Journal of Software Engineering and Applications, **2013, 6, 41-53**
http://dx.doi.org/10.4236/jsea.2013.64A006 Published Online April 2013 (http://www.scirp.org/journal/jsea)

Aperiodic Checkpoint Placement Algorithms—Survey and Comparison*

Shunsuke Hiroyama, Tadashi Dohi, Hiroyuki Okamura

Department of Information Engineering, Hiroshima University, Hiroshima, Japan.
Email: dohi@rel.hiroshima-u.ac.jp, okamu@rel.hiroshima-u.ac.jp

Received January 23rd, 2013; revised February 24th, 2013; accepted March 4th, 2013

ABSTRACT

In this article we summarize some aperiodic checkpoint placement algorithms for a software system over infinite and finite operation time horizons, and compare them in terms of computational accuracy. The underlying problem is formulated as the maximization of steady-state system availability and is to determine the optimal aperiodic checkpoint sequence. We present two exact computation algorithms in both forward and backward manners and two approximate ones; constant hazard approximation and fluid approximation, toward this end. In numerical examples with Weibull system failure time distribution, it is shown that the combined algorithm with the fluid approximation can calculate effectively the exact solutions on the optimal aperiodic checkpoint sequence.

Keywords: Checkpoint Placement; Aperiodic Policy; Availability Models; Computation Algorithms; Comparison

1. Introduction

It is well known that the system failure in large-scale computer systems can lead to a huge economic or critical social loss. Checkpointing and rollback recovery is a commonly used technique for improving the reliability/availability of fault-tolerant computing systems, and is regarded as a low-cost software dependability technique from the standpoint of environment diversity. Especially, when file systems to write and/or read data are designed, checkpoint (CP) generations back up periodically/aperiodically the significant data on a primary medium to safe secondary media, and play a significant role to limit the amount of data processing for recovery actions after system failures occur. If CPs are frequently taken, a larger overhead will be incurred. Conversely, if only a few CPs are taken, a larger overhead after a system failure will be required in rollback recovery actions. Hence, it is important to determine the optimal CP sequence taking account of the trade-off between two kinds of overhead factor above. In many cases, the system failure phenomenon is described with a probability distribution called the *system-failure time distribution*, and the optimal CP sequence is determined based on any stochastic model. For the excellent survey on this topic, see [2,3].

First Young [4] obtains the optimal CP interval approximately for a computation restart after system failures. Baccelli [5], Chandy *et al.* [6,7], Dohi *et al.* [8-10], Gelenbe and Derochette [11], Gelenbe [12], Gelenbe and Hernandez [13], Goes and Sumita [14], Goes [15], Grassi *et al.* [16], Kobayashi and Dohi [17], Kulkarni *et al.* [18], Nicola and van Spanje [19], Sumita *et al.* [20], among others, propose performance evaluation models for database recovery, and calculate the optimal CP intervals which maximize the system availability or minimize the mean overhead during the normal operation. L'Ecuyer and Malenfant [21] formulate a dynamic CP placement problem by a Markov decision process. Ziv and Bruck [22] analyze an online algorithm for a probabilistic CP placement. Vaidya [23] examines an impact of checkpoint latency on overhead ratio for a simple CP model. Okamura *et al.* [24] reformulate the Vaidya model [23] with a semi-Markov decision process and further develop a reinforcement adaptive learning algorithm for CP placement. For several CP models in the literature, the periodic CP intervals are implicitly assumed. This is

*This is an extended version of the conference paper [1] presented at The 2nd International Conference on Advanced Computer Science and Information Technology (AST 2010), Miyazaki, Japan, June 23-25, 2010.

This work is supported by the Grant-in-Aid for the Scientific Research (C) from the Ministry of Education, Science, Sports and Culture of Japan, under Grant Nos. 23510171 (2011-2013) and 23500047 (2011-2013).

because the periodic CP intervals maximize the steady-state system availability, and in many cases, are better than the randomized CP ones which are given by independent and identically distributed random variables. However, it is worth noting that the periodic CP strategies can not be always validated in some cases and less performe than the aperiodic CP placement. In general, it is known that the way to place the optimal CP sequence strongly depends on both kind of objective functions (system availability, mean overhead, etc.) and kind of system-failure time distribution. Since the aperiodic CP involves the periodic CP as a special case, it is meaningful to consider the aperiodic CP placement algorithm for file systems.

When the system-failure time obeys a non-exponential distribution, it is easily shown that the aperiodic CP placement is not worse than the periodic CP one. Toueg and Babaoğlu [25] develop a dynamic programming (DP) algorithm which minimizes expected execution time of tasks placing CPs between two consecutive tasks under very general assumptions. Kaio and Osaki [26] consider an approximate aperiodic CP placement algorithm under the asssumption that the conditional system-failure probability is constant during the successive CPs. Fukumoto *et al.* [27,28] and Ling *et al.* [29] propose fluid approximation methods based on a variational calculus approach to derive the cost-optimal aperiodic CP sequence. Ozaki *et al.* [30,31] give an exact aperiodic CP placement algorithm and further develop an estimation scheme under the incomplete knowledge on system-failure time distribution. In a fashion similar to the above approach, Dohi *et al.* [32] formulate another aperiodic CP placement problem with equality constraints. Iwamoto *et al.* [33], Okamura *et al.* [34,35], and Okamura and Dohi [36] propose different DP-based algorithms from Toueg and Babaoğlu [25] under the availability criterion, by taking account of another dependability technique, called the *software rejuvenation* in the presense of *software aging*, where the system failure caused by the aging is not exponentially distributed. Recently, Ozaki *et al.* [37] propose a fixed-point type algorithm for an aperiodic CP placement with an infinite operation-time horizon. In this way, considerable attentions have been paid for aperiodic CP placement problems in past.

Nevertheless, it can be pointed out that no effective aperiodic CP placement algorithm has been known yet when the number of CPs is very large. The constant hazard approximation [26] and fluid approximation [27-29] may poorly work in such a case. The search-based iteration algorithm in [30,31] and the DP-based algorithm in [33-36], which are regarded as exact computation algorithms, also require the very careful adjustment to determine the number of CPs if the operation time for a file system is finite. As the operation time becomes longer, in general, the number of CPs is sensitive to not only the determination of the aperiodic CP sequence but also the resulting dependability measures. In this article we summarize some aperiodic CP placement algorithms for a software system over infinite and finite operationtime horizons, and compare them in terms of computational accuracy. It is proposed to combine the fluid approximation with an exact computation algorithm in determining the initial value of the number of CPs. The idea is quite simple, but we show that the combined algorithm with the fluid approximation can calculate effectively the exact solutions on the optimal aperiodic CP sequence.

2. Formulation of Optimal CP Placement

First, consider a centralized file system with sequential checkpoint (CP) over an infinite time horizon. The system operation starts at time $t=0$, and the CP is sequentially placed at time $\{t_1, t_2, \cdots, t_k, \cdots\}$ to back up the data processed in the file system. At each CP, $t_k\ (k=1,2,\cdots)$, all the file data on the main memory is saved to a safe secondary medium, where the fixed cost (time overhead) $c_0\ (>0)$ is needed per each CP placement. It is assumed that the system operation stops during the checkpointing, so during the period c_0 the file system does not deteriorate. System failure may occur according to an absolutely continuous and non-decreasing probability distribution function $F(t)$ having density function $f(t)$ and finite mean $\mu\ (>0)$. Upon a system failure, a rollback recovery takes place immediately where the file data saved at the last CP creation is used. Next, a CP restart is performed and the file data is recovered to the state just before the system failure occurs. The time length required for the CP restart is given by the function $L(\cdot)$, which depends on the system failure time, and is assumed to be differentiable and increasing. We call the function $L(\cdot)$ *the recovery function* in this article. After the completion of CP restart, an additional CP must be created to save the current state and the system operation restarts with the same condition as the initial point of time $t=0$. The similar cycle repeats again and again over an infinite time horizon.

The problem is to determine the optimal CP sequence $\mathbf{t}_\infty = \{t_1, t_2, t_3, \cdots\}$ maximizing the steady-state system availability:

$$AV_\infty(\mathbf{t}_\infty) = \frac{\int_0^\infty \overline{F}(t)\mathrm{d}t}{V_\infty(t_\infty) + \int_0^\infty \overline{F}(t)\mathrm{d}t} = \frac{\mu}{V_\infty(t_\infty) + \mu} \quad (1)$$

where

$$\overline{F}(\cdot) = 1 - F(\cdot)$$

and

$$V_\infty(\mathbf{t}_\infty) = \sum_{k=0}^{\infty} \int_{t_k}^{t_{k+1}} \left[c_0(k+1) + L(t - t_k) \right] \mathrm{d}F(t) \quad (2)$$

denotes the expected operaing cost with $t_0 = 0$. It is evident that the underlying problem is reduced to a simple minimization problem $\min_{\mathbf{t}_\infty} V_\infty(\mathbf{t}_\infty)$. In this problem, the expected recovery cost is usually given by the affine form

$$L(t-t_k) = a_0(t-t_k) + b_0 \quad (t > t_k, k = 0,1,2,\cdots)$$

for the system failure time t, where $a_0(>0)$ and $b_0(>0)$ are given constants. Instead, by replacing the above CP cost and recovery cost by $c_0 k$ and $L(t_{k+1}-t) = a_0(t_{k+1}-t)$, this is equivallent to the classical inspection problem by Barlow and Proschan [38]. **Figure 1** illustrates the configuration of the underlying CP placement with a finite operation-time horizon $T(>0)$.

