the boundary conditions (17) and

(19) $$N_m^\pi(\tau)=1+N_m^\pi(0)\ .$$

Note particularly that our problem differs from the single-job problem only in the (more challenging) boundary conditions; (18) has precisely the same form as (2).

Define $N_m^\star(t)=\sup_\pi N_m^\pi(t)$ and consider the optimality equations

(20) $$\dot n_m(t)=\min\{\varphi_1(n_m,n_{m-1})(t),\ \varphi_2(n_m,n_{m-1})(t)\},\ \ n_m(\tau)=1+n_m(0),$$

with $n_1(t)=e^t/(e^\tau-1)$.

We define the policy $\pi^\star$ according to

(21) $$\pi^\star(m,t)=\begin{cases}1, & \varphi_1(n_m,n_{m-1})<\varphi_2(n_m,n_{m-1})\ ,\\ 2, & \varphi_1(n_m,n_{m-1})\ge\varphi_2(n_m,n_{m-1})\ .\end{cases}$$

We have the following result analogous to Lemma 1 in Section 2.1.

Lemma 7. *For each m we have $n_m(t)=N_m^\star(t)$ and hence, $\pi^\star$ is optimal.*

Proof: The proof is by induction. We have $n_1(t) = N_1^\star(t)$ trivially. Let π be an arbitrary policy. As in the proof of Lemma 1 we have $\frac{d}{dt}(n_m - N_m^\pi)(t) < 0$ almost everywhere on the set $\{t : n_m(t) < N_m^\pi(t)\}$. It follows that if $n_m(t) < N_m^\pi(t)$ for some $t \leq \tau$ then $n_m(\tau) - n_m(0) < N_m^\pi(\tau) - N_m^\pi(0) = 1$, a contradiction. ∎

3.2. $\pi^\star(m,t)$ is monotone in t

In analogy with its use in Section 2.2, redefine the function $h_m(t)$ as

$$h_m(t) := 2n_{m-1}(t) - n_m(t)$$

This section shows that $h_m(t)$ is increasing in t for each m, so there can be at most one time τ_m at which shadowing begins. In other words, we have

Theorem 3. *The policy $\pi^\star$ is of threshold type.*

Proof: The proof uses Lemma 4. We need only show that $h_2(t)$ is increasing and that $h_m(\tau) \geq h_{m-1}(\tau)$. To see that $h_2(t)$ is increasing note that

$$\frac{d}{dt}h_2(t) = \max\{h_2(t) + n_1(0), 2h_2(t)\},$$

and since $h_2(\tau) - h_2(0) = 1$ we conclude that $\frac{d}{dt}h_2(t) > 0$ for $t \in (0,\tau)$. Now assume that, for $n \leq m-1$, $h_n(t)$ is increasing and that, contrary to our claim, $h_m(\tau) < h_{m-1}(\tau)$. Note that $h_m(\tau) - h_{m-1}(\tau) = h_m(0) - h_{m-1}(0)$ so $h_m(0) < h_{m-1}(0)$. The proof of Lemma 4 shows that $h_m(t)$ is decreasing on $[0,\tau]$, but this contradicts $h_m(\tau) - h_m(0) = 1$. We conclude that $h_m(\tau) \geq h_{m-1}(\tau)$ and that $h_m(t)$ is increasing for each m. ∎

To illustrate threshold behavior, we briefly describe a numerical technique for evaluating $n_m(t)$ and τ_m. Assume we have computed $\tau_1, \ldots, \tau_{m-1}$ and $n_1(t), \ldots, n_{m-1}(t)$. The computation of $n_m(t)$ is done by generating successive approximations, τ_m^k and $n_m^k(t)$, $k = 1, 2, \ldots$, to τ_m and $n_m(t)$, respectively. The initial approximations τ_m^1 and $n_m^1(t)$ are obtained by solving the optimality equation (using the method of Section 2.4)

$$\frac{d}{dt}n_m^1(t) = \min\{\varphi_1(n_m^1, n_{m-1})(t),\ \varphi_2(n_m^1, n_{m-1})(t)\},$$

with $n_m^1(\tau) = n_{m-1}(\tau)$. The optimality equation guarantees

$$n_m^1(\tau) - n_m^1(0) < n_{m-1}(\tau) - n_{m-1}(0) = 1\ .$$

The next approximation is calculated using the boundary condition

$$n_m^2(\tau) = n_m^1(\tau) + \frac{1 - (n_m^1(\tau) - n_m^1(0))}{1 - e^{-2\tau + \tau_m^1}}. \tag{22}$$

