J. Appl. Prob. Spec. Vol. **48A,** 195–207 (2011)

NEW FRONTIERS IN APPLIED PROBABILITY

A Festschrift for SØREN ASMUSSEN
Edited by P. GLYNN, T. MIKOSCH and T. ROLSKI

Part 4. Simulation

CHECKPOINTING FOR THE RESTART PROBLEM IN MARKOV NETWORKS

LESTER LIPSKY, *University of Connecticut*

Department of Computer Science and Engineering, University of Connecticut, 371 Fairfield Road, Storrs, CT 06269-2155, USA. Email address: lester@engr.uconn.edu

DEREK DORAN, *University of Connecticut*

Department of Computer Science and Engineering, University of Connecticut, 371 Fairfield Road, Storrs, CT 06269-2155, USA.

SWAPNA GOKHALE, *University of Connecticut*

Department of Computer Science and Engineering, University of Connecticut, 371 Fairfield Road, Storrs, CT 06269-2155, USA.

APPLIED PROBABILITY TRUST
AUGUST 2011

CHECKPOINTING FOR THE RESTART PROBLEM IN MARKOV NETWORKS

BY LESTER LIPSKY, DEREK DORAN AND SWAPNA GOKHALE

Abstract

We apply the known formulae of the RESTART problem to Markov models of software (and many other) systems, and derive new equations. We show how checkpoints might be included, with their resultant performance under RESTART. The result is a complete procedure for finding the mean, variance, and tail behavior of the job completion time as a function of the failure rate. We also provide a detailed example.

Keywords: Checkpoint; exponential failure; Markov model; RESTART; performance; power-tailed distribution; subexponential distribution; asymptotic behavior

2010 Mathematics Subject Classification: Primary 60J28
Secondary 60K10

1. Introduction

There are three general scenarios on how to continue when a system crashes during execution that have often been discussed in the literature. One possibility is to continue from the point of failure after the system has been repaired. This was labeled *preemptive resume* in [9] and *RESUME* in [13]. Another possibility is to discard the results so far and start from scratch, thereby replacing everything. This is alternately called *REPLACE* or *preemptive repeat different*. Both of these scenarios are amenable to mathematical treatment and can usually be analyzed by Markov models. The third scenario, however, has proven difficult to treat. Here, all the accumulated work is lost and the job must start from the beginning, retracing all steps previously taken. This is alternately called *RESTART* or *preemptive repeat identical*. Clearly, it is not memoryless, and Markov methods have fallen short.

Still, in recent years there has been much progress in analyzing systems under RESTART. Restricting ourselves to *exponential failure distributions* with failure rate β, we now know that this can lead to *subexponential* completion times. Let T be the random variable (RV) denoting the time for a job to complete if failures cannot occur, and let $X(\beta)$ be the time for completion, including restarts after failures (but not repair time). Any particular job ($T = t$) will take, on average (see, e.g. [5] or [13])

$$\mathrm{E}[X(\beta)] = \frac{\mathrm{e}^{\beta t} - 1}{\beta}.$$

Even if T has a distribution with finite support, $X(\beta)$ will be exponentially distributed. Practitioners have been forced to employ various strategies to counter this dismal behavior. The most common is *checkpointing*.

Over the past thirty years, architecture-based analysis of software reliability has gained prominence [11], [14]. In the architecture-based approach, the control flow graph of a software application is mapped to a state space model (e.g. a continuous-time Markov chain (CTMC)) [6].

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

Each node of the state space model represents the execution of a block of code (one or more functions or methods). A CTMC is an appropriate choice to represent the application architecture if the execution time in each code block is exponentially distributed, or can itself be represented by a matrix exponential distribution. These models have long been applied to the performance analysis of computer software [11] (among many other uses), where random failures are often common. Here we derive formulae for evaluating the performance of such systems under RESTART, including checkpointing. We provide expressions for the mean, variance, and tail behavior of the job completion time. We also provide a detailed example to demonstrate how all the parts fit.

2. RESTART and checkpointing

Let $F(t)$, $f(t)$, and $\bar{F}(t)$ be the distribution, density, and complementary distribution functions for T, respectively. Assume that the failure distribution is exponential, with failure rate β, and that $X(\beta)$, under RESTART, has probability distribution, probability density, and complementary distribution functions $H(x)$, $h(x)$, and $\bar{H}(x)$, respectively. We now present two definitions.

Definition 1. Let

$$\alpha := \sup\left\{\ell \;\middle|\; \int_0^\infty x^\ell h(x)\,\mathrm{d}x < \infty\right\}. \tag{1}$$

Then $X(\beta)$ is *power tailed* (PT) with index α if $\alpha < \infty$.

Clearly, if $\alpha < 1$ then $\mathrm{E}[X(\beta)] = \infty$, and even if $\alpha > 1$ but $\alpha < 2$, $\mathrm{E}[X(\beta)^2] = \infty$, and $X(\beta)$ has infinite variance.

Definition 2. Let

$$\lambda_\mathrm{s} := \sup\left\{\lambda \;\middle|\; \int_0^\infty \exp(\lambda t) f(t)\,\mathrm{d}t < \infty\right\}. \tag{2}$$

Then $f(t)$ has an *exponential tail* with parameter λ_s if $0 < \lambda_\mathrm{s} < \infty$. If $\lambda_\mathrm{s} = 0$ then $f(t)$ is *subexponential*. (For more precise definitions, see [1, A5, p. 412]).

Even if T has finite support (e.g. $T = t$, a constant) the time under RESTART, $X(\beta)$, will have an exponential tail. Asmussen *et al.* [3] (see also [8]) showed that if T has infinite support then $X(\beta)$ will be subexponential. Furthermore, it was shown in [3] and [13] that if T has an exponential tail with parameter λ_s then $X(\beta)$ will be PT with index

$$\alpha = \frac{\lambda_\mathrm{s}}{\beta}. \tag{3}$$

Clearly, as β becomes bigger, α becomes smaller, and the system behavior becomes more and more unstable.

For many years, system designers have attempted to alleviate the instability problem by *checkpointing*. During execution, a program is interrupted at some predetermined point so that sufficient information can be placed in a nonvolatile place, and then the program continues. If failure subsequently occurs, the job can return to this point without having to go back to the beginning. The optimal strategy (if the cost is negligible) would be to interrupt at fixed time intervals. But this is usually inconvenient and time costly, for the job could be in any state, thus necessitating transfer of an inordinate amount of information in order for the job to continue from this point if failure subsequently occurs. Other strategies are also used. For instance, one

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

can find a place where the amount of information needed to restart would be minimal, and place the checkpoint there, usually where a particular procedure ends. This is what we attempt here (see, e.g. [13] for other strategies).

3. Markov models with exponential failures and checkpointing

In recent years, researchers have been analyzing the performance of computer software by constructing Markov models based on flowcharts (see, e.g. [11] and [14]). We first discuss such models, then include failures, and, finally, after some mathematical preliminaries, add checkpoints.