From the analogy to the inspection problem, it can be easily shown that the optimal CP sequence $\mathbf{t}_\infty^* = \{t_1^*, t_2^*, t_3^*, \cdots\}$ maximizing the steady-state system availability is a non-increasing sequence under the assumption that the system failure time distribution $F(t)$ is PF_2 (Polya Frequency Function of Order 2) [38], if there exists the optimal CP sequence t_∞^* satisfying $t_1 \geq t_2 - t_1 \geq t_3 - t_2 \geq \cdots$. Then, it must satisfy the following first order condition of optimality:

$$t_k^* - t_{k-1}^* = \frac{F(t_{k+1}^*) - F(t_k^*)}{f(t_k^*)} + \frac{c_0}{a_0}. \tag{3}$$

From the condition of optimality, an algorithm to derive the optimal CP sequence $\mathbf{t}_\infty^* = \{t_1^*, t_2^*, t_3^*, \cdots\}$ which minimezes $V_\infty(\mathbf{t}_\infty)$ or equivalentlly maximizes $AV_\infty(\mathbf{t}_\infty)$ can be derived as follows.

Forward CP Placement Algorithm for an Infinite Operation-Time Horizon: [30,31,37].

Step 1: Set t_1 satisfying $c_0 = a_0 \int_0^{t_1} t \mathrm{d}F(t) + b_0 F(t_1)$.

Step 2: Compute $\mathbf{t}_\infty = \{t_2, t_3, t_4, \cdots\}$ using Equation (3).

Step 3: For k-th CP $(k = 1,2,3,\cdots)$, if $t_{k+1} - t_k > t_k - t_{k-1}$, then decrease t_1 and **Go** to **Step 2**.

Step 4: For k-th CP $(k = 1,2,3,\cdots)$, if $t_{k+1} - t_k < 0$, then increase t_1 and **Go** to **Step 2**.

Step 5: For the resulting CP sequence $t_1 < t_2 < \cdots < t_k$, if $\int_{t_k}^{t_{k+1}} \left[c_0(k+1) + L(t-t_k)\right] \mathrm{d}F(t) \approx \epsilon$, then **Stop** the procedure, where $\epsilon(>0)$ is sufficiently small tolerance value and $t_{k+1} - t_k \approx 0$.

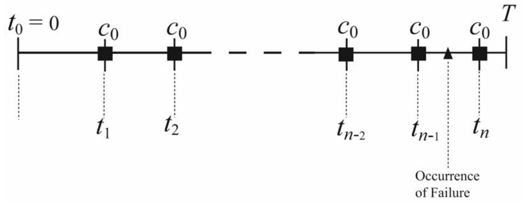

Figure 1. Configuration of the aperiodic CP placement with a finite operation-time horizon *T*.

In the above algorithm, arbitrary increasing and decreasing operations in **Steps 3** and **4** can be taken to speed up the computation. The simplest method would be the bisection serach method. As the simplest case, if the system failure time is given by the exponential distribution with mean μ, it is well known that the optimal CP sequence is periodic, *i.e.*,

$$t_1 = t_2 - t_1 = t_3 - t_2 = \cdots.$$

Since the processing time for a given transaction is in general bounded, the CP placement for an infinite-time horizon may be questionable in many practical applications. As a natural extension of the infinite-time horizon problem, it would be interesting to consider the finite operation-time horizon problem, because $T \to \infty$ is a special case. Suppose that the time horizon of operation for the file system is finite, say, $T(>0)$, which can be regarded as a fixed transaction processing time. For a finite sequence $\mathbf{t}_N = \{t_1, t_2, \cdots, t_N\}$, the expected operating cost is given by

$$\begin{aligned}
&V_T(\boldsymbol{t}_N) \\
&= \sum_{k=0}^{N} \int_{t_k}^{t_{k+1}} \left[c_0(k+1) + L(t-t_k)\right] \mathrm{d}F(t) \\
&\quad + \int_{t_{N+1}}^{\infty} c_0(N+1) \mathrm{d}F(t) \\
&= \sum_{k=0}^{N} c_0(k+1)\left[F(t_{k+1}) - F(t_k)\right] \\
&\quad + \sum_{k=0}^{N} \int_{t_k}^{t_{k+1}} \left[a_0(t-t_k) + b_0\right] \mathrm{d}F(t) \\
&\quad + c_0(N+1)\bar{F}(t),
\end{aligned} \tag{4}$$

where $N = \min\{k : t_{k+1} > T\}$. Also we suppose that the file system restarts with a fixed CP overhead c_0 just after the time T, if the system failure does not occur. Since the steady-state system availability is given by

$$AV_T(\mathbf{t}_N) = \frac{\int_0^T \bar{F}(t) \mathrm{d}t}{V_T(\mathbf{t}_N) + \int_0^T \bar{F}(t) \mathrm{d}t}, \tag{5}$$

the underlying maximization problem reduces to $\min_{\mathbf{t}_N} V_T(\mathbf{t}_N)$. It should be noted that the recovery cost does not occur at $t > T$. To simplify the notation, we define $t_{N+1} = T$ in this article. When the recocery cost function is the affine form *i.e.*, $L(t) = a_0 t + b_0$, differentiating Equation (4) with respect to t_k $(k = 1,2,\cdots,N)$ and setting it equal to 0 yield Equation (3) again for $t_k - t_{k-1} > 0$ $(k = 1,2,3,\cdots,N)$ and a given N.

Since the finite operation-time horizon problem involves the constraint N on the number of CPs, it is impossible to apply directly the forward CP placement algorithm for an infinite operation-time horizon problem. However, by adjusting the value of N, we can develop the similar algorithm to compute the optimal CP sequence. The basic idea is to utilize the non-increasing property of CP sequence under the PF_2 assumption for an arbitrary number N. Based on the result for an infinite time horizon [30,31,37], we modify the forward CP placement algorithm as follows.

Forward CP Placement Algorithm for a Finite Operation-Time Horizon: [30,31].

Step 1: Set the lower and upper bounds of t_1 by $z_l := 0$ and $z_u = T$, respectively.

Step 2: $t_1 := (z_l + zu)/2$.

Step 3: For $k = 0,1,\cdots,N$, compute the CP sequence $t_2, t_2, \cdots, t_N, t_{N+1}$ by

$$t_{k+1} := F^{-1}\left(F(t_k) + (t_k - t_{k-1}) f(t_k) - \frac{c_0}{a_0}\right).$$

Step 4: For $k = 1,2,\cdots,N$,

Step 4.1: If $t_{k+1} - t_k > t_k - t_{k-1}$, then $z_u = t_1$ and **Go** to **Step 2**.

Step 4.2: If $t_{k+1} - t_k < 0$, then $z_l = t_1$ and **Go** to **Step 2**.

Step 5: For an arbitrary tolerance level ϵ, if $t_{N+1} - T < -\epsilon$, then $z_u := t_1$ and **Go** to **Step 2**.

Step 6: For an arbitrary tolerance level ϵ, if $t_{N+1} - T > \epsilon$, then $z_l := t_1$ and **Go** to **Step 2**.

Step 7: End.

For all possible combinations of N, we calculate all expected operating costs using the above algorithm, and determine both the optimal number of CPs, N^* and its associated CP sequence $t_N^* = \{t_1^*, t_2^*, \cdots, t_N^*\}$. It should be noted that the above two algorithms can be validated only when the system failure time distribution is PF_2 and the resulting CP sequence is non-increasing, *i.e.*, $t_k \geq t_{k+1}$. The most significant point is that these algorithms are very fast to derive the optimal CP sequence, but strongly depend on the initial value t_1. In the worst case, it is evident that these algorithms are sometime unstable and that the resulting CP sequence may not converge to the optimal solution. To overcome this point, the careful selection of the initial value t_1 is essentially needed, so we improve it by the following algorithm.

Improved Forward CP Placement Algorithm for a Finite Operation-Time Horizon:

Step 1: Set $t_1 = 0$, $N = 1$, $\Delta t\,(\ll 1)$ and the upper bound of serach range N_{Max}.

Step 2: Set $j = 0$ and $V_0 = 1.0 \times 10^5$.

