An Optimal Checkpoint/Restart Model for a Large Scale High Performance Computing System

Yudan Liu¹, Raja Nassar¹, Chokchai (Box) Leangsuksun¹, Nichamon Naksinehaboon¹, Mihaela Paun¹, Stephen L. Scott²

¹College of Engineering & Science
Louisiana Tech University, Ruston, LA 71270, USA
²Computer Science and Mathematics Division, Oak Ridge National Laboratory
Oak Ridge, TN 37831, USA
¹{yli010, nassar, box, nna003, mpaun}@latech.edu, ²scottsl@ornl.gov

Abstract—The increase in the physical size of High Performance Computing (HPC) platform makes system reliability more challenging. In order to minimize the performance loss (rollback and checkpoint overheads) due to unexpected failures or unnecessary overhead of fault tolerant mechanisms, we present a reliability-aware method for an optimal checkpoint/restart strategy. Our scheme aims at addressing fault tolerance challenge, especially in a large-scale HPC system, by providing optimal checkpoint placement techniques that are derived from the actual system reliability. Unlike existing checkpoint models, which can only handle Poisson failure and a constant checkpoint interval, our model can deal with a varying checkpoint interval and with different failure distributions. In addition, the approach considers optimality for both checkpoint overhead and rollback time. Our validation results suggest a significant improvement over existing techniques.

Key word: Large-scale distributed system events log analysis, reliability, fault-tolerance, checkpoint/restart model, and HPC.

I. INTRODUCTION

Computational power demand for large challenging problems has significantly increased the physical size of HPC systems. As witnessed in Top500.org, the volatile top500 list currently has the LLNL/IBM Blue Gene/L, with 131,072 processors, as the world fastest system. The mean time between failures of such a system is inversely proportional to its size.² The larger the system, the lower is the reliability. Thus, challenges arise in handling reliability of such large-scale systems.

Since Message Passing Interface (MPI) is one of the most popular parallel programming paradigms, we target our reliability study on the system using the MPI tool. Normally, an MPI application is decomposed and executed among a group of nodes, where individual subtasks communicate through MPI. Because of a static view of an MPI environment [3], a single node failure would cause the whole application to fail and requires an application restart. The checkpoint/restart scheme has been widely used [5][7][8][9][11][12][13] to address application outages. In the existing studies, the optimal checkpoint placement strategy is either based on the cost function models [7][8][9] or Markov availability models [6][10][14]. In addition, it is typically assumed that the system failure is a Poisson process (with a fixed failure rate). However, in practice, the system failure may not always follow the Poisson model [4] and the overall system reliability is more complex than that of individual components.

In this paper, we propose a stochastic model for optimizing the checkpoint/restart scheme in an HPC environment. The reliability function is obtained by analysing historical failure data from the system event log files. Statistical methods such as distribution fitting and goodness-of-fit test were performed on the failure data. We then applied the theory of a stochastic renewal reward process for developing an optimal checkpoint/restart model that aims at minimizing an application waste time consisting of checkpoint overhead, checkpoint restart time, and rollback time. Also, we present an algorithm to estimate the rollback coefficient.(Section 4.4)

The rest of the paper is organized as follows. Section 2 introduces related work on optimal checkpoint/restart models. Failure data analysis and reliability prediction methods are discussed in section 3. In section 4, the mathematical solution of the reliability-aware optimal checkpoint/restart model is presented. Finally, results and conclusions are presented in sections 5 and 6.
II. RELATED WORK