3.1. Markov models of software (MMS): preliminary performance analysis

Each of the M nodes of a given flowchart is a program section that takes some time to execute. The flowchart becomes an M-dimensional Markov matrix, $\boldsymbol{P}$. If it is assumed that the service time at each node is exponentially distributed with rate parameter $\mu_i := [\boldsymbol{M}]_{ii} > 0$, and there is a path to exit the system from each node, then the usual mathematical formulae (see, e.g. [10] for full details) apply for obtaining the distribution of the time to execution, T. That is, let $\boldsymbol{p}$ be the *entrance vector*, where $[\boldsymbol{p}]_i$ is the probability that execution begins at node i, and define

$$\boldsymbol{B} := \boldsymbol{M}(\boldsymbol{I} - \boldsymbol{P}).$$

Then

$$\bar{F}(t) := \Pr[T > t] = \boldsymbol{p} \exp(-t\boldsymbol{B})\boldsymbol{\varepsilon}', \tag{4}$$

where $\boldsymbol{\varepsilon}'$ is the M-dimensional column vector of 1s. Furthermore, let $\boldsymbol{V} := \boldsymbol{B}^{-1}$. Then all the moments can be found by

$$\mathrm{E}[T^\ell] = \int_0^\infty t^\ell f(t)\,\mathrm{d}t = \ell!\,\boldsymbol{p}\boldsymbol{V}^\ell\boldsymbol{\varepsilon}'. \tag{5}$$

These are called matrix exponential (ME) distributions (in this case, phase distributions because $\mu_i > 0$ and $[\boldsymbol{P}]_{ij} \geq 0$; see [12, p. 45]), and have exponential tails, as defined by (2). From the spectral decomposition theorem (see, e.g. [10, p. 143]), it is not hard to show that

$$\lambda_{\mathrm{s}} = \min[|\lambda_i|], \tag{6}$$

where $\{\lambda_i \mid 1 \leq i \leq M\}$ is the set of eigenvalues of $\boldsymbol{B}$ whose eigenvectors are not orthogonal to $\boldsymbol{p}$ or $\boldsymbol{\varepsilon}'$.

It is clear that if this system is subject to failure, the time to completion, $X(\beta)$, will be PT distributed, with, from (3), $\alpha = \lambda_{\mathrm{s}}/\beta$. Because $H(x)$ is not ME, we cannot find out what it looks like except by approximate methods such as numerical inversion of Laplace transforms (see, e.g. [5]). However, $\mathrm{E}[X(\beta)]$ and $\mathrm{E}[X(\beta)^2]$ can be calculated easily with the following formulae, taken from [13] (higher moments can be found, but with increasing difficulty):

$$\mathrm{E}[X(\beta)] = \boldsymbol{p}[\boldsymbol{V}(\boldsymbol{I} - \beta\boldsymbol{V})^{-1}]\boldsymbol{\varepsilon}' \tag{7}$$

and

$$\mathrm{E}[X(\beta)^2] = 2\boldsymbol{p}[\boldsymbol{V}^2(\boldsymbol{I} - 2\beta\boldsymbol{V})^{-2}(\boldsymbol{I} - \beta\boldsymbol{V})^{-1}]\boldsymbol{\varepsilon}'. \tag{8}$$

Note that $(\boldsymbol{I} - \beta\boldsymbol{V})$ is not invertible when $\beta = \lambda_{\mathrm{s}}$, and can yield negative results when $\beta > \lambda_{\mathrm{s}}$. In other words, the expression for $\mathrm{E}[X(\beta)]$ is meaningless when $\alpha = \lambda_{\mathrm{s}}/\beta \leq 1$, with a similar statement for $\mathrm{E}[X(\beta)^2]$ when $\alpha \leq 2$.

Our next task is to insert a checkpoint into the MMS. But before describing how this is done in detail we must first establish a useful theorem.

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

3.2. Continuous Markov chains with two absorbing states

Consider the $(M+2)$-dimensional Markov matrix $\bar{\boldsymbol{P}}$ with two absorbing states, a and b. That is,

$$\bar{\boldsymbol{P}}\bar{\boldsymbol{\varepsilon}}' = \bar{\boldsymbol{\varepsilon}}' \quad \text{and} \quad (\bar{\boldsymbol{P}})_{aa} = (\bar{\boldsymbol{P}})_{bb} = 1,$$

where $\bar{\boldsymbol{\varepsilon}}'$ is an $(M+2)$-dimensional column vector of 1s. Next delete the rows and columns of a and b to produce the M-dimensional matrix $\boldsymbol{P}$. Then

$$[\boldsymbol{Z}]_{ij} := [(\boldsymbol{I} - \boldsymbol{P})^{-1}]_{ij}$$

is the expected number of visits to j before absorption, given that the chain started at i. Now define the M-dimensional column vectors

$$(\boldsymbol{q}'_a)_i := \bar{P}_{ia} \quad \text{and} \quad (\boldsymbol{q}'_b)_i := \bar{P}_{ib}, \quad \text{where } i \neq a, b.$$

These are the probability vectors of being absorbed by a and b, respectively. It follows that the ith components of

$$\boldsymbol{\varepsilon}'_a := \boldsymbol{Z}\boldsymbol{q}'_a \quad \text{and} \quad \boldsymbol{\varepsilon}'_b := \boldsymbol{Z}\boldsymbol{q}'_b$$

are the probabilities that the process will end at a or b, respectively, given that the process started at i. Note that $\boldsymbol{\varepsilon}'_a + \boldsymbol{\varepsilon}'_b = \boldsymbol{\varepsilon}'$.

Let $\boldsymbol{p}_{\mathrm{o}}$ be the entrance vector. Then

$$p_a = \boldsymbol{p}_{\mathrm{o}}\boldsymbol{\varepsilon}'_a \quad \text{and} \quad p_b = \boldsymbol{p}_{\mathrm{o}}\boldsymbol{\varepsilon}'_b, \quad \text{where } p_a + p_b = 1,$$

are the probabilities that the process will be absorbed by a or b, respectively. As in Section 3.1, let T be the RV denoting the time until absorption, while T_a and T_b are the conditional RVs denoting the time until absorption at a and b. Then $\boldsymbol{B} = \boldsymbol{M}(\boldsymbol{I} - \boldsymbol{P})$ is the generator of the distribution function for T, whose complementary distribution function is given by (4), where $\boldsymbol{p}_{\mathrm{o}}$ is used instead of $\boldsymbol{p}$.

It is well known that $[\boldsymbol{p}_{\mathrm{o}} \exp(-\boldsymbol{B}t)]_i$ is the probability that absorption has not occurred by time t, and the system is in state i. This leads to the following result.