Step 2.1: $t_1 := t_1 + \Delta t$.

Step 2.2: For $k = 1,\cdots,N$, compute $t_{k+1}\,(i = 1,2,\cdots,N)$ satisfying

$$t_{k+1} := F^{-1}\left(F(t_k) + (t_k - t_{k-1}) f(t_k) - \frac{c_0}{a_0}\right).$$

Step 2.3: Compute the corresponding expected operating cost and set it as V_j based on $t_{k+1}\,(k = 1,2,\cdots,N)$.

Step 2.4: If $V_{j-1} < V_j$, then $V_N = V_{j-1}$ and $t_N = t_{j-1}$, and **Go** to **Step 3**, otherwise $j = j+1$ and **Go** to **Step 2.1**, where $j < 1 \times 10^4$.

Step 3: If $t_k < T\,(k = 1,2,\cdots,N)$ and $N \leq N_{\mathrm{Max}}$, then $N = N+1$ and **Go** to **Step 2**, otherwise **Go** to **Step**.

Step 4: For all $N = 1,2,\cdots,N_{\mathrm{Max}}$, search the minimum value C_{N^*} and its associated CP sequence t_{N^*}.

Since the initial value t_1 in the above algorithm can be adjusted gradually from 0, the stability for the original forward CP placement algorithm could be rather improved. However, when Δt is relatively large, the solution may still drop in the local minimum, and even the improved algorithm may fail to converge. In our numerical experiments, even when $\Delta t > 1 \times 10^{-2}$, the search of the initial value t_1 was sometimes unsuccsessful. In addition, it can be obvious that the computation cost of the improved algorithm is much larger than the original forward CP placement algorithm. In the following section, we introduce more stable algorithm on computation.

3. Backward CP Placement Algorithm

For the same aperiodic CP placement problem, Naruse *et al.* [39,40] propose to solve the optimality condition in the backward manner. Letting $V_T(\mathbf{t}_N) = V_T(\mathbf{t}_N, N)$ for a given N, the optimal CP sequence $\mathbf{t}_N^* = \{t_1^*, t_2^*, \cdots, t_N^*\}$ has to satisfy the first ortder condition $\partial V_T(\mathbf{t}_N^*)/\partial \mathbf{t}_N^* = 0$, and should be the solution of the following $(N-1)$ simultaneous equations:

$$\begin{aligned}
t_{N-1} - t_{N-2} &= \frac{F(t_N) - F(t_{N-1})}{f(t_{N-1})} + \frac{c_0}{a_0}, \\
&\vdots \\
t_k - t_{k-1} &= \frac{F(t_{k+1}) - F(t_k)}{f(t_k)} + \frac{c_0}{a_0}, \\
&\vdots \\
t_1 &= \frac{F(t_2) - F(t_1)}{f(t_1)} + \frac{c_0}{a_0}.
\end{aligned} \tag{6}$$

Although this algorithm does not depend on the PF_2 property, it is not feasible for a large number of CPs, because an explosion of the number of simultaneous equations occurs for increasing the number of CPs. In fact, the authors in [40] present only a toy problem with a very small number of CPs.

The most realistic backward algorithm is already given by Iwamoto *et al.* [33], and is based on the well-known dynamic programing (DP). Since this algorithm does not also depend on the PF_2 property, it is applicable even to the more general failure time distribution. During the time period between two successive CPs,

$$[t_{k-1}, t_k)\,(k = 1,2,\cdots,N,N+1),$$

the expected operation time $U(t_k \mid t_{k-1})$ and the mean time length of one cycle $S(t_k \mid t_{k-1})$ are given by

$$U(t_k \mid t_{k-1}) = \int_0^{t_k - t_{k-1}} x\mathrm{d}F(x \mid t_{k-1}) + (t_k - t_{k-1})\bar{F}(t_k - t_{k-1} \mid t_{k-1}), \tag{7}$$

$$S(t_k \mid t_{k-1}) = \int_0^{t_k - t_{k-1}} \{x + L(x) + c_0\} \mathrm{d}F(x \mid t_{k-1}) + (t_k - t_{k-1} + c_0)\bar{F}(t_k - t_{k-1} \mid t_{k-1}), \tag{8}$$

respectively, where one cycle is defined as the time interval between two successive renewal points. In Equations (7) and (8), $F(\cdot \mid \cdot)$ represents the conditional probability distribution:

$$F(s \mid t) = 1 - \bar{F}(t+s)/\bar{F}(t). \tag{9}$$

At the end of the operation-time $T = t_{N+1}$, the above expressions are rewritten as follows.

$$U(T \mid t_N) = \int_0^{T - t_N} x\mathrm{d}F(x \mid t_N) + (T - t_N)\bar{F}(T - t_N \mid t_N), \tag{10}$$

$$S(T \mid t_N) = \int_0^{T - t_N} \{x + L(x) + c_0\} \mathrm{d}F(x \mid t_N) + (T - t_N + c_0)\bar{F}(T - t_N \mid t_N). \tag{11}$$

From the principle of optimality, we obtain the following DP equations:

$$h_k = \max_{t_k} w\left(t_k \mid t_{k-1}^*, h_1, h_{k+1}\right), \quad k = 1, \cdots, N, \tag{12}$$

$$h_{N+1} = w\left(T \mid t_N^*, h_1, h_1\right), \tag{13}$$

where the function $w(t_k \mid t_{k-1}, s_0, s_1)$ is given by

$$w(t_k \mid t_{k-1}, s_0, s_1) = U(t_k \mid t_{k-1}) - \xi S(t_k \mid t_{k-1}) + s_0 F(t_k - t_{k-1} \mid t_{k-1}) + s_1 \bar{F}(t_k - t_{k-1} \mid t_{k-1}). \tag{14}$$

In the above equation, ξ indicates the maximum steady-state system availability and h_k, $k = 1, \cdots, N+1$, are relative value functions in the DP. The derivation of the optimal CP intervals is equivalent to finding $\mathbf{t}_N = \{t_1^*, \cdots, t_N^*\}$ which satisfy the DP equations. Following Iwamoto *et al.* [33], we apply the *policy iteration algorithm* which is effective to solve the above type of functional equations. Instead of the original function $w(\cdot)$, define for convenience the following function:

$$w\left(t_k \mid t_{k-1}, h_1, w\left(t_{k+1} \mid t_k, h_1, h_{k+2}\right)\right). \tag{15}$$

Then the DP-based CP placement algorithm is given in the following:

Backward CP Placement Algorithm: [33].

Step 1: Give initial values

$$i := 0, \tag{16}$$

$$t_0 := 0, \tag{17}$$

$$\mathbf{t}_N^{(0)} := \left\{t_1^{(0)}, \cdots, t_N^{(0)}\right\}, \tag{18}$$

where i is the iteration number.

Step 2: Compute $h_1^{(i)}, \cdots, h_{N+1}^{(i)}, \xi^{(i)}$ under the policy $\mathbf{t}_N^{(i)}$.

Step 3: Solve the following optimization problems:

$$t_k^{(i+1)} := \underset{t_{k-1}^{(i)} \le t \le t_{k+1}^{(i)}}{\arg\max}\, w\left(t \mid t_{k-1}^{(i)}, 0, w\left(t_{k+1}^{(i)} \mid t_k, 0, h_{k+2}^{(i)}\right)\right), \quad \text{for } k = 0, 1, \cdots, N-1, \tag{19}$$

$$t_N^{(i+1)} := \underset{t_{N-1}^{(i)} \le t \le T}{\arg\max}\, w\left(t \mid t_{N-1}^{(i)}, 0, w(T \mid t, 0, 0)\right). \tag{20}$$

Step 4: For all $k = 1, \cdots, N$, if $\left|t_k^{(i+1)} - t_k^{(i)}\right| < \delta$, stop the algorithm, where δ is an error tolerance, otherwise, let $i := i+1$ and go to **Step 2**.

In **Step 2** of the above algorithm, we have to calculate the relative value functions. From the original DP Equations (12) and (13), we find that the relative value functions under a fixed policy $\mathbf{t}_N = \{t_1, \cdots, t_N\}$ must satisfy the following linear equation:

$$\boldsymbol{Mx} = \boldsymbol{b}, \tag{21}$$

where

$$[\boldsymbol{M}]_{k,j} = \begin{cases} -\bar{F}(t_k - t_{k-1} \mid t_{k-1}) & \text{if } k = j \text{ and } j \neq N+1, \\ 1 & \text{if } k = j+1, \\ T(t_k \mid t_{k-1}) & \text{if } j = N+1, \\ 0 & \text{otherwise}, \end{cases} \tag{22}$$

$$\boldsymbol{x} = (h_2, \cdots, h_N, h_{N+1}, \xi)^{\mathrm{tr}}, \tag{23}$$

$$\boldsymbol{b} = \left(U(t_1 \mid t_0), \cdots, U(t_N \mid t_{N+1}), U(T \mid t_N)\right)^{\mathrm{tr}}, \tag{24}$$

$[\cdot]_{k,j}$ denotes the (k, j)-element of matrix, and tr represents transpose of vector. Without a loss of generality, we set $h_1 = 0$ in the above algorithm.