The checkpoint/restart method is a typical fault tolerant technique in computer systems. Chandy [11][12] and Treaster [13] presented a review of the checkpoint/restart technique. Young [9] presented an optimal checkpoint and rollback recovery model, and obtained a constant optimal checkpoint interval by which the total waste time was minimized. Based on Young’s work, Daly [7][8] improved the solution to an optimal checkpoint placement from a first order to a higher order approximation. Both Young’s and Daly’s studies established a principle on a cost function through which the whole execution period was considered, and the optimal checkpoint interval was derived to minimize the output of the cost function. An alternate technique concerning the optimal checkpoint interval is to consider the system availability. Geist et al.[14], Plank et al. [6], and Wong et al.[10] presented Markov availability models, and obtained an optimal checkpoint placement that maximizes system availability. Essentially, with a particular system failure rate, minimizing the cost function is equivalent to maximizing system availability. In fact, most of existing optimal placement models assume that the system failure follows a Poison process with a constant failure rate, which in most cases is not a good representation for the actual system failure characteristic [4]. Nevertheless, there are some models that do not possess this assumption. [17][19] Ling et al. [19] presented optimal checkpoint scheduling models for an infinite horizon time by using calculus of variation technique. The authors theoretically concluded that the fixed checkpoint interval is optimal if and only if the system failure follows a Poison process. Ozaki et al.[17] extended the concept of the variational calculus in Ling to a finite horizon time and incomplete system failure information. However, both of these papers considered the rollback time as a linear function for demonstrating model applicability. In our paper, we represent the rollback time by a coefficient, named the rollback coefficient. Moreover, we propose a novel algorithm to estimate this coefficient in order to obtain an optimal checkpoint scheduling.

Recently, there were two studies [1] on the reliability-aware checkpoint placement approach for the Blue Gene/L. Oliner et al. [1] proposed a so-called cooperative checkpointing, which is a hybrid checkpointing scheme based on Young’s [9] periodic checkpointing model. The cooperative checkpointing scheme may reduce the checkpointing cost, especially in a large-scale parallel system. However, it risks increasing the rollback cost. Our technique addresses both overhead and rollback time.

III. RELIABILITY ANALYSIS IN A LARGE-SCALE SYSTEM

Before describing the reliability-aware checkpoint restart model, we first present the system reliability analysis in HPC platforms. The analysis is based on the historical failure information from HPC production systems.

A. Historical Failure Information

In a production environment, large-scale cluster systems are typically well-managed and tracked for user/system activities and any critical events. The event logs are important historical information especially for failure and reliability analysis. In this paper, we analysed the event logs of major HPC systems obtained from the Lawrence Livermore National Laboratory, LLNL. The log file contains significant system events, from past years, among four ASC machines, namely White, Frost, Ice, and Snow. We then performed a detailed analysis on these data sets. For the purpose of brevity, we present only results of the analysis on ASC White. White, the largest among the aforementioned systems, is a 512-node system. Each node is a 16-way symmetric multiprocessor (SMP), aggregated to a total of 8,192 processors in the system. Failure information consists of a large data set including significant failure events over the period of four years, from 2000 to 2004.

B. Methodology of Failure Data Analysis

For the analysis, the times between failures of each node were collected to determine the reliability of a system of k nodes. We used distribution fitting techniques and tested the results by goodness-of-fit tests.

Distribution fitting is an important technique in data analysis. The time between failures (TBF) is a continuous positive random variable. The distribution of TBF can be described by its probability density function (PDF) \( f(t) \), and cumulative distribution function (CDF) \( F(t) \).

In our study, four commonly used distributions, namely exponential, Weibull, gamma, and lognormal, were fitted to the TBF data set in order to identify which failure distribution the given system follows. We then applied the best fitted distribution to derive an optimal checkpoint placement. Detailed derivation of CDF/PDF and mean time to failure (MTTF) for each distribution can be found in [20].

The goodness-of-fit test is a statistical technique used to identify the distribution that best fits a specific data set. In this paper, the Chi-square and the Kolmogorov-Smirnov tests were used. We conducted 72 goodness-of-fit tests on different groups of nodes at different time periods. From the ASC White failure sample data set, our results (Table 1) indicated that the Weibull distribution gave the best overall fit among the 72 data sets.

<table>
<thead>
<tr>
<th>Distribution Type</th>
<th>Times of Best Fit</th>
<th>Percentages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weibull</td>
<td>58</td>
<td>80.56%</td>
</tr>
<tr>
<td>Exponential</td>
<td>8</td>
<td>11.11%</td>
</tr>
<tr>
<td>Gamma</td>
<td>4</td>
<td>5.56%</td>
</tr>
<tr>
<td>Lognormal</td>
<td>2</td>
<td>2.78%</td>
</tr>
</tbody>
</table>

TABLE 1: RESULTS OF THE GOODNESS-OF-FIT FOR DIFFERENT DISTRIBUTIONS.
IV. RELIABILITY-AWARE CHECKPOINT/RESTART MODEL

In this section, we present an optimal checkpoint/restart model with the rollback coefficient for fault-tolerant parallel applications. We assume that the application model is MPI and supports a coordinated checkpoint/restart mechanism similar to the one in LAM/MPI [16]. The characteristic of the model is directly related to the system reliability function where one of the node outage will result in an application outage.