Theorem 1. *Let $\boldsymbol{q}'_u$, $\boldsymbol{\varepsilon}'_u$, $\boldsymbol{p}_{\mathrm{o}}$, $\boldsymbol{B}$, and $\boldsymbol{V}$, where $u \in \{a, b\}$, be defined as above. Then T_u has distribution*

$$\bar{F}_u(t) := \Pr[T_u > t] = \frac{\boldsymbol{p}_{\mathrm{o}} \exp(-\boldsymbol{B}t)\boldsymbol{\varepsilon}'_u}{p_u}, \qquad u = a, b.$$

The moments of these distributions follow easily from (5):

$$\mathrm{E}[T_u^{\ell}] = \frac{\ell!\, \boldsymbol{p}_{\mathrm{o}}[\boldsymbol{V}^{\ell}]\boldsymbol{\varepsilon}'_u}{p_u}.$$

We then say that $\bar{F}_u(t)$ is generated by the triplet $\langle \boldsymbol{p}_{\mathrm{o}}, \boldsymbol{B}, \boldsymbol{\varepsilon}'_u \rangle$.

It is interesting that the conditional probabilities require that the final vector be different from $\boldsymbol{\varepsilon}'$. This implies that the components of these final vectors, $[\boldsymbol{\varepsilon}'_u]$, carry meaning. They are the probabilities that the process will be absorbed by u, given the starting state. If there is only one absorbing state then all probabilities of absorption are 1, hence $\boldsymbol{\varepsilon}'$. (Thus ends a 30-year debate that one of the authors has had with Appie van de Liefvoort [15] over whether $\boldsymbol{\varepsilon}'$ must always be a vector of 1s—vdL wins.)

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

Furthermore, even though both $\bar{F}_a(t)$ and $\bar{F}_b(t)$ have the same $\boldsymbol{p}_o$ and $\boldsymbol{B}$ as $\bar{F}(t)$, they often have completely different properties. They may even have different asymptotic behavior. This can happen if the left eigenvector of $\boldsymbol{B}$ belonging to λ_s ($\boldsymbol{u}_s\boldsymbol{B} = \lambda_s\boldsymbol{u}_s$) is orthogonal to $\boldsymbol{\varepsilon}'_a$ and/or $\boldsymbol{\varepsilon}'_b$ (e.g. $\boldsymbol{u}_s\boldsymbol{\varepsilon}'_a = \boldsymbol{0}$). This is not an unusual occurrence. Let

$$\boldsymbol{S}_a := \{s_i,\ i \neq a \mid [\boldsymbol{P}^k]_{ia} = 0 \text{ for all } k > 0\}$$

(i.e. there is no path from s_i to the absorbing state s_a). If $\ell := |\boldsymbol{S}_a| > 0$ then there must be at least ℓ left eigenvectors of $\boldsymbol{B}$ that are orthogonal to $\boldsymbol{\varepsilon}'_a$. If $\boldsymbol{u}_s$ is one of them then

$$\lambda_a := \min\{\lambda_i \mid s_i \notin \boldsymbol{S}_a\} > \lambda_s$$

will be the exponential tail parameter for $\bar{F}_a$. Thus, λ_a, λ_b, and λ_s can all be different, and $\bar{F}_a$, $\bar{F}_b$, and $\bar{F}$ can all have different asymptotic behaviors. We make use of these equations in the next section.

3.3. Inserting a checkpoint in MMS (MMSC)

Consider the MMS described in Section 3.1. First select some node m to be the node that is followed by a system checkpoint (ideally, we might select each node, one at a time, and test to see which node yields the best performance). Then

$$q_m = [\boldsymbol{q}']_m := [(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{\varepsilon}']_m$$

is the probability that execution will end after finishing at m. Now add one row and one column to $\boldsymbol{P}$ at index $M+1$, representing the system checkpoint state, to produce the matrix $\boldsymbol{P}_c$ with the following specifications. For $i \neq m$, $M+1$, and $j \neq M+1$,

$$[\boldsymbol{P}_c]_{ij} = \boldsymbol{P}_{ij}, \qquad [\boldsymbol{P}_c]_{i,M+1} = 0, \qquad [\boldsymbol{P}_c]_{mi} = 0, \qquad [\boldsymbol{P}_c]_{m,M+1} = 1 - q_m,$$
$$[\boldsymbol{P}_c]_{M+1,k} = 0 \quad \text{for all } k.$$

This process can be visualized in the following way. Suppose that a customer enters the network and goes to node i ($[\boldsymbol{p}_o]_i$), and then wanders from node to node ($[\boldsymbol{P}_c]_{ij}$) until he/she either leaves ($[\boldsymbol{q}']_j$) and the job is finished or goes to node m ($[\boldsymbol{P}_c]_{jm}$). At node m he/she either finishes (q_m) or goes to the checkpoint node ($1 - q_m$). When work is completed at the checkpoint, operation is suspended temporarily ($[\boldsymbol{P}_c]_{M+1,i} = 0$). The checkpoint can only be reached through m ($[\boldsymbol{P}_c]_{i,M+1} = 0$, $i \neq m$).

We have defined a Markov chain with two absorbing states: a subscript 'e' will be used to indicate the end absorbing state and a subscript 'c' will be used to indicate the checkpoint absorbing state. We can use the formulae presented in Theorem 1 once we have identified $\boldsymbol{q}'_e$ and $\boldsymbol{q}'_c$. The exit vector $\boldsymbol{q}'_e$ is the same as for the original model, with the additional component $[\boldsymbol{q}'_e]_{M+1} = 0$, so

$$[\boldsymbol{q}'_e]_i = [(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{\varepsilon}']_i, \quad \text{but} \quad [\boldsymbol{q}'_e]_{(M+1)} = 0,$$

while $[\boldsymbol{q}'_c]_i = 0$ for $i \leq M$ and $[\boldsymbol{q}'_c]_{M+1} = 1$. As before, define the $(M+1)$-dimensional matrix

$$\boldsymbol{Z}_c = (\boldsymbol{I} - \boldsymbol{P}_c)^{-1}.$$

Then

$$\boldsymbol{\varepsilon}'_e = \boldsymbol{Z}_c\boldsymbol{q}'_e \quad \text{and} \quad \boldsymbol{\varepsilon}'_c = \boldsymbol{Z}_c\boldsymbol{q}'_c.$$

From this we obtain the probability of finishing without reaching the checkpoint, $p_{oe} := \boldsymbol{p}_o\boldsymbol{\varepsilon}'_e$, and the probability of reaching the checkpoint $p_{oc} := \boldsymbol{p}_o\boldsymbol{\varepsilon}'_c$. Let T_{oe} and T_{oc} be the conditional