For both forward and backward CP placement algorithms, it is essential to determine the number of CPs, N, during the finite operation-time horizon. In other words, if the initial value of N in the algorithms can be known in advance, it can be easily explored with any low-cost search technique. In the following section, we introduce two approximate algorithms for the finite operation-time horizon problem.

4. Approximate CP Placement Algorithms

4.1. Constant Hazard Approximation

If the time interval between two successive CPs, $(t_k, t_{k+1}]$ $(k = 0, 1, 2, \cdots, N)$, is sufficiently short, the system-failure probability during the time interval can be approximately considered as a constant, *i.e.*,

$$\frac{F(t_{k+1})-F(t_k)}{\bar{F}(t_k)}=1-\alpha\in(0,1). \quad (25)$$

Kaio and Osaki [26] approximate the expected operating cost, $V_T(\mathbf{t}_N)$ as a function of α under the above assumption. Here we derive the same result as [26] in a different way. Let X be the system-failure time having the probability distribution $F(t)$. For an arbitrary probability $\alpha\in(0,1)$, define the CP sequence satisfying the following quantile condition:

$$t_1=\sup\{t>0;\Pr\{X>t\}\ge\alpha\}=\bar{F}^{-1}(\alpha),$$

$$t_2=\sup\{t>t_1;\Pr\{X>t\mid X>t_1\}\ge\alpha^2\}=\bar{F}^{-1}(\alpha^1),$$

$$\vdots$$

$$t_k=\sup\{t>t_{k-1};\Pr\{X>t\mid X>t_{k-1}\}\ge\alpha\}=\bar{F}^{-1}(\alpha^k),$$

$$\vdots$$

$$t_N=\sup\{t>t_{N-1};\Pr\{X>t\mid X>t_{N-1}\}\ge\alpha\}=\bar{F}^{-1}(\alpha^N),$$

where $T=\bar{F}^{-1}(\alpha^{N+1})>t_N$ and $\bar{F}(t_k)=\alpha^k$. From a few algebraic manipulations, the expected operating cost can be represented as a function of α as

$$\begin{aligned}V_T(\mathbf{t}_N)\approx V_T(\alpha)=\sum_{k=0}^{N}\Big[&\{c_0(k+1)+b_0\}\{(1-\alpha^{k+1})-(1-\alpha^k)\}\\&+a_0\{\bar{F}^{-1}(\alpha^{k+1})-\bar{F}^{-1}(\alpha^k)\}(1-\alpha-k+1)\Big]\\&-a_0\sum_{k=0}^{N}\int_{F^{-1}(1-\alpha^k)}^{F^{-1}(1-\alpha^{k+1})}F(t)\mathrm{d}t+c_0(N+1)\bar{F}(T).\end{aligned} \quad (26)$$

By minimizing the expected operating cost with respect to α and substituting the optimal α into $\bar{F}^{-1}(\alpha^k)$, an aperiodic CP sequence is approximately derived. For this approximate algorithm, we need to determine the number of CPs in advance. Also, even though the exact number of CPs is known, the approximate algorithm does not guarantee an exactly optimal CP sequence.

4.2. Fluid Approximation

The next approximate algorithm focuses on the determination of the number of CPs. Let $n(t)$ be the average frequency of CP placement at time instant t. Then the time interval between two succsessive CPs at time t is approximately given by $1/n(t)$. Using $n(t)$, the expected operating cost over an infinite operation-time horizon is approximately expressed as a functional of $n(t)$:

$$\begin{aligned}V_\infty(\mathbf{t}_\infty)\approx V(n(t),F(t))&=\int_0^\infty\int_0^t c_0 n(x)\mathrm{d}x\mathrm{d}F(t)\\&+\int_0^\infty\left\{\frac{a_0}{2n(t)}+b_0\right\}\mathrm{d}F(t).\end{aligned} \quad (27)$$

Then, the optimization problem with an infinite-operation time horizon reduces to a variational culculus $\min_{n(t)}V(n(t),F(t))$. By solving the corresponding Euler equation, we have the optimal CP frequency

$$n_0(t)=\sqrt{a_0\lambda(t)/2c_0}.$$

On the other hand, in the case with a large operation-time horizon, Ozaki *et al.* [30,31] assume that the probability of the occurrence of a system failure can be negligible even if the file system survives after the time horizon, and derive the average frequency of CP placement by

$$n_1(t)=\sqrt{a_0 f(t)/2c_0(\beta-F(t))},$$

where the control parameter β is determined so as to satisfy $N+1=\int_0^T n_1(t)\mathrm{d}t$. Naruse *et al.* [40] also propose a modified average frequency of CP placement by

$$n_2(t)=(n_b/n_a)n_0(t),$$

where

$$n_a=\int_0^T n_0(t)\mathrm{d}t,\quad n_b=\left\lfloor\int_0^T n_0(t)\mathrm{d}t\right\rfloor, \quad (28)$$

and $\lfloor\cdot\rfloor$ is the integer part satisfying $x-1<\lfloor x\rfloor\le x$.

Hence, the optimal aperiodic CP sequence is determined by $k=\int_0^{t_k}n_1(t)\mathrm{d}t$ or $k=\int_0^{t_k}n_2(t)\mathrm{d}t$ for $k=1,2,\cdots,N$. Substituting the approximate CP sequence yields the following approximate expected operating cost:

$$\begin{aligned}V_T(\mathbf{t}_N)&\approx V_T(n_j(t))\\&=\int_0^T\int_0^t c_0 n_j(t)\mathrm{d}t\mathrm{d}F(t)+\int_0^T\left\{\frac{a_0}{2n_j(t)}+b_0\right\}\mathrm{d}F(t)\\&+c_0\left\{1+\bar{F}(T)\int_0^T n_j(t)\mathrm{d}t\right\}\end{aligned} \quad (29)$$

for $j=1,2$. As mentioned before, both two approximate algorithms do not also guarantee an exactly optimal CP sequence. However, it is worth mentioning that n_b in Equation (28) provides a very near value of the exact number of CPs. By setting n_b as the initial value of N in the forward or backward CP placement algorithm and adjusting its integer value via a simple bisection method, we can seek the number of CPs placed up to the finite operation time T.

The main difference between the constant hazard approximation and the fluid approximation is that the latter is based on the number of CPs by

$$N=\left\lfloor\int_0^T n_j(t)\mathrm{d}t\right\rfloor-1,$$

where $j=1,2,3$. For a given T and N, both forward and backward algorithms are applicable. By combining the fluid approximation with the forward or backward CP placement algorithm, it is possible to speed up the computation to calcurate the optimal CP sequense.

5. Numerical Examples

We calculate numerically the optimal CP sequence and the corresponding steady-state system availability. Suppose that the failure time distribution obeys the Weibull distribution:

$$F(t) = 1 - e^{-(t/\theta)^{\gamma}} \tag{30}$$

with shape parameter $\gamma(>0)$ and scale parameter $\theta(>0)$. In this case, the failure (hazard) rate $\lambda(t)$ and the inverse function $F^{-1}(t)$ in the algorithms are given by

$$\lambda(t) = \frac{f(t)}{\bar{F}(t)} = \frac{\gamma t^{\gamma-1}}{\theta^{\gamma}}, \tag{31}$$

$$F^{-1}(t) = \theta\{-\log(1-t)\}^{\frac{1}{\gamma}}, \quad 0 \le t \le 1. \tag{32}$$

For the operation-time horizon $T = 10, 15, 20$, we calculate the optimal CP sequence with an exact solution algorithm (forward or backward CP placement algorithm) and two approximate algorithms, and derive both the number of CPs and the steady-state system availability. When $\gamma < 1.0$, it is noted that the system failure time distribution is strictly DFR (Decreasing Failure Rate) and is not PF_2. Hence we apply only the backward CP placement algorithm for this case. In the case with PF_2, two exact solution algorithms provide the exactly same results, where the number of CPs is adjusted from the initial value n_b given in Equation (28). For the other model parameters, we set $c_0 = 0.003$, $a_0 = 0.200$ and $b_0 = 0.300$.