A. Coordinated Checkpointing on a Distributed System

![Figure 1: Coordinated Checkpointing for a Parallel Application](image)

In our model construction, we experimented and measured actual checkpoint overhead based on the Berkeley Lab’s Linux Checkpoint/Restart (BLCR) implementation. BLCR is the Linux kernel-based coordinated Checkpoint/Restart technique, and is integrated with the LAM/MPI to provide checkpoint and restart for a parallel application.

Figure 1 shows a schematic diagram of the coordinated checkpoint mechanism in the case of a parallel application. Consider a parallel program running with k processes, P0, P1, ..., Pk, and coordinated checkpoint placements, Tc1, Tc2, ..., Tcm, that represent the checkpoint intervals between adjacent checkpoints. Let Tc1, Tc2, ..., Tcm be the overhead of each checkpoint. Since the coordinated checkpoint protocol behaves in a synchronized fashion, we assume that there is no time difference for each individual process checkpoint, and treat it as a single checkpoint overhead, Tsi. Thus, in this study, we focus on how to determine a checkpoint interval or placement that minimizes the waste time.

B. Checkpoint/Restart and Stochastic Renewal Process

We consider a failure model that allows more than one failure during the lifetime of a given application. Moreover, after each failure, the application will be restarted. Our checkpoint/restart model is shown in Figure 2. It follows a renewal reward process in which \( \omega_i \) denotes the \( i \)th time between failures in each repeated cycle.

We assume that the waste time (checkpoint overhead, restart time, and rollback time) of each cycle is a random variable, \( W_1, W_2, W_3, \ldots \), since it depends on when a failure occurs. Hence, the total waste time may be expressed as

\[
C_t = \sum_{i=1}^{m} W_i
\]

where \( m = \max \left \lfloor n \left( \sum_{i=1}^{n} \omega_i \right) \leq t \right \rfloor \), and \( C_t \) is called a renewal reward process.

![Figure 2: Checkpoint/restart as a stochastic renewal reward process](image)

![Figure 3: Behaviors of the Checkpoint/Restart model](image)
An important theorem by Ross [15] of a renewal reward process is given as
\[ \lim_{t \to \infty} \frac{E(\sum_{i=1}^{n} W_i)}{t} = \frac{E(W_i)}{E(\alpha_i)} \] (2)

In the checkpoint/restart model, Eq.(2) shows that the mean of the overall waste time (left hand side of the equation) can be expressed as a function of the mean waste time of the 1st cycle. This means that minimizing the overall time lost is equivalent to minimizing the waste time in the 1st cycle.

C. Checkpoint/Restart Model

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_c$</td>
<td>Checkpoint Interval</td>
</tr>
<tr>
<td>$T_s$</td>
<td>Checkpoint Overhead</td>
</tr>
<tr>
<td>$T_r$</td>
<td>Recovery Time</td>
</tr>
<tr>
<td>$T_b$</td>
<td>Time to rollback to the last checkpoint</td>
</tr>
<tr>
<td>$n(t)$</td>
<td>Checkpoint frequency function</td>
</tr>
<tr>
<td>$f(t)$</td>
<td>Probability density function of TBF</td>
</tr>
<tr>
<td>$\omega_i$</td>
<td>The cycle between failure i and failure $(i+1)$</td>
</tr>
</tbody>
</table>

Consider a checkpoint restart scheme in the first cycle $\omega_i$ (time between two failures). Figure 3 illustrates the model with parameters such as checkpoint interval ($T_c$), checkpoint overhead ($T_s$), restart ($T_r$) and rollback time ($T_b$). Through the rest of the paper, our failure model is based on the following assumptions:

1) A running application may be interrupted by a series of random failures $S_i$ ($i = 1, 2, 3, \ldots$), where the time between failures has a certain PDF, $f(t)$.