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

RVs for the times to finish without and with checkpointing, respectively. Furthermore, define the diagonal matrices

$$[\boldsymbol{M}_{\mathrm{c}}]_{ii} = [\boldsymbol{M}]_{ii} \quad \text{and} \quad [\boldsymbol{M}_{\mathrm{c}}]_{M+1,M+1} = \mu_{\mathrm{c}},$$

where $t_{\mathrm{c}} = 1/\mu_{\mathrm{c}}$ is the mean time to process a checkpoint. Then

$$\boldsymbol{B}_{\mathrm{c}} := \boldsymbol{M}_{\mathrm{c}}(\boldsymbol{I} - \boldsymbol{P}_{\mathrm{c}})$$

and

$$\bar{F}_{\mathrm{o}u}(t) := \Pr[T_{\mathrm{o}u} > t] = \frac{\boldsymbol{p}_{\mathrm{o}} \exp(-t\boldsymbol{B}_{\mathrm{c}})\boldsymbol{\varepsilon}'_u}{p_{\mathrm{o}u}} \quad \text{for } u \in \{\mathrm{e}, \mathrm{c}\}.$$

If the execution of the software takes the path described by oe, then the process is over. But if the path leads instead to m, and thence to $M + 1$ (the checkpoint), then execution must subsequently return to the system and transition between phases again. But now, the *restart vector* corresponds to where the execution would have gone if it had left the original phase m in $\boldsymbol{P}$ but did not leave the system. That is,

$$\boldsymbol{p}_{\mathrm{c}} := \frac{[P_{m1}, P_{m2}, \ldots, P_{mM}, 0]}{1 - q_m}.$$

Then $p_{\mathrm{ce}} := \boldsymbol{p}_{\mathrm{c}}\boldsymbol{\varepsilon}'_{\mathrm{e}}$ is the probability that execution will finish without returning to m, while $p_{\mathrm{cc}} := \boldsymbol{p}_{\mathrm{c}}\boldsymbol{\varepsilon}'_{\mathrm{c}}$ is the probability that it will return to execute another checkpoint. The distributions for the two RVs T_{ce} and T_{cc} are

$$\bar{F}_{\mathrm{c}u}(t) := \Pr[T_{\mathrm{c}u} > t] = \frac{\boldsymbol{p}_{\mathrm{c}} \exp(-t\boldsymbol{B}_{\mathrm{c}})\boldsymbol{\varepsilon}'_u}{p_{\mathrm{c}u}} \quad \text{for } u \in \{\mathrm{e}, \mathrm{c}\}.$$

What we have described is an embedded Markov chain with four nodes whose sojourn time distributions are given by the four functions above. The transition matrix for this process is

$$\hat{\boldsymbol{P}}_{\mathrm{c}} := \begin{array}{c} \\ \mathrm{oe} \\ \mathrm{oc} \\ \mathrm{ce} \\ \mathrm{cc} \end{array} \begin{array}{c} \begin{array}{cccc} \mathrm{oe} & \mathrm{oc} & \mathrm{ce} & \mathrm{cc} \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & p_{\mathrm{ce}} & p_{\mathrm{cc}} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & p_{\mathrm{ce}} & p_{\mathrm{cc}} \end{bmatrix} \end{array}, \quad \text{with} \quad \hat{\boldsymbol{p}}_{\mathrm{c}} := [p_{\mathrm{oe}}, p_{\mathrm{oc}}, 0, 0]. \tag{9}$$

This is simple enough so that the inverse of $(\hat{\boldsymbol{I}} - \hat{\boldsymbol{P}}_{\mathrm{c}})$ can be written down without difficulty:

$$\hat{\boldsymbol{Z}}_{\mathrm{c}} := (\hat{\boldsymbol{I}} - \hat{\boldsymbol{P}}_{\mathrm{c}})^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & p_{\mathrm{cc}}/p_{\mathrm{ce}} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1/p_{\mathrm{ce}} \end{bmatrix}.$$

This will be useful below when $\mathrm{E}[X_{\mathrm{c}}(\beta)]$ is computed, but it also yields the expected number of times the checkpoint is executed, another important design parameter. Since

$$\hat{\boldsymbol{p}}_{\mathrm{c}}\hat{\boldsymbol{Z}}_{\mathrm{c}} = \left[p_{\mathrm{oe}}, p_{\mathrm{oc}}, p_{\mathrm{oc}}, \frac{p_{\mathrm{oc}}p_{\mathrm{cc}}}{p_{\mathrm{ce}}}\right]$$

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

is the vector of the expected number of visits to each node, and since nodes oc and cc include visits to c, then

$$\mathrm{E}[N_{\mathrm{c}}] = p_{\mathrm{oc}} + \frac{p_{\mathrm{oc}}\,p_{\mathrm{cc}}}{p_{\mathrm{ce}}} = \frac{p_{\mathrm{oc}}}{p_{\mathrm{ce}}}. \tag{10}$$

This can also be computed by recognizing that node cc is reached a geometric number of times and that node oc is reached at most once, with probability p_{oc}.

As it stands $\hat{\boldsymbol{P}}_{\mathrm{c}}$, together with its node ME distributions $\{F_u\}$, is an ME representation, and, for $t_{\mathrm{c}} = 0$, it generates the same distribution as the original model generated by $\langle \boldsymbol{p}, \boldsymbol{B}\rangle$. For $t_{\mathrm{c}} > 0$, it generates the distribution for T_{c}, the time to finish, including the multiple times the checkpoint is executed (but with no failures). A straightforward representation of the distribution for T_{c} when $t_{\mathrm{c}} > 0$ (but $\beta = 0$) can be constructed by replacing the last row of $\boldsymbol{P}_{\mathrm{c}}$ with $\boldsymbol{p}_{\mathrm{c}}$. This is equivalent to replacing $\boldsymbol{P}_{\mathrm{c}}$ with

$$\bar{\boldsymbol{P}}_{\mathrm{c}} = \boldsymbol{P}_{\mathrm{c}} + \boldsymbol{q}'_{\mathrm{c}}\boldsymbol{p}_{\mathrm{c}},$$

thus allowing the process to continue after checkpointing. Then $\langle \boldsymbol{p}_{\mathrm{c}}, \bar{\boldsymbol{B}}_{\mathrm{c}}\rangle$ generates $\Pr[T_{\mathrm{c}} > t]$, where

$$\bar{\boldsymbol{B}}_{\mathrm{c}} = \boldsymbol{M}_{\mathrm{c}}(\boldsymbol{I} - \bar{\boldsymbol{P}}_{\mathrm{c}}) = \boldsymbol{M}_{\mathrm{c}}(\boldsymbol{I} - \boldsymbol{P}_{\mathrm{c}} - \boldsymbol{q}'_{\mathrm{c}}\boldsymbol{p}_{\mathrm{c}}) = \boldsymbol{B}_{\mathrm{c}} - \boldsymbol{M}_{\mathrm{c}}\boldsymbol{q}'_{\mathrm{c}}\boldsymbol{p}_{\mathrm{c}} = \boldsymbol{B}_{\mathrm{c}}[\boldsymbol{I} - \boldsymbol{\varepsilon}'_{\mathrm{c}}\boldsymbol{p}_{\mathrm{c}}].$$

