International Journal of Foundations of Computer Science
Vol. 21, No. 3 (2010) 329–344

DOI: 10.1142/S0129054110007283

INCREMENTAL CHECKPOINT SCHEMES FOR WEIBULL FAILURE DISTRIBUTION

MIHAELA PAUN*

College of Engineering and Science, Louisiana Tech University,
Ruston, Louisiana 71272, United States
Faculty of Finance, Spiru Haret University, Bucharest, Romania
mpaun@latech.edu
http://⟨www2.latech.edu/ mpaun/⟩

NICHAMON NAKSINEHABOON

College of Engineering and Science, Louisiana Tech University,
Ruston, Louisiana 71272, United States
nna003@latech.edu

RAJA NASSAR

College of Engineering and Science, Louisiana Tech University,
Ruston, Louisiana 71272, United States
nassar@latech.edu

CHOKCHAI LEANGSUKSUN

College of Engineering and Science, Louisiana Tech University,
Ruston, Louisiana 71272, United States
box@latech.edu
http://⟨www2.latech.edu/ box/⟩

STEPHEN L. SCOTT

Oak Ridge National Laboratory,
Oak Ridge, Tennessee 37831-6016, United States
scottsl@ornl.com

NARATE TAERAT

College of Engineering and Science, Louisiana Tech University,
Ruston, Louisiana 71272, United States
nta008@latech.edu

Received 13 April 2009
Accepted 24 November 2009
Communicated by Michael Palis

*To whom correspondence should be addressed.

Incremental checkpoint mechanism was introduced to reduce high checkpoint overhead of regular (full) checkpointing, especially in high-performance computing systems. To gain an extra advantage from the incremental checkpoint technique, we propose an optimal checkpoint frequency function that globally minimizes the expected wasted time of the incremental checkpoint mechanism. Also, the re-computing time coefficient used to approximate the re-computing time is derived. Moreover, to reduce the complexity in the recovery state, full checkpoints are performed from time to time. In this paper we present an approach to evaluate the appropriate constant number of incremental checkpoints between two consecutive full checkpoints. Although the number of incremental checkpoints is constant, the checkpoint interval derived from the proposed model varies depending on the failure rate of the system. The checkpoint time is illustrated in the case of a Weibull distribution and can be easily simplified to the exponential case.

Keywords: Incremental checkpoint; Weibull; failure rate.

1. Introduction

Computational power demand necessitates an increase in the physical size of computing systems. In spite of the large size of computing systems, system reliability has declined because of the high failure rate. Consequently, a system possibly spends extra time to re-compute an application after a failure occurs. A checkpoint/restart mechanism is a regular technique used to reduce the recomputing time. The systems in which a checkpoint/restart package has been installed will occasionally save the application states and resume the computing from the last saved state rather than the beginning. A checkpoint/restart mechanism, conversely, comes with auxiliary costs; time spent to save the application states, checkpoint overhead, and time spent to recover the application. Since in large-scale systems the checkpoint overhead is significantly large, an incremental checkpoint/restart mechanism has been introduced to reduce the checkpoint overhead by saving the changed pages instead of the whole application process [1]. In this paper the regular checkpoint will be called the full checkpoint to represent the whole process of an application being saved.

A checkpoint mechanism, unfortunately, comes at a price. Excessive checkpointing causes high overhead, but on the other hand, insufficient checkpointing leads to a large amount of time spent to re-compute the application. Young has proposed the first approximation of the optimal checkpoint interval [3] by which the cost due to random failures was minimized. Daly [4] has improved Young's work [3] to a higher order approximation by proposing and minimizing a more accurate cost function. Young's [3] and Daly's [4] studies have established a principle for derivation of the optimal checkpoint interval through which the cost function of the whole execution period was considered. However, both models assume that random failures follow a Poisson process with a constant failure which, in fact, does not adequately represent the failure characteristics of the analysis in [5]. However, there are some studies that do not possess this assumption.

All of the studies above are concerned with the optimization of the regular checkpoint mechanism. Yi et al [7] determined adaptive incremental checkpoint placements based on the expected execution time of an application. In this paper, after a checkpointing time is determined, the proposed mechanism will estimate the expected recovery time of the case of taking and skipping the determined incremental checkpoint. If the expected recovery time of skipping the determined checkpoint is cheaper than the other, the incremental checkpoint will be skipped. So adaptation is in response to the expected recovery time of whether taking or skipping the determined incremental checkpoint. However, to derive the application expected execution time they assumed that failure occurrences follow a Poisson process with a constant rate which may not be true. The checkpoint time is the value that makes the discriminant of the expected recovery costs of two cases equal to zero. In their experiment, the proposed model has been compared with the fixed checkpoint interval, and the result suggests that the checkpoint time derived from the adaptive model is larger than the fixed checkpoint interval.