Figure 2 depicts the optimal CP time sequence with different shape parameter $\gamma = 0.5, 1.0, 2.0$ for $\theta = 10$ and $T = 20$, in the strict DFR case (a) with $\gamma = 0.5$, the optimal CP time behaves as convex functions with respect to the number of CPs for both exact and approximate methods. It can be seen that the two approximate methods poorly work except around 14-th CP. In the CFR (Constant Failure Rate) case (b) with $\gamma = 1.0$, the optimal CP time becomes a linear function, so all the methods give the almost same periodic CP time sequence. In the strict IFR (Increasing Failure Rate) case (c) with $\gamma = 2.0$, the optimal CP time shows concave functions of the number of CPs, and two approximate methods provide rather close values to the exact solution. In **Figures 3** and **4**, we show the optimal CP time sequence with $T = 15$ and $T = 20$. As the finite operation time becomes longer, the constant hazard approximation tends to be far from the exact solution, when the system failure time distribution is strict IFR. On the other hand, the fluid approximation gives the almost similar CP time sequence to the exact solution. However, in **Figure 3(a)**, the fluid approximation takes a bit differnt value of the optimal CP time sequence from the exact solution. In

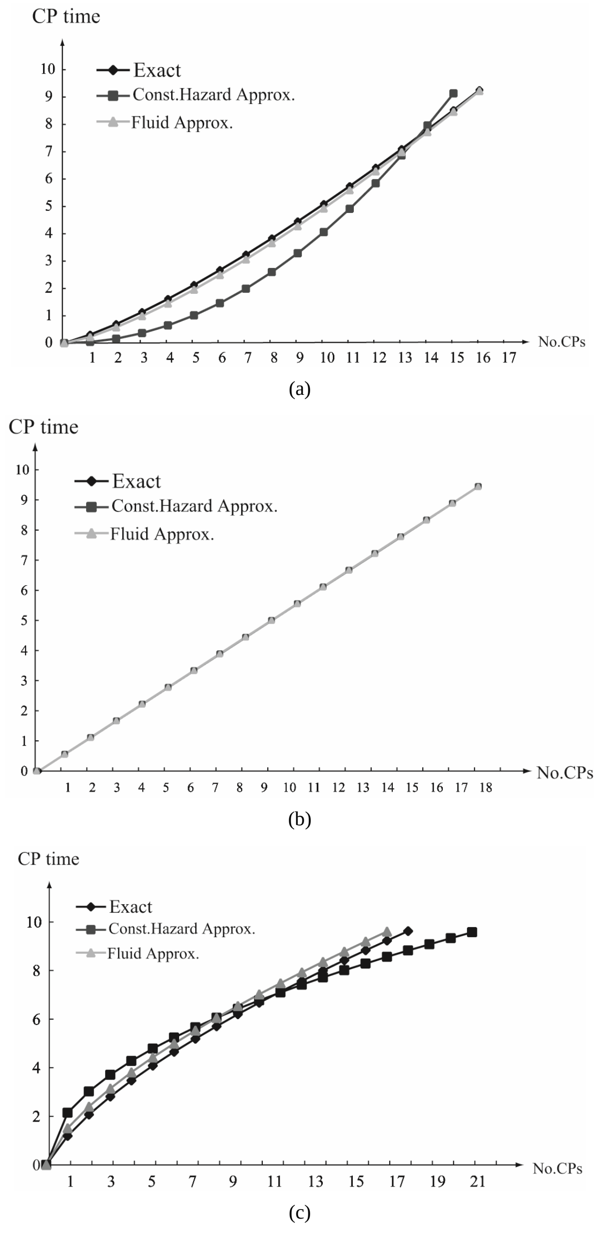

Figure 2. Aperiodic CP placement with different shape parameters for *T* = 10. (a) Case 1: *γ* = 0.5 and *θ* = 10; (b) Case 2: *γ* = 1.0 and *θ* = 10; (c) Case 3: *γ* = 2.0 and *θ* = 10.

other words, the computation accuracy for two approximate algorithms becomes worse as the shape parameter deviates from $\gamma = 1.0$ more and more. In **Figure 5**, we investgate the dependence of the optimal aperiodic CP time on the scale parametr and the operation time in the strict IFR case. Looking at (a) to (f), only the constant hazard approximation shows the different behavior from the exact solutions.

Next, we compare two approximation methods with the exact computation in terms of steady-state system availability more precisely. In **Table 1**, we present the steady-state system availability and the number of CPs

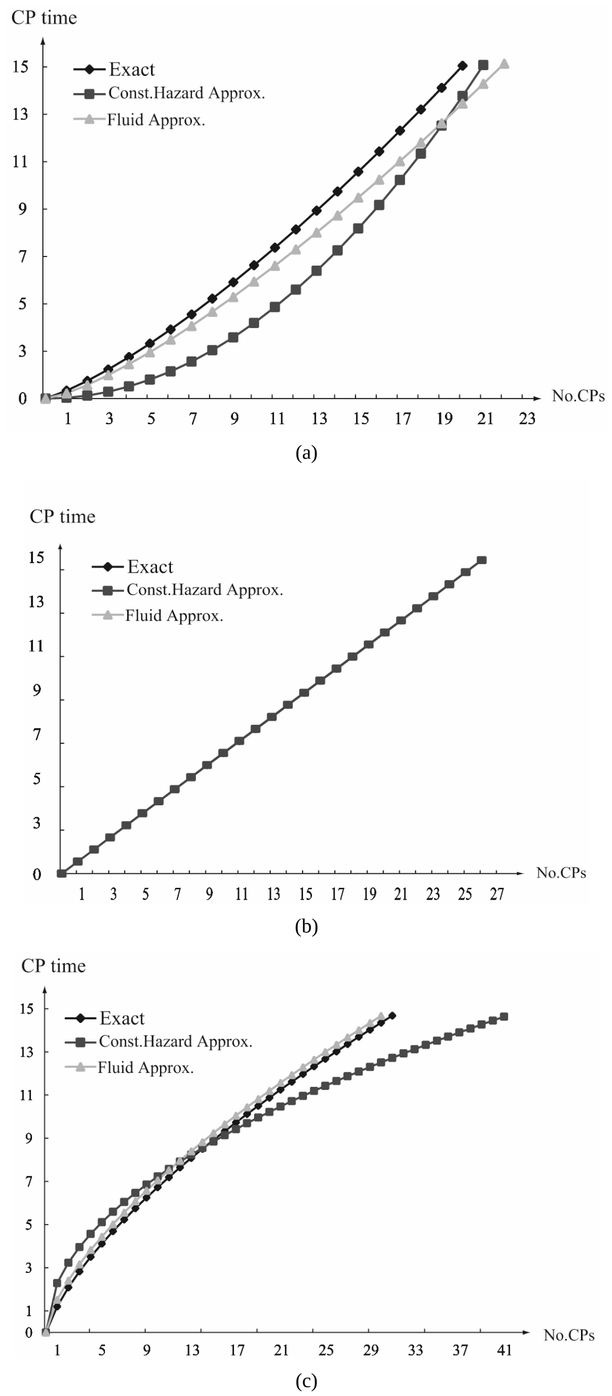

Figure 3. Aperiodic CP placement with different shape parameters for T = 15. (a) Case 1: γ = 0.5 and θ = 10; (b) Case 2: γ = 1.0 and θ = 10; (c) Case 3: γ = 2.0 and θ = 10.

for varying the failure parameters (γ,θ) when three algorithms are used. In the terms of approximate algorithms, $AV_T(\mathbf{t}_N)$ is caluculated by substituting each approximate CP sequence into Equation (5), so that $AV_T(\alpha^*)$ and $AV_T(n_b^*)$ in Equations (26) and (29) are calculated, where $k=\int_0^{t_k} n_2(t)\mathrm{d}t$ is used for the