The proof will not be given here because this formulation cannot be used if $\beta > 0$. However, we present an expression for $\mathrm{E}[T_{\mathrm{c}}]$, the time to finish with checkpoints active, but no failures (see [10, p. 167] for algebraic details):

$$\mathrm{E}[T_{\mathrm{c}}] = \mathrm{E}[X_{\mathrm{c}}(\beta = 0)] = \boldsymbol{p}_{\mathrm{o}}\bar{\boldsymbol{V}}_{\mathrm{c}}\boldsymbol{\varepsilon}' = p_{\mathrm{oc}}\left(\mathrm{E}[T_{\mathrm{oc}}] + \mathrm{E}[T_{\mathrm{ce}}] + \frac{p_{\mathrm{cc}}}{p_{\mathrm{ce}}}\,\mathrm{E}[T_{\mathrm{cc}}]\right) + p_{\mathrm{oe}}\,\mathrm{E}[T_{\mathrm{oe}}].$$

Note that this is different from $\boldsymbol{p}_{\mathrm{o}}\boldsymbol{V}_{\mathrm{c}}\boldsymbol{\varepsilon}'$, which is the mean time to either finishing without checkpointing or visiting the checkpoint just once and then stopping.

3.4. The MMSC with exponential failures

We are now able to apply our model with checkpoints to system failures under RESTART. As already mentioned, there is no way of obtaining the distribution of $X(\beta)$ without performing a heroic numerical effort to invert its Laplace transform. But we have the tools to find the moments of the distribution, the first two of which are easy to obtain using (7) and (8) with $\langle \boldsymbol{p}_{\mathrm{o}}, \boldsymbol{B}\rangle$. Asmussen *et al.* [3] showed how to find the asymptotic behavior, but we will be satisfied here with only the power of the tail, $\alpha = \lambda_{\mathrm{s}}/\beta$. This is important because it tells us how big the failure rate β can get before the system becomes unstable. That is, if $\beta \geq \lambda_{\mathrm{s}}$ then $\mathrm{E}[X(\beta)] = \infty$, and if $\beta \geq \lambda_{\mathrm{s}}/2$ then $\mathrm{E}[X^2(\beta)] = \infty$.

Evaluating the moments and α_{c} for $X_{\mathrm{c}}(\beta)$ requires more effort, but not as much as we might expect. Consider the four-node system prepared in the previous section. If it should fail during execution, it will be in one of the nodes. After repair, it must redo whatever it had accomplished in that node only, and continue from there. In other words we need to apply (7) and (8) to each node separately, getting $\mathrm{E}[X_u(\beta)]$ and $\mathrm{E}[X_u^2(\beta)]$ for $u \in \{\mathrm{oe}, \mathrm{oc}, \mathrm{cc}, \mathrm{ce}\}$. These can then be used to evaluate $\mathrm{E}[X_{\mathrm{c}}^{\ell}(\beta)]$ ($\ell = 1, 2$). The formulae for computing these are given in Theorem 9.3.1 of [10, p. 518]. First define the four matrices (note that all objects with the 'hat' symbol come from (9), and are four dimensional)

$$[\hat{\boldsymbol{T}}_{\mathrm{c}}]_{uu} := \mathrm{E}[X_u(\beta)], \quad \hat{\boldsymbol{V}}_{\mathrm{c}} := [\hat{\boldsymbol{I}} - \hat{\boldsymbol{P}}_{\mathrm{c}}]^{-1}\hat{\boldsymbol{T}}_{\mathrm{c}}, \quad \text{and} \quad [\hat{\boldsymbol{\Gamma}}]_{uu} := C_u^2 - 1,$$

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

where $C_u^2 = \sigma_u^2(\beta)/(\mathrm{E}[X_u(\beta)])^2$ is the squared coefficient of variation of $X_u(\beta)$. Then

$$\mathrm{E}[X_{\mathrm{c}}(\beta)] = \hat{\boldsymbol{p}}_{\mathrm{c}}\hat{\boldsymbol{V}}_{\mathrm{c}}\hat{\boldsymbol{\epsilon}}'. \tag{11}$$

The generator of the distribution that would have occurred if all the node distributions had been exponential is given by $\hat{\boldsymbol{B}}_{\mathrm{c}} = \hat{\boldsymbol{V}}_{\mathrm{c}}^{-1}$, so (11) states that the mean time is independent of node distributions even if they are not ME. The variance can be evaluated from [10, p. 518] as

$$\sigma_{\mathrm{c}}^2(\beta) = \sigma_{\mathrm{exp}}^2 + \hat{\boldsymbol{p}}_{\mathrm{c}}\hat{\boldsymbol{V}}_{\mathrm{c}}\hat{\boldsymbol{T}}_{\mathrm{c}}\hat{\boldsymbol{\Gamma}}\hat{\boldsymbol{\epsilon}}', \tag{12}$$

where

$$\sigma_{\mathrm{exp}}^2 = 2(\hat{\boldsymbol{p}}_{\mathrm{c}}\hat{\boldsymbol{V}}_{\mathrm{c}}^2\hat{\boldsymbol{\epsilon}}') - (\hat{\boldsymbol{p}}_{\mathrm{c}}\hat{\boldsymbol{V}}_{\mathrm{c}}\hat{\boldsymbol{\epsilon}}')^2$$

is the variance of the similar exponential network.

This procedure can be extended to include two or more checkpoints, but the number of nodes grows quadratically. That is,

$$\mathrm{Dim}[\hat{\boldsymbol{P}}_{\mathrm{c}}] = (c+1)^2,$$

where c is the number of checkpoint nodes, and $\hat{\boldsymbol{P}}_{\mathrm{c}}$ is the extension of (9).

These formulae are necessary for exploring the major design issue of when or whether to use checkpointing at all. This is discussed in Section 3.6.

3.5. Asymptotic behavior

In Section 3.2 we showed that it is possible for T, T_a, and T_b to have completely different asymptotic behaviors. It is even more likely that the four T_us and T_{c} will have different minimum λs. In Section 3.2 it was assumed that every node could be reached by $\boldsymbol{p}_{\mathrm{o}}$. In checkpointing, by judiciously choosing m, some nodes will be 'downstream', and thereby not be reachable from $\boldsymbol{p}_{\mathrm{o}}$ without passing through m. Furthermore, nodes that are strictly 'upstream' from m will not be reachable from $\boldsymbol{p}_{\mathrm{c}}$. We give an example of this in Section 4.