2) The system failure can be detected by a monitoring mechanism.

3) Checkpoint intervals need not be fixed, but can vary.

4) Each checkpoint overhead $T_i$ is a constant. In practice, we can take this constant to be the average value of multiple checkpoint overheads.

5) The system can be recovered from the last checkpoint, and the rollback cost $T_b$ is a period between the last checkpoint and the present failure.

6) The repair time $T_r$ is a constant.

Remark: Assumption 2 is satisfied since a well-managed system [3] can be engineered with an efficient mechanism to immediately detect the failure. In assumption 3, the checkpoint interval may be constant or variable depending on the PDF of system failure. Thus, the checkpoint interval can be described by a checkpoint frequency function, $n(t)$. The mathematical derivation of $n(t)$ is given in the next subsection. Finally, assumption 6 about $T_r$ is satisfied if there is a mechanism in place to replace the failed node with a spare node. In fact, HA-OSCAR and its new fault tolerance extension [3] can provide a transparent recovery time $T_r$ less than a minute.

In Figure 3, $\omega_i$ is the $i^{th}$ cycle between the $i^{th}$ and the $(i+1)^{th}$ failures. From the discussion in section 4.2, we recognize that an optimal checkpoint/restart model must follow a specific failure distribution. Therefore, we look for the best checkpoint placement sequence that minimizes the total waste time. In addition, according to the renewal reward process and from Eq.(2), one can minimize the total waste time by minimizing lost execution time in the first cycle, $\omega_i$.

Let the sequence of discrete checkpoint placements be $0 = t_0 < t_1 < \ldots < t_n$, and let $n(t)$ be the checkpoint frequency function. Therefore, for any given time interval from the beginning of each cycle, denoted by L, the number of checkpoint placements is given by
\[ N(L) = \int_{0}^{L} n(\tau) d\tau \] (3)

In Figure 3, the time lost $W_i$ in a given cycle $\omega_i$ can be expressed as
\[ W_i = T_s \int_{0}^{\omega_i} n(\tau) d\tau + T_b + T_r \] (4)

From assumption 6, we have that $T_r$ is a constant, and, from assumption 5, we suppose that the system can be successfully recovered from the last checkpoint. The relationship between rollback time $T_b$ and checkpoint interval is illustrated in Figure 4.

![Figure 4: the relationship between rollback T_b and checkpoint interval](image)

Ling et al. [19] and Ozaki et al. [17] considered the recovery cost including both rollback time and recovery time similar to those in our model. They represented their recovery cost by a function, called the recovery cost function. Moreover, they illustrated their model with respect to recovery cost by assuming the recovery function to be linear. Assuming linearity may be restrictive and may not lead to optimality. In our model, we consider a rollback coefficient $k$ ($0 < k < 1$) instead of a recovery cost function and propose an algorithm (section 4.4) to estimate this rollback coefficient. The rollback coefficient is general and can be determined for any system failure distribution. This makes our approach useful for application purposes.

Since $\omega_i$ is the value between these checkpoint placements, by the Mean Value theorem, we can estimate the frequency of
this interval by $n(\omega_i)$. Therefore, $T_b$ can be approximated by Eq.(5), where $k$ is a rollback coefficient variable between $(0,1)$.

\[ T_b \approx k/n(\omega_i), \quad (0 < k < 1) \quad (5) \]

Replacing $T_b$ in Eq.(4) by its value from Eq.(5) gives

\[ W_i = T_s \int_{0}^{\tau} n(\tau) d\tau + \frac{k}{n(\omega_i)} + T_r \quad (6) \]

According to the theorem of a renewal reward process in Eq.(2), the total waste time in the checkpoint and recovery process can be minimized by minimizing $E(W_i)$. Let $f(t)$ be the probability density function of time between failures. Then, the probability that the system fails in the interval $[t, t + \Delta t]$ is $f(t) \cdot \Delta t$. The expected waste time during a cycle in the checkpoint/restart process is

\[ E(W_i) = \int_{0}^{\tau} \left[ T_s \cdot x(t) + \frac{k}{x'(t)} + T_r \right] \cdot f(t) dt \quad (7) \]

Our interest is in determining the optimal checkpoint frequency $n^*(t)$ in order to minimize the expected waste time as defined by Eq.(7).