The proposed model in this paper can be used for arbitrary distributions. Also, the proposed model integrates full checkpoints with the incremental checkpoint mechanism, so the recovery time of the proposed model is potentially less than that of the Yi's model. This is because in the proposed model only the incremental checkpoints, after the last full checkpoint are loaded, but in the Yi's study all incremental checkpoints performed are loaded. Because of the complexity of the recovery state in the incremental checkpoint mechanism, we introduce full checkpoints in the incremental checkpoint process in order to reduce the recovery cost from the incremental checkpoints after a failure. The detail of the scheme is described in Section 2.

The existing studies in the literature regarding incremental checkpoints have focused on implementation of the efficient incremental checkpoint mechanism. The model we propose focuses on the waste time of the full/incremental checkpoint mechanism and the full/incremental checkpoint placements. The following are a few studies that analyzed the overhead of the incremental checkpoints. Palaniswamy [1] compared the overhead of the regular (full) checkpoint mechanism against the incremental checkpoint mechanism. In their experiments, they assume an optimal checkpoint interval for the checkpoint mechanisms. According to their experiments, they concluded that the incremental checkpoint mechanism potentially outperforms the full checkpoint mechanism. Palaniswamy [1] did not study checkpoint scheduling, they only analyzed the overhead of the incremental checkpoint mechanisms.

One objective of this work is to determine the full and incremental checkpoint placements that optimize the expected wasted time (checkpoint overhead, recovery time, re-computing time) by modeling the optimal checkpoint frequency function. The derivation is given in Section 3. Although full checkpoints introduced to the checkpoint process bring down the number of incremental checkpoints required to load in the recovery state, the number of incremental checkpoints between two consecutive full checkpoints influences the overall recovery cost. Another objective of this work is to determine the constant number of incremental checkpoints that leads to the optimality of the expected wasted time. In Section 4, the proofs of the existence and the uniqueness of such a number are provided. Moreover, the estimation of the re-computing time coefficient, the ratio between the re-computing time and the checkpoint interval, is provided in Section 5. In aging systems, the number of checkpoints should increase over time because of the increasing failure rate. In the proposed Weibull model, while the number of incremental checkpoints between two consecutive full checkpoints is a constant, the checkpoint intervals derived from the model decrease. In Section 6. we illustrate the checkpoint placement function in the case where the time-to-failure follows a Weibull distribution.

In summary, in this paper we propose a waste time model due to failure occurrence. We determined the checkpoint placements that minimize the waste time and reduce the overhead. The optimality of the expected waste time can be obtained by the proposed checkpoint frequency function and the derived number of incremental checkpoints.

2. Incremental Checkpoint Scheme

Due to the high failure rate in high performance computing systems, when a system fails, an application that is running on it has to be re-computed from the beginning, which is a significant waste of time. The checkpoint/restart mechanism is a typical fault-tolerant mechanism that deals with the reliability issue in an application runtime environment. In general, there are two important states in the checkpoint/restart mechanism, first being the checkpointing state. In this state, the mechanism may save the whole current state of the application in a way that allows the application to be recovered from that state at later time. The mechanism will be in this state until a failure occurs. The time spent to save a state is called a checkpoint overhead. The second state is the recovery state which occurs after a failure occurs. In this state, the most recently saved state of the application will be loaded. After the saved state is loaded, the application execution is resumed. Still, the part of the application that has been computed after the last checkpointing but before the failure has to be re-computed. There are two costs involving the recovery state: the recovery time spent to load the last checkpointing and the re-computing time. Hence, with the checkpoint mechanism, after a failure, we can save an amount of time from re-computing an application from scratch by re-computing the application from the most recent checkpointing. The checkpointing which saves the whole application state is called full checkpointing, and it is a time consuming process, especially in high performance computing systems. As such, the incremental checkpoint/restart mechanism has been introduced. The first checkpoint is in general a full checkpoint. For other checkpoints in the checkpointing state, instead of saving the whole application, the section of memory that differs from the last checkpointing is saved. As a result, if the memory does not change much from the previous checkpointing, a checkpoint overhead is significantly reduced.

The checkpoint mechanism that uses the incremental checkpointing technique is called the incremental checkpoint/restart mechanism. While the incremental checkpointing technique helps to diminish the checkpoint overhead, it comes with additional costs in the recovery state in the incremental checkpoint/restart mechanism.