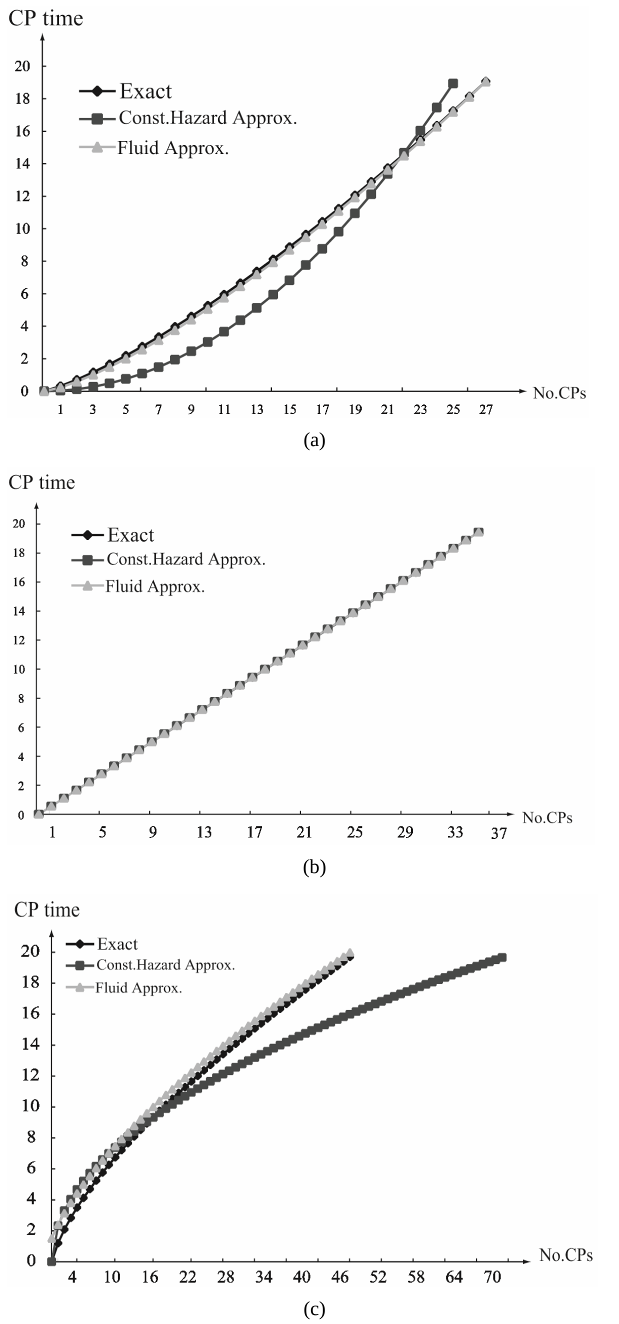

Figure 4. Aperiodic CP placement with different shape parameters for T = 20. (a) Case 1: γ = 0.5 and θ = 10; (b) Case 2: γ = 1.0 and θ = 10; (c) Case 3: γ = 2.0 and θ = 10.

fluid approximation. **Tables 1** and **2** present the dependence of the shape and the scale parameters on the steady-state system availability, respectively. When γ increases, then the system tends to fail as the operation time goes on, and the system availability does not always decrease in **Table 1**. In this case, the number of CPs does not always increase from **Table 1**. When θ increases, then the mean time to system failure (MTTSF) also increases and the steady-state system availability is

Table 1. Dependence of the shape parameter γ on the steady-state system availability. (a) Case 1: T = 10; (b) Case 2: T = 15; (c) Case 3: T = 20.

(a)

$T=10$	Hazard Approx.			Fluid Approx.			Exact.	
(θ, γ)	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	No. CPs
(10, 0.5)	98.7527	98.7529	15	98.7047	98.7611	17	98.7771	16
(10, 1.0)	97.4702	94.4704	17	97.4425	97.4764	18	97.4704	17
(10, 1.5)	97.1502	97.1502	19	97.1415	97.1919	16	97.1750	17
(10, 2.0)	97.0513	97.0513	20	97.0891	97.1511	17	97.1246	17
(10, 2.5)	97.0146	97.0146	20	97.0996	97.1730	16	97.1408	16
(10, 3.0)	97.0004	97.0004	21	97.1270	97.2119	15	97.1755	16
(10, 3.5)	96.9950	96.9950	21	97.1648	97.2552	15	97.2146	15
(10, 4.0)	96.9940	96.9940	22	97.1934	97.2954	14	97.2524	15
(10, 4.5)	96.9952	96.9952	22	97.2278	97.3336	14	97.2880	14
(10, 5.0)	96.9976	96.9976	22	97.2480	97.2480	13	97.3204	14

(b)

$T=15$	Hazard Approx.			Fluid Approx.			Exact.	
(θ, γ)	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	No. CPs
(10, 0.5)	98.5791	98.5786	21	98.5948	98.6065	23	98.6093	20
(10, 1.0)	96.9090	96.9091	26	96.8831	96.9164	27	96.9091	26
(10, 1.5)	96.3061	96.3062	33	96.3070	96.3589	28	96.3386	30
(10, 2.0)	95.9951	95.9952	41	96.0611	96.1259	31	96.0970	31
(10, 2.5)	95.8043	95.8045	50	95.9434	96.0188	33	95.9833	33
(10, 3.0)	95.6864	95.6866	62	95.8945	95.9796	35	95.9393	35
(10, 3.5)	95.6198	95.6200	76	95.8884	95.9828	37	95.9390	37
(10, 4.0)	95.5881	95.5883	93	95.9107	96.0119	40	95.9651	39
(10, 4.5)	95.5773	95.5775	113	95.9421	96.0532	42	96.0043	41
(10, 5.0)	95.5768	95.5770	134	95.9794	96.0981	45	96.0473	42

(c)

$T=10$	Hazard Approx.			Fluid Approx.			Exact.	
(θ, γ)	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	No. CPs
(10, 0.5)	98.4563	98.4564	25	98.4771	98.4887	28	98.4922	27
(10, 1.0)	96.5717	96.5718	35	96.5440	96.5798	36	96.5718	35
(10, 1.5)	95.9123	95.9123	49	95.9183	95.9710	41	95.9501	41
(10, 2.0)	95.6574	95.6575	71	95.7351	95.8005	48	95.7707	47
(10, 2.5)	95.5744	95.5744	102	95.7268	95.8025	55	95.7663	53
(10, 3.0)	95.5577	95.5577	146	95.7768	95.8613	63	95.8206	62
(10, 3.5)	95.5591	95.5591	208	95.8355	95.9284	72	95.8843	71
(10, 4.0)	95.5564	95.5564	282	95.8893	95.9908	82	95.9439	80
(10, 4.5)	95.5695	95.5696	260	95.9374	96.0467	94	95.9977	92
(10, 5.0)	95.5747	95.5749	212	95.9796	96.0966	108	96.0457	107

Table 2. Dependence of the scale parameter θ on the steady-state system availability. (a) Case 1: T = 10; (b) Case 2: T = 15; (c) Case 3: T = 20.

(a)

$T=10$	Hazard Approx.			Fluid Approx.			Exact.	
(θ,γ)	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	No. CPs
(2, 2.0)	83.5027	83.5027	197	83.5707	83.8078	86	83.7033	83
(5, 2.0)	92.2638	92.2638	49	92.3486	92.4677	34	92.4143	33
(8, 2.0)	95.6733	95.6733	26	95.7281	95.8064	21	95.7724	21
(10, 2.0)	97.0513	97.0515	20	97.0891	97.1511	17	97.1246	17
(13, 2.0)	98.2550	98.2551	14	98.2749	98.3222	13	98.3031	13
(15, 2.0)	98.7205	98.7205	12	98.7320	98.7736	11	98.7577	11
(18, 2.0)	99.1519	99.1519	10	99.1557	99.1907	9	99.1781	9
(20, 2.0)	99.3348	99.3348	9	99.3354	99.3565	8	99.3561	8
(23, 2.0)	99.5196	99.5196	8	99.5180	99.5451	7	99.5359	7
(25, 2.0)	99.6052	99.6052	7	99.5995	99.6268	6	99.6183	6
(28, 2.0)	99.6976	99.6976	6	99.6943	99.7152	6	99.7078	6
(30, 2.0)	99.7426	99.7426	6	99.7361	99.7572	5	99.7517	5

(b)

$T=15$	Hazard Approx.			Fluid Approx.			Exact.	
(θ,γ)	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	AV_k	No. CPs	$AV_T(\mathbf{t}_N^*)$	No. CPs
(2, 2.0)	83.5027	83.5027	235	83.5027	83.8078	156	83.7033	152
(5, 2.0)	92.1413	92.1413	112	92.2308	92.3497	63	92.2960	62
(8, 2.0)	94.8230	94.8230	56	94.9024	94.9820	39	94.9459	39
(10, 2.0)	95.9951	95.9952	41	96.0611	96.1259	31	96.0970	31
(13, 2.0)	97.2857	97.2857	29	97.3325	97.3822	24	97.3604	24
(15, 2.0)	97.8891	97.8891	24	97.7926	97.9685	21	97.9499	21
(18, 2.0)	98.5157	98.5157	20	98.5420	98.5776	17	98.5632	17
(20, 2.0)	98.8074	98.8074	17	98.8257	98.8580	15	98.8456	15
(23, 2.0)	99.1168	99.1168	15	99.1290	99.1568	13	99.1465	13
(25, 2.0)	99.2650	99.2650	14	99.2764	99.2999	12	99.2906	12
(28, 2.0)	99.4301	99.4301	12	99.4369	99.4589	11	99.4506	11
(30, 2.0)	99.5127	99.5127	11	99.5171	99.5379	10	99.5306	10

(c)