**Solution:** Letting $x(t) = \int_{0}^{\tau} n(\tau) d\tau$, Eq.(7) becomes

\[ E(W_i) = \int_{0}^{\tau} \left[ T_s \cdot x(t) + \frac{k}{x'(t)} + T_r \right] \cdot f(t) dt \quad (8) \]

Let $\Phi(x, x', t) = [T_s \cdot x(t) + \frac{k}{x'(t)} + T_r] \cdot f(t)$.

Based on the extreme value theorem, the minimum value satisfies

\[ \frac{\partial \Phi}{\partial x} - \frac{d}{dt} \frac{\partial \Phi}{\partial x'} = 0 \quad (10) \]

From Eq.(10), we obtain the optimal checkpoint frequency function in Eq.(11).

\[ n^*(t) = \frac{k \cdot f(t)}{\sqrt{T_s(1 - F(t))}} = \sqrt{\frac{k}{T_s}} \cdot \sqrt{R(t)}, \quad (11) \]

A detailed derivation of Eq.(11) can be found in [20].

**D. Rollback Coefficient Estimation**

In Figure 4, $T_b$ is the rollback time of the application recovered after the failure. It is the time interval between the last checkpoint and the failure, which is a random variable depending on the time when the failure occurs from the checkpoint placement.

In an application without checkpoints (Figure 5(a)), if a failure occurs at time $T_f$, then $T_b = T_f - t_0$. With checkpoints (Figure 5(b)), it is obvious that $T_b$ is a random variable which depends on the time the failure occurs.

Therefore, if we know the distribution of the time between failures, then $T_b$ can be estimated.

**Definition:** The rollback coefficient $k$, (RBCoef), is the ratio between the rollback time and the checkpoint interval in which a failure occurs. As such,

\[ k = \frac{T_b}{T_{i+1} - t_i} \]

To estimate $k$, we first obtain the expected rollback time for each checkpoint interval.

**Definition:** Excess life is a random variable, $S \geq 0$, which denotes system survival until time $t + S$ given that it survives till time $t$. We denote the CDF, the PDF, and the expected value of the excess life $S$ as the followings. (12)

\[ F(t + s) = P(t + s | t) \]

\[ f(t + s) = \frac{dF(t + s)}{ds} \]

\[ E(S) = \int_{0}^{\infty} sf(t + s | t)ds \]

In our checkpoint model, each checkpoint time, $t_i$, is the time that we expect a failure to occur. The rollback time during the interval $(t_i, t_{i+1})$, $T_{bi}$, is a random variable such that its value is in the interval $(0, t_{i+1} - t_i)$. According to the excess life definition, the expected value of the rollback time can be calculated as

\[ E(T_{bi}) = \int_{0}^{t_{i+1} - t_i} sf(t_i + s | t_i)ds \quad (13) \]

Therefore, for the expected $k$ of the $i$th checkpoint interval, $\bar{k}_i$, we obtain

\[ \bar{k}_i = E(T_{bi})/(t_{i+1} - t_i) \quad (14) \]
Hence, the expected \( k, \bar{k} \), can be expressed as
\[
\bar{k} = \frac{\sum_{i=1}^{N} P_{k_i}}{\sum_{i=1}^{N} P_i},
\]
where \( P_i = P(t_i + s \mid t) \) and \( N \) is the number of the checkpoints.

To estimate \( k \) iteratively, we use the idea of the Fix point method as the following procedure. First, we assume an initial value \( \hat{k} \) between 0 and 1. We then calculate the corresponding checkpoint sequence, \( t_1, t_2, \ldots, t_N \), from Eq.(11). Next, we calculate \( k \) corresponding to the checkpoint sequence using Eqs.(13)(14)(15). We repeat the above procedure by varying \( \hat{k} \) until we obtain a \( k \) value that is equal to \( \hat{k} \).