Because an incremental checkpointing saves only the differences between the current application state and the previous checkpointing, to resume the normal execution, every incremental checkpoint before the recent failure has to be loaded and combined together in the same order as they have been performed. Thus, the recovery state in the incremental checkpoint mechanism is more complicated and more costly than that in the full checkpoint mechanism. To reduce the complexity of the recovery state in the incremental checkpoint mechanism, after a set of incremental checkpoints a full checkpoint will be performed. As a result, in the checkpointing state of the incremental checkpoint mechanism, there are many sequences of a full checkpoint following by a set of incremental checkpoints. In the recovery state, consequently, after the presence of a failure, only the incremental checkpoints between the most recent failure going back to the last full checkpoint are loaded. Performing checkpoints results in the reduction of the re-computing time because the length of the re-computing time is bounded above by the checkpoint interval. Having an excessive number of checkpoints causes unreasonable overhead. The challenge arises in how often checkpoints should be performed and at what intervals.

Since the number of incremental checkpoints between two consecutive full checkpoints affects the recovery cost of the incremental checkpoint mechanism, the appropriate number of incremental checkpoints should be evaluated in order to minimize the total wasted time of the mechanism which consists of checkpoint overheads of both full and incremental checkpoints, recovery cost, and re-computing time.

In this paper we consider that the failures follow a Weibull distribution. The model is general and can be applied to any time varying distribution. In order to optimize the number of checkpoints between two consecutive failures, we propose a model that will checkpoint the system at different times t_i. These checkpoints will be performed at intervals of different lengths. Initially, when we start the application the checkpointing intervals are relatively large. As the application is running and we anticipate a failure to occur, the checkpointing intervals become smaller, as it is seen in Figure 1. The derivation for these checkpointing times is presented in Section 6.

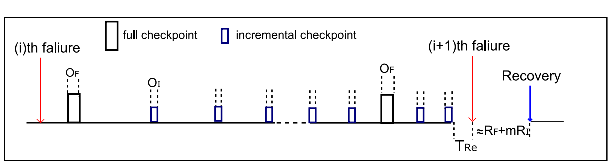

Fig. 1. Incremental checkpoint mechanism scheme.

The following assumptions are made before deriving the proposed model.

1. A running application may be interrupted by a series of random failures where the time between failures has a certain probability density function (PDF) $f(t)$.

2. The system failure can be detected by a monitoring mechanism.
3. The incremental checkpoint overhead (O_I) and the recovery cost of the incremental checkpoint (R_I) are assumed to be constant. In practice, we may use the average value of each parameter.
4. The first checkpoint in an application is a full checkpoint. After an application is recovered from failure, the first checkpoint is also a full checkpoint.
5. The number of incremental checkpoints between two consecutive full checkpoints (m) is a constant.

The incremental checkpoint overhead O_I is an average of each incremental checkpoint overhead. Although, for application purposes, there are both small and large incremental checkpoint overheads, this assumption may be valid because we aim to globally minimize the waste time caused by the incremental checkpoint mechanism.

Notations used throughout this paper are summarized in Table 1.

Table 1. Notation in the incremental checkpoint/restart model descriptions.

Parameter	Descriptions
O_F	Overhead of a full checkpoint
O_I	Overhead of an incremental checkpoint
T_{Re}	Re-computing Time
R_F	Recovery cost of a full checkpoint
R_I	Recovery cost of an incremental checkpoint
m	Number of incremental checkpoints between two consecutive full checkpoints
k	Re-computing time coefficient
T	Random variable of time-to-failure of the first failure

In the next section, the wasted time of the incremental checkpoint mechanism is derived. Denote the random variable of time-to-failure of the i^{th} failure by T_i. Hence, the sequence $T_i|i = 1, 2, ...$ can be treated as a renewal reward process. Then, by means of the calculus of variations, we derive a checkpoint frequency function that globally optimizes the expected wasted time for a general failure distribution.

3. Checkpoint Frequency Function

While the number of system components, in large-scale high performance computing (HPC) system keeps rising to achieve petscale performance, the failure rate of such systems has noticeably elevated. Fault tolerance (FT) mechanisms, such as checkpoint/restart and process mitigation, can help HPC applications mitigate such failures. However, using the FT mechanisms costs the application some additional overhead and resources as well. The idea of the frequency function when dealing with these fault-tolerant mechanisms is to minimize the overhead, reduce the cost function of performing checkpoints and mitigate the failure of the system. In order to optimize the the checkpoint placements we introduce the checkpoint frequency function defined bellow.

Definition 1. *Let $n(t)$ be the checkpoint frequency function such that:*

$$\int_{t_{i-1}}^{t_i} n(t)dt = 1, \tag{1}$$

where t_i, $(i = 1, 2, ...)$, is the i^{th} checkpoint placement (whether full or incremental checkpoint), and $t_0 = 0$.