We determine this condition by solving the following differential equation for $\hat{n}^1_m(t)$

$$\hat{n}^1_m(t) = \begin{cases} \varphi_1(\hat{n}^1_m, n_{m-1})(t), & t < \tau^1_m \\ \varphi_2(\hat{n}^1_m, n_{m-1})(t), & t \geq \tau^1_m \end{cases}$$

with the boundary condition $\hat{n}^1_m(0) = n^1_m(0) + \delta$. Straightforward calculation shows that $\hat{n}^1_m(\tau) = n^1_m(\tau) + \delta e^{2\tau - \tau^1_m}$. Selecting δ such that $\hat{n}^1_m(\tau) - \hat{n}^1_m(0) = 1$ results in a boundary condition for the next iteration (viz., (22)) which satisfies: $n^2_m(0) > n^1_m(0)$ and $n^2_m(\tau) - n^2_m(0) \leq 1$. This procedure is repeated until $n^k_m(\tau) - n^k_m(0)$ is sufficiently close to 1, at which point we set $\tau_m = \tau^k_m$ and $n_m(t) = n^k_m(t)$.

In Figure **??** we show the values of τ_m for $m = 2, \ldots, 15$, where $\tau = 1.5$. As can be seen in the figure, τ_m is not monotone in m. In general, monotonicity in m does not even hold asymptotically, i.e., for all m sufficiently large, as we will see in the next section.

3.3. The threshold function τ_m

This subsection works out an asymptotic analysis of $n_m(t)$ that establishes the oscillatory behavior of τ_m. If $\tau \leq \ln 2$ then we find that shadowing is never called for and the optimal solution is given by $n_m(t) = mn_1(t)$. Henceforth we will assume $\tau > \ln 2$.

It is intuitively clear that if m is so large that the probability of completing a job is very nearly 1 then, roughly speaking, the optimal strategy will be to minimize the number of processors spent completing the job. Here we see the connection with the variant in Section 2.5. It is fairly easy to prove that $\tau_m \to \ln 2$ as $m \to \infty$. Knowing this, it is possible to obtain the asymptotic form of n_m.

Consider

$$\tilde{n}_m(t) := \begin{cases} c + \frac{m-1+e^t}{1+2(\tau-\ln 2)}, & 0 \leq t \leq \ln 2, \\ c + \frac{m+1+2(t-\ln 2)}{1+2(\tau-\ln 2)}, & \ln 2 \leq t \leq \tau, \end{cases}$$

where c is some fixed constant. It is easy to verify that the $\tilde{n}_m$, $m \geq 2$, satisfy equation (20). It follows from the above that $n_m(t) - n_{m-1}(t)$ and $\tilde{n}_m(t) - \tilde{n}_{m-1}(t)$ are asymptotically equal. But, although there is a constant c such that $n_m(t)$ is asymptotic to $\tilde{n}_m(t)$, it is not readily determined.

The oscillation of τ_m is really an artifact of the oscillation of $d_m(0) := n_m(0) - n_{m-1}(0)$, where $d_m(t)$ is redefined here in analogy with its use in Section 2.3. Since n_m is C^1, we have

$$e^{\tau_m} = \frac{2d_{m-1}(0)}{2d_{m-1}(0) - d_m(0)} + O((\tau_m - \tau_{m-1})^2)\,.$$

This equation is exact, i.e., the $O(.)$ error term vanishes, when $\tau_m \leq \tau_{m-1}$. As $m \to \infty$, we have $d_m(0) \to \frac{1}{1+2(\tau-\ln 2)}$, so if we define $\beta_m := \frac{1}{1+2(\tau-\ln 2)} - d_m(0)$, then

$$\tau_m = \ln 2 + (1 + 2(\tau - \ln 2))(\beta_m - \beta_{m-1}) + O(\beta_m^2 + \beta_{m-1}^2)\,,$$

Thus, we can study the asymptotic oscillation in τ_m by studying that of β_m.