$T=20$	Hazard Approx.			Fluid Approx.			Exact.	
(θ,γ)	$AV(\boldsymbol{t}_n^*)$	AV_k	No. CPs	$AV(\boldsymbol{t}_n^*)$	AV_k	No. CPs	$AV(\boldsymbol{t}_n^*)$	No. CPs
(2, 2.0)	83.5027	83.5027	261	83.5706	83.8078	243	83.7033	228
(5, 2.0)	92.1404	92.1404	156	92.2300	92.4389	97	92.2952	95
(8, 2.0)	94.6948	94.6948	99	94.7790	94.8589	60	94.8226	60
(10, 2.0)	95.6574	95.6574	71	95.7351	95.8005	48	95.7707	47
(13, 2.0)	96.7417	96.7417	49	96.8056	96.8565	37	96.8333	37
(15, 2.0)	97.3154	97.3154	40	97.3694	97.4136	32	97.3936	31
(18, 2.0)	97.9871	97.9871	32	98.0289	98.0652	27	98.0490	27
(20, 2.0)	98.3284	98.3284	28	98.3631	98.3958	24	98.3816	24
(23, 2.0)	98.7178	98.7178	24	98.7444	98.7724	21	98.7605	21
(25, 2.0)	98.9147	98.9147	22	98.9367	98.9626	19	98.9520	19
(28, 2.0)	99.1420	99.1420	19	99.1590	99.1818	17	99.1726	17
(30, 2.0)	99.2592	99.2592	19	99.2737	99.2947	16	99.2862	16

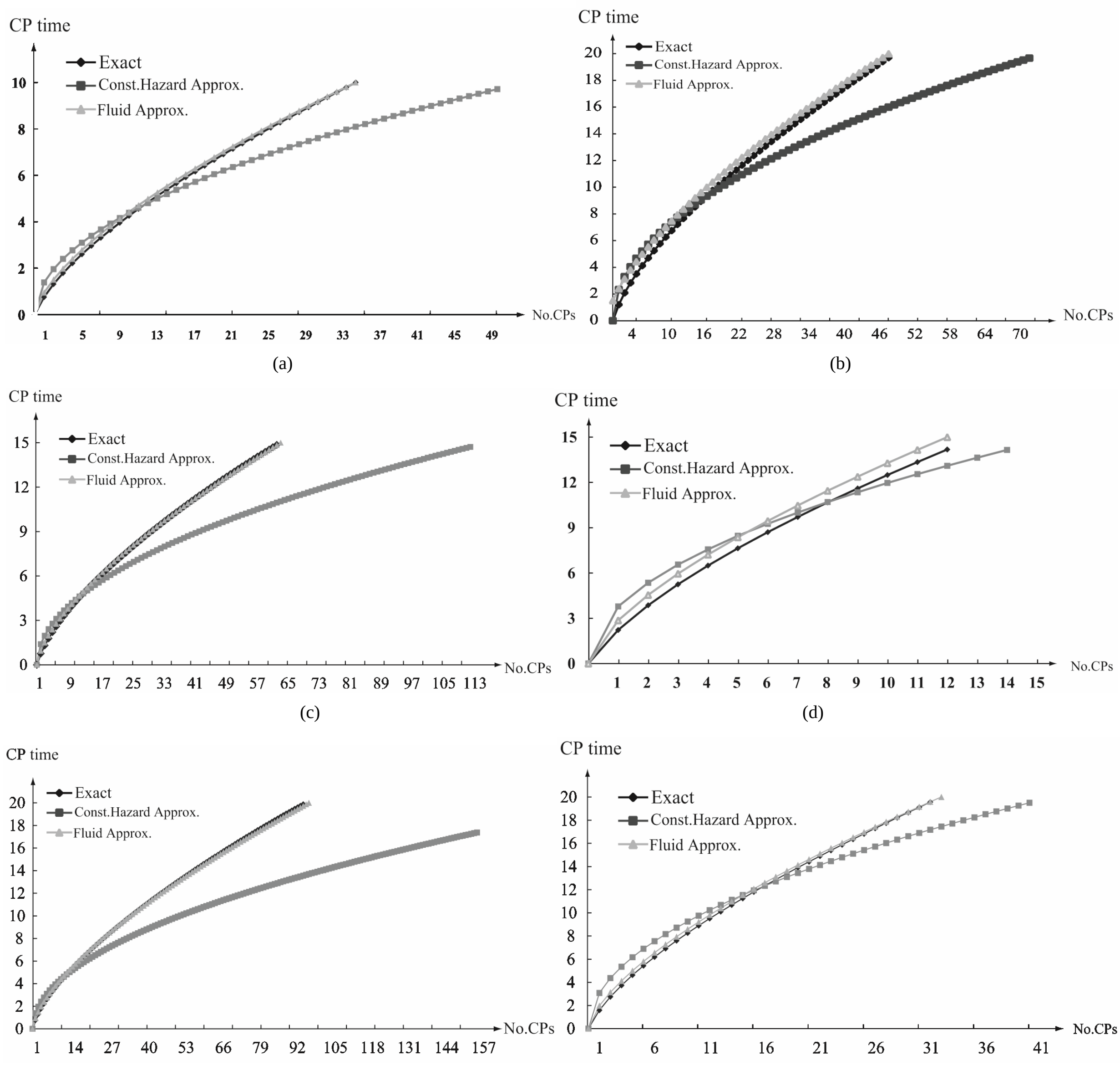

Figure 5. Aperiodic CP placement with different scale parameters and operation time for $\gamma = 2.0$. (a) Case 1: $\theta = 5$ and $T = 10$; (b) Case 2: $\theta = 20$ and $T = 10$; (c) Case 3: $\theta = 5$ and $T = 15$; (d) Case 4: $\theta = 25$ and $T = 15$; (e) Case 5: $\theta = 5$ and $T = 20$; (f) Case 6: $\theta = 20$ and $T = 20$.

expected to increse.This intuitive observation as well as the decreasing trend of the number of CPs are corect from **Table 2**. If we compare the minimum steady-state system availability calculated by the exact solution algorithm with the other ones, the relative error in both approximate methods can be found at the order of 0.01%. Especially, the reason why the constant hazard approximation works well is that it increases the number of CPs so as to increase the system availability. This implies that even the constant hazard approximation probvides the nice approximate performance on the maximum system availability. On the other hand, the number of CPs in the fluid approximation is also close to the exact one. Through these numerical examples, it can be concluded that if the steady-state system availability is evaluated with higher accuracy such as four or five nines, it is needed to apply the exact solution algorithms, where the initial value of the number of CPs is decided by the fluid approximation. Otherwise, *i.e.*, the three nines level is enough for calculating the steady-state system availability, then the fluid approximation provides rather good CP schedule.

6. Conclusion

In this article we have introduced some exact and approximate algorithms to create the aperiodic checkpoint schedule maximizing the steady-state system availability, when the file system operation terminates at a fixed time horizon. Since the determination of the number of checkpoints within the finite operation-time period has been an essential problem, we have combined the fluid approximation with the exact solution algorithm. In numerical examples with Weibull system failure time distribution, we have calculated the optimal aperiodic checkpoint sequence under different parametric circumstances. It has been shown that the combined algorithm with the fluid approximation could calculate effectively the exact solutions on the optimal aperiodic checkpoint sequence.

REFERENCES

[1] S. Hiroyama, T. Dohi and H. Okamura, "Comparison of Aperiodic Checkpoint Placement Algorithms," In: G. S. Tomar, R.-S. Chang, O. Gervasi, T. H. Kim and S. K. Bandyopadhyay, Eds., *Advanced Computer Science and Information* (*AST* 2010), *Communications in Computer and Information Science*, Vol. 74, Springer-Verlag, Berlin, 2010, pp. 145-156.

[2] V. F. Nicola, "Checkpointing and Modeling of Program Execution Time," In: M. R. Lyu, Ed., *Software Fault Tolerance*, John Wiley & Sons, New York, 1995, pp. 167-188,

[3] K. Naruse and S. Maeji, "Optimal Checkpoint Intervals for Computer Systems," In: S. Nakamura and T. Nakagawa, Eds., *Stochastic Reliability Modeling*, *Optimization and Applications*, World Scientific, Singapore City, 2010, pp. 205-239.

[4] J. W. Young, "A First Order Approximation to the Optimum Checkpoint Interval," *Communications of the ACM*, Vol. 17, No. 9, 1974, pp. 530-531. doi:10.1145/361147.361115

[5] F. Baccelli, "Analysis of Service Facility with Periodic Checkpointing," *Acta Informatica*, Vol. 15, No. 1, 1981, pp. 67-81. doi:10.1007/BF00269809

[6] K. M. Chandy, "A Survey of Analytic Models of Roll-Back and Recovery Strategies," *IEEE Computer*, Vol. 8, No. 5, 1975, pp. 40-47. doi:10.1109/C-M.1975.218955

[7] K. M. Chandy, J. C. Browne, C. W. Dissly and W. R. Uhrig, "Analytic Models for Rollback and Recovery Strategies in Database Systems," *IEEE Transactions on Software Engineering*, Vol. SE-1, No. 1, 1975, pp. 100-110. doi:10.1109/TSE.1975.6312824

[8] T. Dohi, N. Kaio and S. Osaki, "Optimal Ccheckpointing and Rollback Sstrategies with Media Failures: Statistical Estimation Algorithms," *Proceedings of* 1999 *Pacific Rim International Symposium on Dependable Computing* (*PRDC* 1999), Hong Kong, 16-17 December 1999, pp. 161-168.