From Eq.(1) the integral over the interval $[a, b]$ of the checkpoint frequency function gives the number of checkpoints whether full or incremental checkpoints in that period of time. Since there are $m + 1$ checkpoints in a sequence, the number of full checkpoints, from the beginning until a failure occurs, is approximated by $\frac{1}{m+1} \int_0^T n(t)dt$. Therefore the number of incremental checkpoints is estimated by $\frac{m}{m+1} \int_0^T n(t)dt$. The total checkpoint overhead is approximately $\frac{O_F + mO_I}{m+1} \int_0^T n(t)dt$. Also, the re-computing time can be estimated by the checkpoint frequency function. This relationship is illustrated in Figure 2. Since T is the value between these checkpoint placements, by the Mean Value Theorem, we can estimate the frequency of this interval by $n(T)$. k is the ratio of re-computing time T_{re} and the checkpoint interval which is estimated by $1/n(T)$. Therefore T_{Re} can be approximated by Eq.(2), see([9]), where k is a re-computing time coefficient variable between $(0, 1)$, its estimation will be derived in Section 5.

$$T_{Re} \approx \frac{k}{n(T)}, \; where \;\; k \in (0, 1). \tag{2}$$

Failure occurs at random, in the interval $\left(0, \frac{1}{n(T)}\right)$, as can be seen in Figure 2. As a result, the fraction $\frac{k}{n(T)}$ is a random variable, which makes k a random variable between $(0, 1)$ to be determined. Variable k is used to determine the re-computing time, T_{Re}.

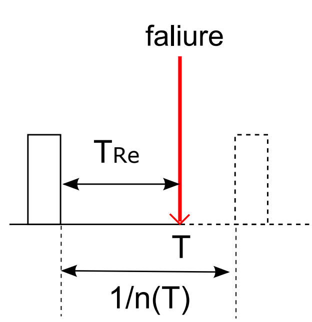

Fig. 2. Relationship between the re-computing time and the checkpoint interval.

Definition 2. *The wasted time is a random variable corresponding to the time to failure which can be expressed as*

$$W(T) = \frac{O_F + mO_I}{m+1}\int_0^T n(\tau)d\tau + \frac{k}{n(T)} + (R_F + mR_I). \tag{3}$$

In this process we allow more than one failure during the lifetime of a running application. However, the application will be restarted after each failure. The checkpoint process follows a renewal reward process in which W_i denotes the wasted time from the starting or restarting point to the i^{th} failure. The total wasted time of the running application can be expressed as $\sum_{i=1}^{r} W_i$, where r is the number of failures.

Denoting the first time-to-failure as T_1, the theorem of a renewal reward process [8] is given as

$$lim_{t\to\infty}\frac{\sum_{i=1}^{r} W_i}{t} = \frac{E[W_1]}{E[T_1]}. \tag{4}$$

Eq. (4) suggests that minimizing the overall expected wasted time is equivalent to minimizing the wasted time from the starting point to the first failure. In the rest of this paper, we consider only the wasted time due to the first failure and denote it as W, and we will talk about only the time-to-first failure denoted by T. Let $f(t)$ be the probability density function of the time-to-failure random variable. The expected wasted time can be expressed as

$$E[W] = \int_0^\infty \left[\frac{O_F + mO_I}{m+1}\int_0^t n(\tau)d\tau + \frac{k}{n(t)} + (R_F + mR_I)\right] f(t)dt. \tag{5}$$

The objective of this paper is to schedule full and incremental checkpoints that would minimize the expected wasted time. Applying the calculus of variations theory, we obtain the optimal checkpoint frequency function $n(t)$.

Theorem 3. *The checkpoint frequency function that minimizes the expected wasted time can be expressed as*

$$n(t) = \sqrt{\frac{(m+1)k}{O_F + mO_I}}\sqrt{\frac{f(t)}{1-F(t)}}. \tag{6}$$

Proof.

See the Appendix. □

Now we have the globally optimal checkpoint frequency function. However, to determine which checkpoint should be either a full checkpoint or an incremental checkpoint, the number of incremental checkpoints between two consecutive full checkpoints m that minimizes the expected wasted time would be evaluated. Hence, in the next section, we first prove the existence and the uniqueness of such number of incremental checkpoints and then show the polynomial equation used to calculate such number of incremental checkpoints m.

4. Number of Incremental Checkpoints m Minimizing the Expected Wasted Time

An incremental checkpoint mechanism effectively reduces the checkpoint overhead, but even so, the last full checkpoint and each following incremental checkpoint must be loaded to resume normal computing. Thus, the total recovery cost of the incremental checkpoint mechanism is larger than the recovery cost of the regular checkpoint mechanism. As a result, excessive incremental checkpoints would cause degradation of the application recovery state. In this section, we focus on evaluating an m value that gives the global minimum of the expected wasted time.

Lemma 4. *The expected wasted time as a function of m is strictly convex if $m \geq 0$.*

Proof.
See the Appendix. □

Next, we will show that there is a unique value of $m \geq 0$ that minimizes the expected wasted time.