For asymptotic behavior under RESTART with checkpointing, we need only the smallest λ among the four us, call it λ_{sc}. Note that this is almost surely greater than λ_{s}, and will only be equal if $p_{\mathrm{cc}} = 0$. To see this, consider $\hat{\boldsymbol{P}}_{\mathrm{c}}$ from (9). Fortunately, only node cc is relevant, because the other three nodes are visited at most once. But cc is visited a geometric number of times. It is known (see, e.g. [10, pp. 166–167]) that the sum of N independent and identically distributed (i.i.d.) ME variables, where N is geometrically distributed (here, with parameter p_{cc}), is ME, but λ_{s} for the sum will be less than the λ_{s} for the original function. We leave the full discussion of this for another time. What is important for our construction is that the same is not true for a geometric sum of i.i.d. PT variables. It is known (see, e.g. Lemma 2.2 of [2, p. 302]) that the geometric sum of i.i.d. regularly varying PT variables with power α retains the same α in the sense of (1). Asmussen provided a proof for PT variables (not necessarily regularly varying) which is presented in Appendix A. Thus, the asymptotic behavior of $X_{\mathrm{c}}(\beta)$ is dominated by $\alpha_{\mathrm{c}}(\beta) = \lambda_{\mathrm{sc}}/\beta$ in that $\mathrm{E}[X_{\mathrm{c}}(\beta)] = \infty$ for $\beta \geq \lambda_{\mathrm{sc}}$, and $\mathrm{E}[X_{\mathrm{c}}^2(\beta)] = \infty$ for $\beta \geq \lambda_{\mathrm{sc}}/2$.

It turns out that if $q_m = 0$, one of the eigenvalues is μ_{c}. Furthermore, the other eigenvalues do not depend on μ_{c}. So if $t_{\mathrm{c}} = 1/\mu_{\mathrm{c}}$ is large enough $\lambda_{\mathrm{sc}} = \mu_{\mathrm{c}}$. This can be very useful for estimating how large β can be before a system becomes unstable, or even if checkpointing should be used at all.

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

3.6. The cost of checkpointing versus failure rate

The designer of an MMS creates a flowchart, or general plan, which abstracts to the pair $\boldsymbol{p}$ and $\boldsymbol{P}$. He/she then makes a good-faith effort to estimate the mean time spent at each node for each visit, $1/\mu_i$. In effect, he/she knows from (5) what $\mathrm{E}[T]$ is, and if he/she assumes exponentially distributed service times for the nodes, then he/she can get the distribution. (The model can be extended to include nodes with ME distributions.) But if the system is not error free, he/she has no control over the value of β. From (7), he/she can estimate the total time for execution, $\mathrm{E}[X(\beta)]$, under RESTART. If reasonable values for β make it too large, then he/she must plant checkpoints in the software to reduce the time. At this point, he/she has more decisions to make, e.g. where to place the checkpoints and how much effort should he/she put in to make them efficient (thereby controlling the magnitude of t_c). Obviously, if t_c is negligible then he/she can put checkpoints everywhere, and thus effectively turn the process into RESUME (only the cost of repair remains). On the other hand, if β is very small then there is no need for checkpointing at all.

In real life, β changes over time (but presumably slowly enough so that the software will run many times for a constant β). For any $t_c > 0$, $\mathrm{E}[X(\beta)]$ will be smaller than $\mathrm{E}[X_c(\beta)]$ for small enough β. But as β grows, the two expectations will cross, say at β_{cross}. Thus, the checkpointing procedures can be shut off if it is believed that the β of the day will be less than β_{cross}. If $\mathrm{E}[X_c(\beta)]$ is larger than acceptable, or it is expected that β will approach $\lambda_{sc}/2$, then the designer must return to the 'drawing board' to find a better checkpoint scheme. The formulae described above are ideally suited to this exploration. A detailed example of such studies is presented in the next section.

4. An example

The software model we chose for our example comes from a 1980 paper by Cheung [4]. In that article the nodes themselves have a probability of erring, but here we consider the environment as a whole as failure prone. We make slight modifications to his *program control graph* that abstracts to the eight-dimensional matrix

$$\boldsymbol{P} = \begin{bmatrix} 0 & 0.70 & 0 & 0.30 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.00 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.75 & 0.25 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.40 & 0.60 & 0 \\ 0 & 0.30 & 0 & 0 & 0 & 0.30 & 0.10 & 0.30 \\ 0.80 & 0 & 0 & 0 & 0 & 0 & 0 & 0.20 \\ 0 & 0 & 0.75 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.10 & 0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{q}' = \begin{bmatrix} 0.00 \\ 0.00 \\ 0.00 \\ 0.00 \\ 0.00 \\ 0.00 \\ 0.25 \\ 0.90 \end{bmatrix}$$

($\boldsymbol{q}' = (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{\varepsilon}'$) with entrance vector $\boldsymbol{p} = [0.60, 0.20, 0.20, 0, 0, 0, 0, 0]$. Our choice for $\boldsymbol{M} = \mathrm{diag}[1.20, 2.30, 3.40, 0.80, 2.00, 2.40, 6.50, 1.40]$. We used these matrices to compute $\bar{F}(t)$ from (4); see Figure 1. The logarithmic scale was used to show the exponential tail as a straight line, whose slope is -0.1235, which is exactly the value of $-\lambda_s$ from (6). We have also computed $\mathrm{E}[T] = 9.10$ and $\sigma_T^2 = 66.30$ from (5). Thus, $C_T^2 = 66.30/(9.10)^2 = 0.8006$.

Following the instructions of Section 3.3 we then inserted a checkpoint following node $m = 4$. Noting that $q_4 = 1 - 0.75 - 0.25 = 0.0$, the resulting $((M + 1) = 9)$-dimensional

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

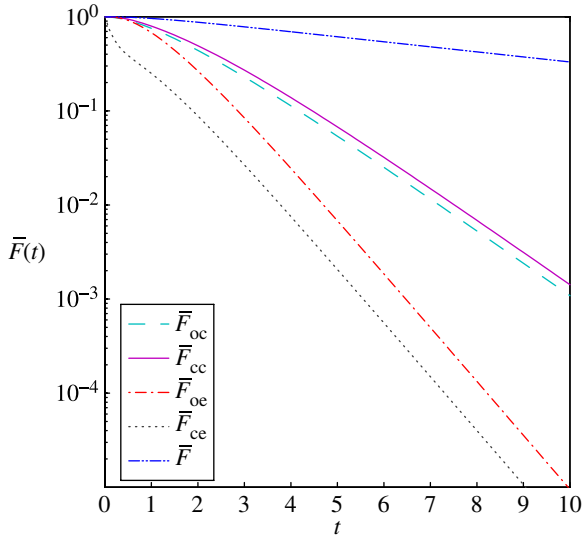

FIGURE 1: Plots of the five $\bar{F}(t)$ functions for T_c, T_{oe}, T_{oc}, T_{ce}, and T_{cc}. All have the same generator matrix $\boldsymbol{B}_c$ (with $t_c = 0$), but are very different in appearance. A semi-log plot was used to clearly show the difference among the five distributions, in particular the exponential tails which show up as straight lines. Note that because $t_c = 0$, T_c and T have the same distribution.