[9] T. Dohi, N. Kaio and S. Osaki, "The Optimal Age-Dependent Checkpoint Strategy for a Stochastic System Subject to General Failure Mode," *Journal of Mathematical Analysis and Applications*, Vol. 249, No. 1, 2000, pp. 80-94. doi:10.1006/jmaa.2000.6939

[10] T. Dohi, N. Kaio and K. S. Trivedi, "Availability Models with Age Dependent-Checkpointing," *Proceedings of* 21*st Symposium on Reliable Distributed Systems* (*SRDS* 2002), Osaka, 13-16 October 2002, pp. 130-139.

[11] E. Gelenbe and D. Derochette, "Performance of Rollback Recovery Systems under Intermittent Failures," *Communications of the ACM*, Vol. 21, No. 6, 1978, pp. 493-499. doi:10.1145/359511.359531

[12] E. Gelenbe, "On the Optimum Checkpoint Interval," *Journal of the ACM*, Vol. 26, No. 2, 1979, pp. 259-270. doi:10.1145/322123.322131

[13] E. Gelenbe and M. Hernandez, "Optimum Checkpoints with Age Dependent Failures," *Acta Informatica*, Vol. 27, No. 6, 1990, pp. 519-531. doi:10.1007/BF00277388

[14] P. B. Goes and U. Sumita, "Stochastic Models for Performance Analysis of Database Recovery Control," *IEEE Transactions on Computers*, Vol. C-44, No. 4, 1995, pp. 561-576. doi:10.1109/12.376170

[15] P. B. Goes, "A Stochastic Model for Performance Evaluation of Main Memory Resident Database Systems," *ORSA Journal of Computing*, Vol. 7, No. 3, 1997, pp. 269-282. doi:10.1287/ijoc.7.3.269

[16] V. Grassi, L. Donatiello and S. Tucci, "On the Optimal Checkpointing of Critical Tasks and Transaction-Oriented Systems," *IEEE Transactions on Software Engineering*, Vol. SE-18, No. 1, 1992, pp. 72-77. doi:10.1109/32.120317

[17] N. Kobayashi and T. Dohi, "Bayesian Perspective of Optimal Checkpoint Placement," *Proceedings of* 9*th IEEE International Symposium on High Assurance Systems Engineering* (*HASE* 2005), Heidelberg, 12-14 October 2005, pp. 143-159.

[18] V. G. Kulkarni, V. F. Nicola and K. S. Trivedi, "Effects of Checkpointing and Queueing on Program Performance," *Stochastic Models*, Vol. 6, No. 4, 1990, pp. 615-648. doi:10.1080/15326349908807166

[19] V. F. Nicola and J. M. Van Spanje, "Comparative Analysis of Different Models of Checkpointing and Recovery," *IEEE Transactions on Software Engineering*, Vol. SE-16, No. 8, 1990, pp. 807-821. doi:10.1109/32.57620

[20] U. Sumita, N. Kaio and P. B. Goes, "Analysis of Effective Service Time with Age Dependent Interruptions and Its Application to Optimal Rollback Policy for Database Management," *Queueing Systems*, Vol. 4, No. 3, 1989, pp. 193-212. doi:10.1007/BF02100266

[21] P. L'Ecuyer and J. Malenfant, "Computing Optimal Checkpointing Strategies for Rollback and Recovery Systems," *IEEE Transactions on Computers*, Vol. C-37, No. 4, 1988, pp. 491-496. doi:10.1109/12.2197

[22] A. Ziv and J. Bruck, "An On-Line Algorithm for Checkpoint Placement," *IEEE Transactions on Computers*, Vol. C-46, No. 9, 1997, pp. 976-985. doi:10.1109/12.620479

[23] N. H. Vaidya, "Impact of Checkpoint Latency on Overhead Ratio of a Checkpointing Scheme," *IEEE Transactions on Computers*, Vol. C-46, No. 8, 1997, pp. 942-947. doi:10.1109/12.609281

[24] H. Okamura, Y. Nishimura and T. Dohi, "A Dynamic Checkpointing Scheme Based on Reinforcement Learn-

ing," *Proceedings of* 2004 *Pacific Rim International Symposium on Dependable Computing* (*PRDC* 2004), Tahiti, 3-5 March 2004, pp. 151-158.

[25] S. Toueg and Ö. Babaoglu, "On the Optimum Checkpoint Selection Problem," *SIAM Journal of Computing*, Vol. 13, No. 3, 1984, pp. 630-649. doi:10.1137/0213039

[26] N. Kaio and S. Osaki, "A Note on Optimum Checkpointing Policies," *Microelectronics and Reliability*, Vol. 25, No. 3, 1985, pp. 451-453. doi:10.1016/0026-2714(85)90195-7

[27] S. Fukumoto, N. Kaio and S. Osaki, "A Study of Checkpoint Generations for a Database Recovery Mechanism," *Computers & Mathematics with Applications*, Vol. 24, No. 1-2, 1992, pp. 63-70. doi:10.1016/0898-1221(92)90229-B

[28] S. Fukumoto, N. Kaio and S. Osaki, "Optimal Checkpointing Strategies Using the Checkpointing Density," *Journal of Information Processing*, Vol. 15, No. 1, 1992, pp. 87-92.

[29] Y. Ling, J. Mi and X. Lin, "A Variational Calculus Approach to Optimal Checkpoint Placement," *IEEE Transactions on Computers*, Vol. 50, No. 7, 2001, pp. 699-707. doi:10.1109/12.936236

[30] T. Ozaki, T. Dohi, H. Okamura and N. Kaio, "Min-Max Checkpoint Placement under Incomplete information," *Proceedings of* 2004 *International Conference on Dependable Systems and Networks* (*DSN* 2004), Florence, June 28-July 1 2004, pp. 721-730.

[31] T. Ozaki, T. Dohi, H. Okamura and N. Kaio, "Distribution-Free Checkpoint Placement Algorithms Based on Min-Max Principle," *IEEE Transactions on Dependable and Secure Computing*, Vol. 3, No. 2, 2006, pp. 130-140. doi:10.1109/TDSC.2006.22

[32] T. Dohi, T. Ozaki and N. Kaio, "Optimal Sequential Checkpoint Placement with Equality Constraints," *Proceedings of* 2nd *IEEE International Symposium on Dependable Autonomic and Secure Computing* (*DASC* 2006), Indianapolis, 29 September-1 October 2006, pp. 77-84.

[33] K. Iwamoto, T. Maruo, H. Okamura and T. Dohi, "Aperiodic Optimal Checkpoint Sequence under Steady-State System Availability Criterion," *Proceedings of* 2006 *Asian International Workshop on Advanced Reliability Modeling* (*AIWARM* 2006), Busan, 24-25 August 2006, pp. 251-258.

[34] H. Okamura, K. Iwamoto and T. Dohi, "A Dynamic Programming Algorithm for Software Rejuvenation Scheduling under Distributed Computation Circumstance," *Journal of Computer Science*, Vol. 2, No. 6, 2006, pp. 505-512. doi:10.3844/jcssp.2006.505.512

[35] H. Okamura, K. Iwamoto and T. Dohi, "A DP-Based Optimal Checkpointing Algorithm for Realtime Applications," *International Journal of Reliability, Quality and Safety Engineering*, Vol. 13, No. 4, 2006, pp. 323-340. doi:10.1142/S0218539306002288

[36] H. Okamura and T. Dohi, "Comprehensive Evaluation of Aperiodic Checkpointing and Rejuvenation Schemes in Operational Software System," *Journal of Systems and Software*, Vol. 83, No. 9, 2010, pp. 1591-1604. doi:10.1016/j.jss.2009.06.058

[37] T. Ozaki, T. Dohi and N. Kaio, "Numerical Computation Aalgorithms for Ssequential Checkpoint Placement," *Performance Evaluation*, Vol. 66, No. 6, 2009, pp. 311-326. doi:10.1016/j.peva.2008.11.003

[38] R. E. Barlow and F. Proschan, "Mathematical Theory of Reliability," Society for Industrial and Applied Mathematics, Philadelphia, 1996. doi:10.1137/1.9781611971194

[39] K. Naruse, T. Nakagawa and Y. Okuda, "Optimal Checking Time of Backup Operation for a Database System," In: T. Dohi, S. Osaki and K. Sawaki, Eds., *Recent Advances in Stochastic Operations Research*, World Scientific, Singapore City, 2007, pp. 131-143.

[40] K. Naruse, T. Nakagawa and S. Maeji, "Optimal Sequential Checkpoint Intervals for Error Detection," In: T. Dohi, S. Osaki and K. Sawaki, Eds., *Recent Advances in Stochastic Operations Research II*, World Scientific, Singapore City, 2009, pp. 213-224.