Theorem 5. *The expected wasted time has a unique minimum point on $[0, \infty)$.*

Proof.
See the Appendix. □

We can evaluate the minimum point by finding a point that makes the first derivative disappear, i.e., by solving the following equation.

$$(O_F + mO_I)(m+1)^3 - \left(\frac{(O_F - O_I)D}{2R_I}\right)^2 k = 0. \tag{7}$$

So far we have already derived the optimal checkpoint frequency function and proved that there is a unique number of incremental checkpoints between two consecutive full checkpoints that minimizes the expected wasted time. The only parameter that we have not shown an approach to estimate is the re-computing coefficient k, and we will do so in the next section.

5. Estimation of the Re-computing Time Coefficient k

According to Figure 2, the re-computing time coefficient k can be estimated by the ratio between a re-computing time $T_{Re}$and a checkpoint interval. In addition, by the definition of the re-computing time, it is the interval between the last checkpoint and the failure. Clearly the re-computing time T_{Re} is a random variable depending on the time-to-failure T.

Definition 6. *Let k denote the re-computing time coefficient which can be expressed as*

$$k = \frac{T_{Re}}{t_i - t_{i-1}} = \frac{T - t_i}{t_i - t_{i-1}}. \tag{8}$$

Since k depends on the re-computing time T_{Re}, we will first find the expected value for T_{Re} of each checkpoint interval. To obtain such expected value, we need the following definition.

Definition 7. *Excess life is a random variable, $X \geq 0$, which denotes system survival until time $t + X$ given that it survives until time t. We denote the CDF, PDF, and the expected value of X as follows.*

$$\begin{aligned} F(t+x|t) &= P(T < t+x|t > t) \\ f(t+x|t) &= \frac{dF(t+x|t)}{dx} \\ E(X) &= \int_0^\infty xf(t+x|t)\,dx. \end{aligned} \tag{9}$$

Since, in our model, each checkpoint time t_i is the time that we expect the presence of a failure, the re-computing time during each checkpoint interval (t_{i-1}, t_i), T^i_{Re}, is a random variable defined on the interval $(0, t_i - t_{i-1})$. According to the excess life definition, the re-computing time of each checkpoint interval T^i_{Re} can be calculated as

$$E(T^i_{Re}) = \frac{\int_0^{t_i - t_{i-1}} xf(t_{i-1}+x|t_{i-1})\,dx}{\int_0^{t_i - t_{i-1}} f(t_{i-1}+x|t_{i-1})\,dx}. \tag{10}$$

Therefore, for the expected k of the i^{th} checkpoint interval (t_{i-1}, t_i) denoted, by $\bar{k}_i$, we obtain

$$\bar{k}_i = \frac{E(T^i_{Re})}{t_i - t_{i-1}}. \tag{11}$$

Hence, the expected k denoted by $\bar{k}$ can be expressed as

$$\bar{k} = \sum_{i=1}^{N} P_i \bar{k}_i / \sum_{i=1}^{N} P_i, \tag{12}$$

where $P_i = P(t_{i-1} + x|t_{i-1})$ and N is the number of checkpoints.

Now, we are ready to calculate the checkpoint placements of both full and incremental checkpoints. Since there are some studies indicating that times between failures fit well a Weibull distribution [5, 9], in the next section, we will illustrate the checkpoint placement function obtained from the checkpoint frequency function for the Weibull distribution. Moreover, for the Weibull distribution, the shape parameter is greater than 1 if the failure rate increases over time, so the number of checkpoints performed in a given time period should be increasing. On the other hand, the shape parameter is less than 1 if the failure rate decreases over time. In this case, the checkpoint frequency should be decreasing. As such, we also prove in the next section that the derived checkpoint frequency has the above properties.

6. Optimal Checkpoint Placements for the Weibull Distribution

To derive the checkpoint placement function for the Weibull distribution, we first recall the PDF and CDF of the Weibull distribution with the shape parameter β and the scale parameter α, respectively.

$$f(t) = \frac{\beta}{\alpha}\left(\frac{t}{\alpha}\right)^{\beta-1} e^{-(t/\alpha)^{\beta}}, \tag{13}$$

$$F(t) = 1 - e^{-(t/\alpha)^{\beta}}. \tag{14}$$

Then, we obtain the optimal checkpoint frequency function for the Weibull distribution as the following.

$$n(t) = \sqrt{\frac{(m+1)k}{O_F + mO_I}}\left(\frac{t}{\alpha}\right)^{\frac{\beta-1}{2}}\sqrt{\frac{\beta}{\alpha}}. \tag{15}$$

Theorem 8. *Let t_i be the i^{th} checkpoint placement whether full or incremental checkpoint.For $i = 1, 2, ...,$ we can express t_i as*

$$t_i = (i\frac{\beta+1}{2A})^{\frac{2}{\beta+1}}, \tag{16}$$

where $A = \sqrt{\frac{(m+1)k}{O_F+mO_I}}\left(\frac{1}{\alpha}\right)^{\frac{\beta-1}{2}}\sqrt{\frac{\beta}{\alpha}}$.