transition matrix is

$$\boldsymbol{P}_c = \begin{bmatrix} 0 & 0.70 & 0 & 0.30 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.00 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.75 & 0.25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0.30 & 0 & 0 & 0 & 0.30 & 0.10 & 0.30 & 0 \\ 0.80 & 0 & 0 & 0 & 0 & 0 & 0 & 0.20 & 0 \\ 0 & 0 & 0.75 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

with entrance and restart vectors

$$\boldsymbol{p}_o = [0.60, 0.20, 0.20, 0, 0, 0, 0, 0, 0] \quad \text{and} \quad \boldsymbol{p}_c = [0, 0, 0, 0, 0, 0.40, 0.60, 0, 0].$$

The exit probability vectors are

$$\boldsymbol{q}'_e = \begin{bmatrix} \boldsymbol{q}' \\ 0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{q}'_c = \begin{bmatrix} \boldsymbol{0}' \\ 1 \end{bmatrix}.$$

After numerically evaluating $\boldsymbol{Z}_c = (\boldsymbol{I} - \boldsymbol{P}_c)^{-1}$, we obtained the column vectors

$$\boldsymbol{\varepsilon}'_e = \boldsymbol{Z}_c \boldsymbol{q}'_e = [0.000, 0.000, 0.092, 0.000, 0.367, 0.186, 0.319, 0.932, 0.000]^\top,$$

$$\boldsymbol{\varepsilon}'_c = \boldsymbol{Z}_c \boldsymbol{q}'_c = [1.000, 1.000, 0.908, 1.000, 0.633, 0.814, 0.681, 0.068, 1.000]^\top.$$

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

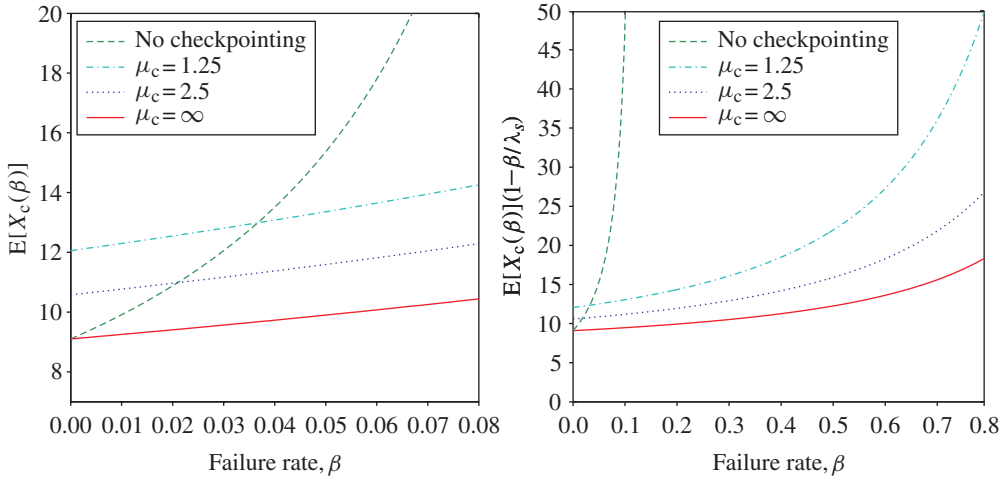

FIGURE 2: (a) Plots of $\mathrm{E}[X(\beta)]$ and $\mathrm{E}[X_c(\beta)]$ as a function of β for three different values of t_c. (b) Same functions as in (a), but now multiplied by $(1 - \beta/\lambda_u)$ $(u \in \{s, sc\})$. Plot (a) shows that, for $t_c > 0$, the *no checkpointing* curve lies below those with checkpointing for small β. But, as β increases, the curves cross, and, finally, $\mathrm{E}[X(\beta)]$ goes off to ∞, as shown in plot (b). Thus, the resulting functions are finite throughout the range up to $\beta = \lambda_u = 0.1235$ or 0.8000, and are meaningless above that point.

Note that $\boldsymbol{\varepsilon}'_e + \boldsymbol{\varepsilon}'_c = \boldsymbol{\varepsilon}'$. From this come the branching probabilities $p_{oe} = \boldsymbol{p}_o\boldsymbol{\varepsilon}'_e = 0.0184$ and $p_{oc} = \boldsymbol{p}_o\boldsymbol{\varepsilon}'_c = 0.9816$. Clearly, $p_{oe} + p_{oc} = 1$. Similarly, we computed $p_{ce} = \boldsymbol{p}_c\boldsymbol{\varepsilon}'_e = 0.2659$ and $p_{cc} = \boldsymbol{p}_c\boldsymbol{\varepsilon}'_c = 0.7341$. As before, $p_{ce} + p_{cc} = 1$. From (10) we calculated the expected number of times the checkpoint will be executed, namely,

$$\mathrm{E}[N_c] = \frac{p_{oc}}{p_{ce}} = \frac{0.9816}{0.2659} = 3.6916.$$

We then computed the four conditional complementary distribution functions, $\bar{F}_u(t)$, $u \in$ [oe, oc, ce, cc], and plotted them in Figure 1 (again for $t_c = 0$). We see two different slopes, $\lambda_{oe} = \lambda_{oc} = 1.3222$ and $\lambda_{ce} = \lambda_{cc} = 0.8000$, both different from $\lambda_s = 0.1235$. Note that $\bar{F}_{oe}$ has a nonzero slope at $t = 0$ [$f_{oe}(0) > 0$]. This can only occur if there is a path from entry to exit that passes through only one node. That is the case here, namely c $\to$ 7 $\to$ e. All other paths pass through at least two nodes. In fact, all paths to c, by construction, must pass through both 4 and 9 (in general, $\cdots \to m \to M + 1 \to$ c).