**Algorithm to estimate \( k \)**

**STEP 1:** Assume \( \hat{k} = a, a \in (0,1) \)

**STEP 2:** Calculate the checkpoint sequence
\( t_1, t_2, \ldots, t_N \) corresponding to \( \hat{k} \) from step 1.

**STEP 3:** Calculate \( \bar{k} \) from Eqs.(13)(14)(15) using the sequence in step 2.

**STEP 4:** IF \( \hat{k} = \bar{k} \),
\[ \text{then set} \quad k = \hat{k} = \bar{k} \quad \text{DONE} \]
ELSE repeat step 1

**V. MODEL EVALUATION**

**A. Model for the Weibull Distribution**

In this section, we evaluate the solution of our optimal checkpoint/restart model described by Eq.(11). From our production system reliability analysis, the goodness-of-fit results discussed in section 3 show that the Weibull distribution gave the best fit to the data 80.56% of the time and the exponential 11.11% of the time. This suggests that the assumption of an exponential failure distribution in the literature may not be valid. In the special case where the shape parameter \( \beta = 1 \), the Weibull distribution reduces to that of the exponential. In this paper, we evaluate the checkpoint/restart model for the Weibull distribution and compare the results with those of the exponential case.

For the Weibull distribution, we obtain from Eq.(11)
\[
n^*(t) = \frac{k\beta}{T_s \eta^\beta} \cdot t^{\beta-1} \cdot e^{-\frac{(t+s)\beta}{\eta}}
\]
(16)

Since \( n^*(t) \) is the checkpoint frequency function, we obtain
\[
t_{i+1}^* = \int_{t_i} t_{i+1} \frac{k\beta}{T_s \eta^\beta} \cdot t^{\beta-1} \cdot e^{-\frac{(t+s)\beta}{\eta}} \, dt = 1, \; i = 1, 2, 3, \ldots
\]
(17)

Therefore, the sequence of optimal checkpoint placements for the Weibull distribution as shown in Eq.(16) is given by
\[
t_i = \left( 1 + \frac{\eta^\beta}{T_s \beta \eta^\beta} \right)^{\frac{1}{\beta}} \quad k \in (0,1)
\]
(18)

where \( t_i \) is the \( i \)th checkpoint placement.

**B. Estimation of the Rollback Coefficient \( k \) for the Weibull distribution**

According to our rollback coefficient estimation in section 4.4, let \( \hat{k} = a, 0 < a < 1 \) and let its corresponding checkpoint time sequence be \( \{ t_1, t_2, \ldots, t_l \} \).

The CDF, PDF, and expected value of the excess life following a Weibull distribution can be derived from Eq.(12) to give
\[
F(t_i + s) = P[t_i + s \mid t] = 1 - e^{-\left(\frac{t_i+s}{\eta}\right)^\beta + \left(\frac{t}{\eta}\right)^\beta}
\]
(19)
\[
f(t_i + s) = \beta \cdot \left(\frac{t_i+s}{\eta}\right)^{\beta-1} \cdot e^{-\left(\frac{t_i+s}{\eta}\right)^\beta + \left(\frac{t}{\eta}\right)^\beta}
\]
(20)

\[
E(T_{bi}) = \frac{\eta^\beta}{\beta} \int_{t_{i+1}-t_i}^{\eta^\beta} \beta \cdot \left(\frac{t_i+s}{\eta}\right)^{\beta-1} e^{-\frac{(t_i+s)^\beta}{\eta}} \, ds
\]
(21)

By Substituting Eq.(21) into Eq.(14), we obtain the expected \( k \) of the \( i \)th checkpoint interval (\( \bar{k}_i \)) and for the expected \( \bar{k} \) we use Eq.(15).

In order to determine \( \beta \) and \( \eta \) in Eq.(16), we fitted the Weibull distribution to the actual failure data from the LLNL ASC White system during the one year period June, 2003 to May, 2004. The fitted Weibull distribution shown in Figure 6 has the shape parameter \( \beta = 0.50944 \) and the scale parameter \( \eta = 20.584 \). In our study, we set the checkpoint overhead as 2 minutes. Using these \( \beta \) and \( \eta \) values, we estimated \( k \) using the algorithm in section 4.4 and replacing Eq.(13) by Eq.(21).