Proof.
See the Appendix. □

Next we will show that the checkpoint intervals $(t_i - t_{i-1})$ derived from Eq. (32) decreases if the system is aging ($\beta > 1$), and increases if the system gains in reliability over time, ($\beta < 1$).

Theorem 9. *The checkpoint interval $I(i)$ is decreasing if the shape parameter β is greater than 1, and it is increasing if β is less than 1.*

Proof.
See the Appendix. □

We note that the parameters k and m in Eq. (16) can be obtained from Eq. (12) and Eq. (7), respectively. Because k and m are related, in practice, we have to calculate both values at the same time. The following is an algorithm based on the fixed point approach to estimate the values of k and m.

Algorithm to calculate the re-computing time coefficient k and the number of incremental checkpoint m
INPUT: O_F, O_I, R_I, and Threshold
OUTPUT: k and m
Step 1: Initialize $\hat{k} = 0.5$.

(Find $\hat{m}$ corresponding to $\hat{k}$)

Step 2: Calculate $\hat{m}$ by solving Eq. 7

(Finish Finding $\hat{m}$ corresponding to $\hat{k}$)

Step 3: Calculate the checkpoint placement sequence $t_1, t_2, ..., t_N$ corresponding to $\hat{k}$ and $\hat{m}$ by Eq. 16

Step 4: Calculate $\bar{k}$ from Eqs. 10-12

Step 5: IF $|\bar{k} - \hat{k}| \leq$ Threshold

THEN Set $k = \bar{k}$ DONE

ELSE Set $\hat{k} = \bar{k}$ and repeat Step 2

It is worth mentioning that our model can be used for the case of a general failure distribution, but the computation might be more complex than that for the Weibull distribution.

7. Conclusion

Because of the high overhead of the regular checkpoint mechanism, an incremental checkpoint mechanism has been introduced with the aim of reducing the overhead. However, in order to recover the computation state after a failure takes place, each increment state saved is loaded, which causes a costly recovery time. In the incremental checkpoint scheme in this paper, full checkpoints are performed occasionally, depending on the appropriate number of incremental checkpoints between two consecutive full checkpoints. We have shown that this optimal number of incremental checkpoints exists and it is unique. Moreover, the optimal checkpoint frequency function has been derived for any distribution of the time-to-failure. The checkpoint time is illustrated in the case of a Weibull distribution which can be simplified to the exponential case. The optimality of the expected wasted time can be achieved theoretically by the proposed checkpoint frequency function and the derived number of incremental checkpoints.

In the future, the proposed model can be generalized in many ways. Two useful extensions would be first to consider a varying incremental checkpoint overhead because the size of the incremental checkpoint overhead depends on memory changes, which may be significant in some applications and second to enable both checkpoint and migration mechanisms.

Appendix

Proof. of Theorem 3.

First we define $y(t) = \int_0^t n(\tau)d\tau$. Then, $y'(t) = n(t)$ Thus, Eq. (5) becomes

$$E[W] = \int_0^\infty \left[\frac{O_F + mO_I}{m+1} y(t) + \frac{k}{y'(t)} + (R_F + mR_I)\right] f(t)dt. \tag{17}$$

Next, we let $h(y, y', t) = \left[\frac{O_F + mO_I}{m+1} y(t) + \frac{k}{y'(t)} + (R_F + mR_I)\right] f(t)$. Thus, Eq. (17) becomes

$$E[W] = \int_0^\infty h(y, y', t)dt. \tag{18}$$

The extremum of Eq. (18) must satisfy the Euler-Lagrange's equation.

$$\frac{\partial h}{\partial y} - \frac{d}{dt}\left(\frac{\partial h}{\partial y'}\right) = 0. \tag{19}$$

Taking the partial derivative of h with respect to y and y', respectively, we have

$$\begin{aligned} \frac{\partial h}{\partial y} &= \frac{O_F + mO_I}{m+1} f(t) \\ \frac{\partial h}{\partial y'} &= -\frac{k}{(y'(t))^2} f(t). \end{aligned} \tag{20}$$

By substituting Eq. (20) into Eq. (19), we obtain.

$$\frac{O_F + mO_I}{m+1} f(t) + \frac{d}{dt}\left(\frac{k}{(y'(t))^2} f(t)\right) = 0. \tag{21}$$

Integrating from 0 to t on both sides of Eq.(21) and keeping in mind that f is a probability density function, we obtain

$$\frac{O_F + mO_I}{m+1} F(t) + \frac{k}{(y'(t))^2} f(t) = C, \tag{22}$$

where C is a constant.