Using the representations for $\bar{F}_u(t)$ in Section 3.3, $\mathrm{E}[X(\beta)]$, $\mathrm{E}[X(\beta)^2]$, $\mathrm{E}[X_u(\beta)]$, and $\mathrm{E}[X_u(\beta)^2]$ $(u \in \{oe, oc, ce, cc\})$ were evaluated from (7) and (8). We then computed $\mathrm{E}[X_c(\beta)]$ and $\mathrm{E}[X_c(\beta)^2]$ from (11) and (12). This was done for three values of t_c, with the values for $\mathrm{E}[X(\beta)]$ and $\mathrm{E}[X_c(\beta)]$ plotted in Figure 2(a) as a function of β. As expected,

$$\mathrm{E}[X(\beta = 0)] = \mathrm{E}[X_c(\beta = 0; t_c = 0)] < \mathrm{E}[X_c(\beta = 0; t_c > 0)].$$

But as β is increased, $\mathrm{E}[X(\beta)]$ crosses the other curves and goes to ∞ at $\beta = \lambda_s = 0.1235$. From such graphs, it is not clear just where the blowups occur. But, from (11), and the spectral decomposition theorem, it is easy to show that

$$Y := \lim_{\beta \to \lambda_s} \left(1 - \frac{\beta}{\lambda_s}\right) \mathrm{E}[X(\beta)] = \frac{1}{\lambda_s}(\boldsymbol{p}_o\boldsymbol{v}'_s)(\boldsymbol{u}_s\boldsymbol{\varepsilon}'),$$

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press

where $\boldsymbol{u}_s$ and $\boldsymbol{v}'_s$ are the eigenvectors of $\boldsymbol{B}$ for eigenvalue λ_s. It is clear that $(1-\beta/\lambda_{sc})\,\mathrm{E}[X_c(\beta)]$ must also approach a constant as β approaches λ_{sc}, so in Figure 2(b) we present the same four functions, now scaled by their *blowup* factors, $(1-\beta/\lambda_u)$, $u = s, sc$. Clearly, they are finite at $\beta = 0.1235$ and $\beta = 0.800$, respectively. Above those points the curves are meaningless. This concurs with our argument that $\bar{H}_c(x)$ behaves like a PT function with $\alpha = \beta/\lambda_{sc}$.

Note that if $\mu_c < 0.8000$, it dominates the tail and the blowup occurs at $\beta = \mu_c$.

5. Conclusion

We have shown that if a procedure that can be described as a Markov chain is subjected to random (exponentially distributed) failures, the formulae previously derived concerning the RESTART problem can be applied to the Markov chain to yield $\mathrm{E}[X(\beta)]$, $\mathrm{E}[X(\beta)^2]$, and its asymptotic behavior as a PT with parameter α. Furthermore, if checkpointing is shown to be needed, we have given a method for embedding the checkpoints and derived formulae for getting $\mathrm{E}[X_c(\beta)]$, $\mathrm{E}[X_c(\beta)^2]$, and $\alpha_c(\beta)$. We have presented a nontrivial example to show how everything fits together.

Appendix A. Compound sums of power tails

Proposition 1. *Let $X_1, X_2, \ldots$ be i.i.d. PT with index α, and let $N \geq 0$ be an independent integer-valued RV, with $S_N = X_1 + X_2 + \cdots + X_N$. Then S_N is also PT with index α provided that $\mathrm{E}[N] < \infty$ for $0 < \alpha \leq 2$ and $\mathrm{E}[N^{\alpha/2}] < \infty$ for $\alpha \geq 2$.*

Proof. That $\mathrm{E}[S_N^\gamma] < \infty$ for $\gamma < \alpha$ follows immediately from Theorem 5.1 of [7, p. 20]. Now look at $\gamma > \alpha$. Choose $n \geq 1$ with $\Pr[N \geq n] > 0$. Then

$$\begin{aligned}
\mathrm{E}[S_N^\gamma] &\geq \Pr[N \geq n]\,\mathrm{E}[\mathrm{E}[S_N^\gamma \mid N \geq n]] \\
&\geq \Pr[N \geq n]\,\mathrm{E}[\mathrm{E}[S_n^\gamma \mid N \geq n]] \\
&= \Pr[N \geq n]\,\mathrm{E}[S_n^\gamma] \\
&\geq \Pr[N \geq n]\,\mathrm{E}[X_1^\gamma] \\
&= \infty,
\end{aligned}$$

completing the proof.

Acknowledgement

We would like to thank Søren Asmussen for contributing Appendix A, and in general for his driving interest in the RESTART problem.

References

[1] Asmussen, S. (2003). *Applied Probability and Queues*, 2nd edn. Springer, New York.
[2] Asmussen, S. and Albrecher, H. (2010). *Ruin Probabilities*, 2nd edn. World Scientific, River Edge, NJ.
[3] Asmussen, S. *et al.* (2008). Asymptotic behavior of total times for jobs that must start over if failure occurs. *Math. Operat. Res.* **33,** 932–944.
[4] Cheung, R. C. (1980). A user-oriented software reliability model. *IEEE Trans. Software Eng.* **6,** 118–125.
[5] Chimento, P. F., Jr. and Trivedi, K. S. (1993). The completion time of programs on processors subject to failure and repair. *IEEE Trans. Comput.* **42,** 1184–1194.
[6] Gokhale, S. S. and Trivedi, K. S. (2006). Analytical models for architecture-based software reliability analysis: a unification framework. *IEEE Trans. Reliab.* **55,** 281–292.
[7] Gut, A. (1988). *Stopped Random Walks*. Springer, New York.
[8] Jelenković, P. R. and Tan, J. (2007). Can retransmissions of superexponential documents cause subexponential delays? In *Proc. INFOCOM '07* (Anchorage, 2007), pp. 892–900.

[9] KULKARNI, V., NICOLA, V. AND TRIVEDI, K. (1986). On modeling the performance and reliability of multimode systems. *J. Systems Software* **6,** 175–183.
[10] LIPSKY, L. (2009). *Queueing Theory: A Linear Algebraic Approach*, 2nd edn. Springer, New York.
[11] LITTLEWOOD, B. (1975). A reliability model for Markov structured software. In *Proc. Internat. Conf. Reliab. Software* (Los Angeles, 1975), Association for Computing Machinery, New York, pp. 204–207.
[12] NEUTS, M. F. (1981). *Matrix-Geometric Solutions in Stochastic Models*. Johns Hopkins University Press, Baltimore, MD.
[13] SHEAHAN, R., LIPSKY, L., FIORINI, P. AND ASMUSSEN, S. (2006). On the completion time distribution for tasks that must restart from the beginning if a failure occurs. In *ACM SIGMETRICS Performance Evaluation Review*, Association for Computing Machinery, New York, pp. 24–26. (Expanded version: (2007), Tech. Rep.)
[14] SHOLL, H. A. AND BOOTH, T. L. (1975). Software performance modeling using computation structures. *IEEE Trans. Software Eng.* **1**, 414–420.
[15] VAN DE LIEFVOORT, A. H. A. (1982). An algebraic approach to the steady-state solution of G/G/1//N-type loops. Doctoral Thesis, University of Nebraska, Lincoln.

LESTER LIPSKY, *University of Connecticut*

Department of Computer Science and Engineering, University of Connecticut, 371 Fairfield Road, Storrs, CT 06269-2155, USA. Email address: lester@engr.uconn.edu

DEREK DORAN, *University of Connecticut*

Department of Computer Science and Engineering, University of Connecticut, 371 Fairfield Road, Storrs, CT 06269-2155, USA.

SWAPNA GOKHALE, *University of Connecticut*

Department of Computer Science and Engineering, University of Connecticut, 371 Fairfield Road, Storrs, CT 06269-2155, USA.

https://doi.org/10.1239/jap/1318940465 Published online by Cambridge University Press