Figure 7 shows a plot of \( \hat{k} \) (assumed \( k \)) vs expected \( k \) (\( \bar{k} \)). The value of \( \bar{k} \) is the point on the curve where \( \bar{k} = \hat{k} \). From our data set, \( \bar{k} \) approximates 0.44505 as shown in Figure 7.
Figure 6: Fitted Weibull distribution to the failure data set of the ASC White system

Figure 7: Expected k value of a failure data set from the ASC White system

Figure 8: a) b) c) d) Comparison between the waste times of the Risk-Based model and the Optimal model for different completion times with requested intervals of 30 minutes, 1 hour, 2 hours, and 2.5 hours, respectively.
C. Comparison Results

For evaluation purposes, we compared our checkpoint/restart model with the Risk-Based scheme from a recent large-scale system checkpoint study [1]. The major idea of this recent study is that the checkpoint placement should be decided from the system cost functions (i.e. reliability vs. performance), so the system may skip some checkpoints requested by the application. In this paper, we call this checkpoint as the requested checkpoint, and call the checkpoint interval requested by the application as the requested interval. For the Risk-Based algorithm, the requested checkpoint will be performed only when the waste time is more than the checkpoint overhead. Details on this model can be found in [1]. In our study, we defined the job completion time to include overall execution time, checkpoint overhead, recovery and rollback times.

In comparing the two models, we first obtain the best fitted Weibull distribution to the data from June, 2003 to May, 2004. Using this Weibull distribution, we determined the checkpoint sequences for both models. We then use these checkpoint sequences to run the simulations on the data from June, 2004 to August, 2004. Our purpose is to compare the waste times of both models by varying the completion time of the application with regard to two aspects. First, we studied the Risk-Based model with respect to the requested interval as shown in Figure 8. Second, the effect of the checkpoint overhead on both models was evaluated as shown in Figure 9 and Figure 10.

In Figure 8 a), b), c), and d), we set the requested checkpoint interval to be 30 minutes, 60 minutes, 90 minutes, and 120 minutes. According to the graphs, our model gives better results for all cases, and in the first case when the requested interval is 30 minutes, the Risk-Based model gives the worst result among the 4 time intervals considered. This is because it performs checkpoint more frequently. If we increase the requested interval, the waste time of the Risk-Based model will decrease until at some point of the requested interval, and the waste time will increase as shown in Figure 8 d). The increase of the waste time comes from an increase in rollback time because the Risk-Based model seldom performs checkpoints.

In Figure 9, the requested interval of the Risk-Based model was 1 hour. For the checkpoint overhead of 0.5, 1, and 2 minutes, the waste time of both models slightly differ from each others, but for the overhead of 5 and 10 minutes, the waste time of the Risk-Based model is larger than the waste time of our model. However, when the checkpoint overhead is 1 minute, the waste time of the Risk-Based model seems to be less than our model as shown in Figure 10. Therefore, if we set the requested interval suitably, the Risk-Based model is likely to be slightly better than our model.

In Figure 10, it is clear that our model provides an improvement over the Risk-Based model, especially when the application total completion time is relatively large. This is because our improvements come from a balance between checkpoint overhead and rollback time factors.

![Figure 9: a) Waste times of the Optimal model for checkpoint overheads of 0.5, 1, 2, 5, and 10 minutes. b) Waste times of the Risk-Based model for checkpoint overheads of 0.5, 1, 2, 5, and 10 minutes](image)
VI. CONCLUSION

In this paper, we have presented an optimal checkpoint/restart model for fault-tolerant applications in a large-scale HPC environment. In this model, the optimal time sequence of checkpoint placements was derived using the theory of a stochastic renewal reward process. The model is general and can be applied to any distribution of time between failures. The Weibull distribution for time between failures was best in representing the time between failures for our system. Hence, it was used to demonstrate the applicability of the model.

Comparison of our model with the existing scheme revealed that our model gave the minimum total waste time. In future work, it would be of interest to extend the model to include other checkpoint schemes and show the cost model derivation based on other distributions of time between failures such as the Gamma and the log normal distributions as well as improve the rollback coefficient estimate.

VII. REFERENCES