It turns out that the oscillation in β_m persists even if the oscillation in τ_m is artificially removed. To see this, consider the effect of choosing a non-optimal policy by perturbing τ_m. We discover that, to first order, the policy is still optimal. If we let $n_m(T)(t)$ represent the value obtained by choosing the threshold time T with m processors, leaving n_{m-1} fixed, then it is easily verified that

$$\frac{d}{dT} n_m(T)(t)|_{T=\tau_m} = 0\,.$$

This arises from the fact that $n_m(t)$ is C^1, which in turn arises from the optimality of τ_m. Thus, if we start with some very large $m \geq M$ and simply set $\tau_m = \ln 2$, then the values of β_m will be altered only on the order of $(\tau_m - \ln 2)^2$. Hence, the oscillation of β_m will persist.

For $m > M$, define $g_m(t) := n_m(\ln 2)(t) - n_{m-1}(\ln 2)(t)$ and for $m = M$, define $g_m(t) := n_m(\ln 2)(t) - n_{m-1}(t)$. Then, for $m > M$, g_m satisfies

$$\begin{aligned} \dot{g}_m(t) &= \begin{cases} g_m(t) - g_{m-1}(0), & 0 \leq t \leq \ln 2, \\ 2(g_m(t) - g_{m-1}(t)), & \ln 2 \leq t \leq \tau, \end{cases} \\ g_m(0) &= g_m(\tau). \end{aligned}$$

Since the threshold is fixed we have $g_m(\ln 2) = 2g_m(0) - g_{m-1}(0) = 2g_m(\tau) - g_{m-1}(\tau)$. It is more convenient, therefore, to consider $f_m(t) := g_m(\tau - t)$ on the interval $[0, \tau_*]$ where $\tau_* := \tau - \ln 2$. Thus, we consider

$$\begin{aligned} \dot{f}_m(t) &:= 2(f_{m-1}(t) - f_m(t)), 0 \leq t \leq \tau_*, \\ f_m(\tau_*) &= 2f_m(0) - f_{m-1}(0)\,. \end{aligned} \tag{23}$$

In principle this equation can be solved as follows. Given any function $r(t)$ satisfying $\dot{r}(t) = 2[f_{m-1}(t) - r(t)]$ on $[0, \tau_*]$, we have $f_m(t) = r(t) + ae^{-2t}$, where a is chosen so that the boundary conditions are satisfied.

One way of obtaining a suitable r is first to represent $f_{m-1}(t)$ via its Fourier series. Given a sinusoid

$$f_{m-1}(t) = a \sin(ut + \varphi)\,.$$

we may then write

$$r(t) = a' \sin(ut + \varphi')$$

where $a' = (1 + u^2/4)^{-1/2} a$ and $\varphi' = \varphi - \arctan(u/2)$. It can be shown that if $f_{m-1} = a_{m-1} \sin(ut + \phi_{m-1})$ for u slightly smaller than $2\pi/\tau_*$, then $f_m \approx r$. The oscillation in $f_m(0)$ arises from this fact. To make this more rigorous we need to estimate the deviation from this approximate solution. This turns out to be a fairly complicated task. For this reason we have relegated the analysis to an appendix.

Briefly, the appendix proceeds by defining a class Φ of functions which are small perturbations of the approximate solution indicated above, and then by proving that if $f_{m-1} \in \Phi$ then $f_m \in \Phi$. For technical reasons we prove this only for the case $\tau_* = 2\pi$, i.e., $\tau = \ln 2 + 2\pi$. The formal definitions and the proof may be found in the appendix. The main result (see Theorem **??**) accounts for the oscillation in τ_m. The restriction on τ_* is not critical but it allows us to replace certain analytical bounds with numerical values, thereby significantly shortening the proof.