We aim to obtain the unique solution of the differential equation in Eq. 22. Because the right-hand endpoint (∞) is undetermined, we will first show that the function y satisfy the two conditions in Eq. (23), as seen in [2].

$$\begin{aligned} y(0) &= 0 \\ \lim_{t\to\infty} \frac{\partial h}{\partial y'} &= 0. \end{aligned} \tag{23}$$

It is easy to see that the function y satisfies the first condition by its definition. The function y satisfies the second condition because $lim_{t\to\infty} f(t) = 0$.

$$\lim_{t\to\infty} \frac{\partial h}{\partial y'} = -\lim_{t\to\infty} \frac{k}{(y'(t))^2} f(t) = 0.$$

Applying the second condition in Eq. (23) to Eq.(22), we obtain that $C = \frac{O_F + mO_I}{m+1}$. By an algebraic manipulation, we obtain the unique solution of the differential equation, Eq.(22),

$$y'(t) = \sqrt{\frac{(m+1)k}{O_F + mO_I}} \sqrt{\frac{f(t)}{1 - F(t)}}. \tag{24}$$

Moreover, because $n(t) = y'(t)$, we obtain the checkpoint frequency function that minimizes the expected wasted time. □

Proof. of Lemma 4.

We will show that the second derivative of the expected wasted time with respect to m is positive. To show that, first we substitute Eq. (6) into Eq. (5) to obtain

$$E[W](m) = \sqrt{\frac{(O_F + mO_I)k}{m+1}} \int_0^\infty \left(\int_0^t \sqrt{\frac{f(\tau)}{1-F(\tau)}} d\tau + \sqrt{\frac{1-F(t)}{f(t)}} \right) f(t)dt + (R_F + mR_I). \tag{25}$$

We denote

$$D = \int_0^t \sqrt{\frac{f(\tau)}{1-F(\tau)}} d\tau + \sqrt{\frac{1-F(t)}{f(t)}} f(t)dt \tag{26}$$

Eq. (25) becomes

$$E[W](m) = \sqrt{\frac{(O_F + mO_I)k}{m+1}} \cdot D + (R_F + mR_I), \tag{27}$$

where $D \geq 0$.

The first and second derivative of the expected wasted time with respect to m can be expressed as follows.

$$\frac{\partial E[W](m)}{\partial m} = \frac{(O_I - O_F)D}{2} \sqrt{\frac{k}{(O_F + mO_I)(m+1)^3}} + R_I \tag{28}$$

$$\frac{\partial^2 E[W](m)}{\partial m^2} = \frac{(O_F - O_I)(3O_F + O_I + 4mO_I)Dk^2}{4} \sqrt{\frac{(m+1)^5}{(O_F + mO_I)k^3}}. \tag{29}$$

Because O_F and O_I are the full and incremental checkpoint overhead, respectively, $O_F - O_I > 0$. If $m \geq 0$, then $\frac{\partial^2 E[W](m)}{\partial m^2} > 0$. Hence, the expected wasted time is strictly convex. □

Proof. of Theorem 5.

Suppose for a contradiction that $\frac{\partial E[W](m)}{\partial m} < 0$, for all $m \geq 0$. Then, according to Eq. (28), we have

$$\frac{(O_I - O_F)D}{2} \sqrt{\frac{k}{(O_F + mO_I)(m+1)^3}} + R_I < 0$$

$$\sqrt{(O_F + mO_I)(m+1)^3} < \frac{(O_F - O_I)D}{2R_I} \sqrt{k}$$

$$(O_F + mO_I)(m+1)^3 < \left(\frac{(O_F - O_I)D}{2R_I}\right)^2 k. \tag{30}$$

Since the righthand side of Eq. (30) is a constant, for a large enough m_0 the lefthand side will be larger than or equal to the righthand side, hence a contradiction. Therefore, for all $m \geq m_0$ the lefthand side is larger than the righthand side. Hence, the expected wasted time is decreasing on $[m_0, \infty)$, and increasing on $[0, \infty)$.

Since the expected wasted time is strictly convex by Lemma 4, the uniqueness holds. □

Proof. of Theorem 8.