3.4. A generalization

Suppose we generalize the problem with infinitely jobs so that every job but the first has its running time drawn independently at random from a known distribution Q. The first, or *current*, job must be finished first and it is known to have τ time units left to run. The remaining jobs must be completed in a given order, and we do not learn a job's duration until it becomes the current job. The previous problem is the special case where Q is concentrated at τ. We assume that Q has a positive expected value.

Let $N_m(\tau, t)$ denote the expected number of jobs that will be completed under an optimal policy given m processors and a current job of duration τ with t units of accrued processing time. By arguments similar to those in Section 2.1, we find that for $m \geq 2$ we can obtain $N_m(\tau, t)$ as the solution $n_m(\tau, t)$ of

$$\begin{aligned} \frac{d}{dt} n_m(\tau, t) &= \min\{\varphi_1(n_m(\tau, t), n_{m-1}(\tau, t)), \varphi_2(n_m(\tau, t), n_{m-1}(\tau, t))\} \\ & \quad n_m(\tau, \tau) = 1 + \hat{n}_m(0), \end{aligned}$$

where

$$\hat{n}_m(0) := E_Q(n_m(t, 0))$$

and

$$E_Q(f(t)) := \int f(t) dQ(t) \, .$$

Directly, we have $n_1(\tau, t) = e^{t-\tau}[1 + \hat{n}_1(0)]$, where $\hat{n}_1(0) = E_Q(e^{-t})/[1 - E_Q(e^{-t})]$.

Theorem 4. *An optimal policy is of threshold type.*

Proof: We define $h_m(\tau, t) := 2n_{m-1}(\tau, t) - n_m(\tau, t)$ and $\hat{h}_m(0) := E_Q(h_m(t, 0))$, and then show that $h_2(\tau, t)$ is increasing and that $\hat{h}_m(0) \geq \hat{h}_{m-1}(0)$. An application of Lemma (4) then completes the proof.

We show that $h_2(\tau, t)$ is increasing by contradiction. Since in general we have

$$\frac{d}{dt} h_2(\tau, t) = \max\{h_2(\tau, t) + n_1(\tau, 0), 2h_2(\tau, t)\},$$

we see that $h_2(\tau, t)$ is monotone. Further, if $h_2(\tau, t)$ is non-increasing for some τ, then $h_2(\tau, \tau) = 1 + \hat{h}_2(0) < 0$ and we can conclude that $h_2(\tau, t) < 0$ for all t and τ. This implies that shadowing never occurs when there are two processors and we can solve for $n_2(\tau, t)$ explicitly. In particular, we obtain

$$n_2(\tau, t) = e^{t-\tau}(1 + \hat{n}_2(0)) + (1 - e^{t-\tau})n_1(\tau, 0)$$

from which we derive

$$\hat{n}_2(0) = \frac{E_Q(e^{-t})}{1 - E_Q(e^{-t})} + \frac{E_Q(e^{-t}) - E_Q(e^{-2t})}{(1 - E_Q(e^{-t}))^2} \,.$$

Then

$$\begin{aligned} \hat{h}_2(0) + 1 &= \frac{1}{1 - E_Q(e^{-t})} + \frac{E_Q(e^{-2t}) - E_Q(e^{-t})}{(1 - E_Q(e^{-t}))^2} \\ &= \frac{(1 - E_Q(e^{-t}))^2 + E_Q((e^{-t} - E_Q(e^{-t}))^2)}{(1 - E_Q(e^{-t}))^2} \\ &> 0\,, \end{aligned}$$

which is a contradiction.