By the definition of the checkpoint frequency function $n(t)$,

$$
\begin{aligned}
1 &= \int_{t_i}^{t_{i+1}} n(t)dt \\
&= \int_{t_i}^{t_{i+1}} A \cdot t^{\frac{\beta-1}{2}} dt \\
&= \frac{2A}{\beta+1}\left(t^{\frac{\beta+1}{2}}|_{t_{i-1}}^{t_i}\right) \\
&= \frac{2A}{\beta+1}\left(t_i^{\frac{\beta+1}{2}} - t_{i-1}^{\frac{\beta+1}{2}}\right).
\end{aligned}
$$

Hence,

$$
t_i = \left(\frac{\beta+1}{2A} + t_{i-1}^{\frac{\beta+1}{2}}\right)^{\frac{2}{\beta+1}}. \tag{31}
$$

By induction, we obtain

$$
t_i = \left(i\frac{\beta+1}{2A}\right)^{\frac{2}{\beta+1}}, \tag{32}
$$

where $i = 0, 1, 2, ...$ □

Proof. of Theorem 9.

Let $I(i)$ be a checkpoint interval between t_{i-1} and t_i defined by $t_i - t_{i-1}$.

$$
\begin{aligned}
I(i) &= (i\frac{\beta+1}{2A})^{\frac{2}{\beta+1}} - ((i-1)\frac{\beta+1}{2A})^{\frac{2}{\beta+1}} \\
&= (i^{\frac{2}{\beta+1}} - (i-1)^{\frac{2}{\beta+1}})(\frac{\beta+1}{2A})^{\frac{\beta+1}{2A}},
\end{aligned} \tag{33}
$$

where $A = \sqrt{\frac{(m+1)k}{O_F+mO_I}}(\frac{1}{\alpha})^{\frac{\beta-1}{2}}\sqrt{\frac{\beta}{\alpha}} > 0$.

The first derivative of $I(i)$ can be expressed as follows.

$$
\begin{aligned}
\frac{d}{di}I(i) &= \frac{d}{di}(i^{\frac{2}{\beta+1}} - (i-1)^{\frac{2}{\beta+1}})(\frac{\beta+1}{2A})^{\frac{\beta+1}{2A}} \\
&= (\frac{\beta+1}{2A})^{\frac{\beta+1}{2A}}\frac{d}{di}(i^{\frac{2}{\beta+1}} - (i-1)^{\frac{2}{\beta+1}})
\end{aligned}
$$

$$= (\frac{\beta+1}{2A})^{\frac{\beta+1}{2A}} \frac{2}{\beta+1}[i^{(\frac{2}{\beta+1}-1)} - (i-1)^{(\frac{2}{\beta+1}-1)}].$$

Hence,

$$I'(i) = (\frac{\beta+1}{2A})^{\frac{\beta+1}{2A}} \frac{2}{\beta+1}[i^{(\frac{2}{\beta+1}-1)} - (i-1)^{(\frac{2}{\beta+1}-1)}]. \tag{34}$$

We can see that if $\beta > 1$, $I'(i) < 0$, and if $\beta < 1$, $I'(i) > 0$. Moreover, if $\beta = 1$ (i.e., failures follow the exponential distribution with rate β), the checkpoint interval is a constant. □

References

[1] A. C. Palaniswamy and P. A. Wilsey, An analytical comparison of periodic checkpointing and incremental state saving, *J. SIGSIM Simul. Dig.* **23**(1)(1993) 127–134, doi:10.1145/174134.158475.

[2] R. Weinstock, *Calculus of Variations: With Applications to Physics and Engineering*, (Courier Dover Publications, 1952).

[3] J. W. Young, A first order approximation to the optimum checkpoint interval, *J. Commun. ACM.* **17**(9)(1974) 530–531, doi:10.1145/361147.361115.

[4] J. T. Daly, A higher order estimate of the optimum checkpoint interval for restart dumps, *J. Future Gener. Comput. Syst.*. **22**(3)(2006) 303–312, doi:10.1016/j.future.2004.11.016.

[5] B. Schroeder and G. A. Gibson, A large-scale study of failures in high-performance computing systems, in *Proc. International Conference on Dependable Systems and Networks* (2006) 249–258, doi:10.1109/DSN.2006.5.

[6] Y. Ling and J. Mi and X. Lin, A Variational Calculus Approach to Optimal Checkpoint Placement, *IEEE Trans. Comput.* **50**(7)(2001) 699–708, doi:10.1109/12.936236.

[7] S. Yi, J. Heo, Y. Cho and J. Hong, Taking Point Decision Mechanism for Page-level Incremental Checkpointing based on Cost Analysis of Process Execution Time, *J. Inf. Sci. Eng.* **23**(5)(2007) 1325-1337.

[8] S.M. Ross, *Stochastic Processes*, 2nd en. (Wiley, 1995).

[9] Y. Liu, R. Nassar, C. B. Leangsuksun, N. Naksinehaboon, M. Paun, and S. L. Scott, An optimal checkpoint/restart model for a large scale high performance computing system, in *Proc. International Par- allel and Distributed Processing Symposium*, (Miami, Florida, 2008) pp.1-9.