To see that $\hat{h}_m(0) \geq \hat{h}_{m-1}(0)$, suppose the contrary. Then, for some minimal m, we have

$$\int (h_m(\tau, 0) - h_{m-1}(\tau, 0))dQ(\tau) < 0.$$

Hence, for some τ we have $h_m(\tau, 0) - h_{m-1}(\tau, 0)) \leq \hat{h}_m(0) - \hat{h}_{m-1}(0) < 0$. But then, as the proof of Lemma 4 shows, we can conclude that $h_m(\tau, \tau)$ is decreasing. Thus, we have $h_m(\tau, \tau) - h_{m-1}(\tau, \tau) < \hat{h}_m(\tau, 0) - \hat{h}_{m-1}(\tau, 0)$, which is a contradiction. ■

Under mild assumptions on Q and $\tau > \ln 2$, there will be an optimal threshold which will converge to $\ln 2$ as $m \to \infty$, as in the previous problem. Furthermore, $n_m(\tau, t) - n_{m-1}(\tau, t)$ will converge to some fixed function. As we are dealing here with a generalization of the previous problem, the convergence of the optimal threshold need not be monotone in general.

4. Conclusions

In an obvious generalization to the variant of Section 3.4, the current job's running time is also a random sample from Q. In spite of our efforts, which we intend to continue, this 'completely stochastic' problem remains open.

As noted in the introduction, more general settings are obtained by allowing for a nonzero, but different failure rate for idling processors, and by allowing for repairs and re-use of failed processors. Also, we have assumed that processor failures are instantly and reliably detected. It would be more realistic to assume that the detection mechanism itself is unreliable and attempt to model this situation. We have also assumed that there is no cost associated with starting up the shadowing processor. These and other complicating factors suggest many new lines of inquiry. Another issue not considered in this paper is the importance of the completion time of the job, given that it is completed. For example, it may be the case that if a job takes too long to finish then its utility is diminished. The tradeoff between successful completion and timely completion might be modeled by a discounted reward which is a function of the job completion time. Intuitively, this would provide an incentive to begin shadowing earlier than determined by the policies in this paper in order to maximize the expected reward.

References

[1] V. I. Arnold (1981). *Ordinary Differential Equations*, MIT Press.

[2] Richard E. Barlow and Frank Proschan (1996). *Mathematical Theory of Reliability*, SIAM, Philadelphia.

[3] L. B. Boguslavsky, E. G. Coffman, Jr., E. N. Gilbert, and Alexander Y. Kreinin (1992). Scheduling Checks and Saves, *ORSA Journal on Computing*, Vol. 4, No. 1.

[4] P. F. Chimento, Jr., and K. S. Trivedi (1993). The Completion Time of Programs on Processors Subject to Failure and Repair, *IEEE Transactions on Computers*, 42(10),1184–1194.

[5] E. G. Coffman, Jr. and E. N. Gilbert (1990). Optimal strategies for Scheduling Saves and Preventive Maintenance, *IEEE Transactions on Reliability*, 39, 9–18.

[6] A. Duda (1983). The Effects of Checkpointing on Program Execution Time, *Information Processing Letters*, 16, 221–229.

[7] R. Geist, R. Reynolds, and J. Westall (1988). Selection of a Checkpoint Interval in a Critical-Task Environment, *IEEE Transactions on Reliability*, 37(4), 395–400.

[8] A. Goyal, V. Nicola, A. Tantawi, and K. Trivedi (1987). Reliability of Systems with Limited Repairs, *IEEE Transactions on Reliability*, 36, 202–207.

[9] B. Kalyanasundaram and K. R. Pruhs (1994). Fault-Tolerant Scheduling (Extended Abstract), *Proceedings, Symp. Th. Comput.*, ACM Press, New York, 115–124.

[10] V. G. Kulkarni, V. F. Nicola, and K. S. Trivedi (1990). Effects of Checkpointing and Queueing on Program Performance, *Commun. Statist.-Stochastic Models*, 6(4), 615–648.

[11] V. G. Kulkarni, V. F. Nicola, and K. S. Trivedi (1987). The Completion Time of a Job on Multimode Systems, *Adv. Appl. Prob.*, 19, 932–954.

[12] P. L’Ecuyer and J. Malenfant (1988). Computing Optimal Checkpointing Strategies for Rollback and Recovery Systems, *IEEE Transactions on Computers*, 37(4), 491–496.