A Flexible Checkpoint/Restart Model in Distributed Systems

Mohamed-Slim Bouguerra[1,2], Thierry Gautier[2], Denis Trystram[1], and Jean-Marc Vincent[1]

[1] Grenoble University, ZIRST 51, avenue Jean Kuntzmann 38330 MONTBONNOT SAINT MARTIN, France
[2] INRIA Rhone-Alpes, 655 avenue de l'Europe Montbonnot-Saint-Martin 38334 SAINT ISMIER, France
{mohamed-slim.bouguerra,thierry.gautier,denis.trystram, jean-marc.vincent}@imag.fr

Abstract. Large scale applications running on new computing platforms with thousands of processors have to face with reliability problems. The failure of a single processor will cause the entire execution to fail. Most existing approaches to guarantee reliable executions are based on fault tolerance mechanisms. Coordinated checkpointing is one of the most popular technique to deal with failures in such platforms. This work presents a new model of coordinated Checkpoint/Restart mechanism for several types of computing platforms. The model is parametrized by the process failure distribution, the cost to save a global consistent state of processes and the number of computational resources. Through mathematical analysis of reliability, we apply this new model to compute the optimal interval between checkpoint times in order to minimize the average completion time. Model independency from the type of the failure law makes it completely flexible. We show that such a model may be used to reduce the checkpoint rate up to 20% in same cases and up to factor 4 the total overhead in same cases. Finally, we report some experiments based on simulations for random failure distributions corresponding to the two most popular laws, namely, the Poisson's process and Weibull's law.

Keywords: Fault tolerance - Reliability modeling - Checkpointing.

1 Introduction

Today the most powerful computing systems involve more and more processors. Reliability is a crucial issue to address while running applications on such large systems because the failure rate grows with the system size. From the IBM source, the BlueGene/L system with 65,536 nodes is expected to have a mean time between failure less than 20 hours [1][2][3]. The users expect several processors to fail during the execution of their applications. Hence, it is necessary to develop strategies for providing a reliable completion of large applications. Fault-tolerant systems have been extensively studied on various computing or

R. Wyrzykowski et al. (Eds.): PPAM 2009, Part I, LNCS 6067, pp. 206–215, 2010.

real-time systems and any approaches have been proposed for dealing with failures. We focus in this work on the coordinated Checkpoint/Restart approach which is one of the most popular mechanism used in practice [4][5][6][7][8]. In checkpointing-based approaches, the state of computations is saved periodically on a reliable storage [9]. Informally, checkpointing is a technique that enables to reduce the completion time of the application by saving intermediate states in a stable storage, and then, to restore from the last stored state when a failure occurs. In the case of distributed computations, checkpointing methods differ from each other in the way the processes are coordinated in order to capture the global state in a consistent manner or not. Coordinated checkpointing requires that all processes coordinate the construction of a consistent global state before they write the individual checkpoints in the stable storage. After one or many failures, the restart mechanism sets up each processor from the last checkpoint and then, it schedules again the tasks of the crashed processors on new ones.

In this paper, we focus on the minimization of both the completion time of an application and the Checkpoint/Restart overhead by determining the optimal interval between two consecutive checkpoints. The main contribution of this work is to derive a new probabilistic model of the execution time of parallel applications with stochastic processes. This explicit analytical cost model enables to address and solve several interesting problems. First, like most of other models, it can be used to determine the optimal interval length between two successive checkpoints that minimizes the checkpoint overhead. Moreover, it is possible to take into account a variable checkpoint cost that depends on residual workload. This issue is not detailed here however, the details can be found in the technical report [10]. Finally, the proposed model can be adapted to several types of computations since it is independent from the failure law type, this make it very flexible. We show how to use it for the two most popular laws (Poisson's process and Weibull distribution).

The rest of this paper is organized as follows: Section 2 briefly recalls and discusses the other existing models for coordinated checkpointing on parallel platforms. Section 3 describes the proposed analytical model that expresses the expected completion time. In Section 4, we detail how to apply this model to two case-studies for the distribution of failure occurrences with Poisson's process and Weibull's law. Before concluding, we report in Section 5 some experiments based on simulations for assessing the model in several scenarii.

2 Review of Related Works

Let us first recall the principle of the coordinated checkpoint/restart protocol proposed by Chandy et al. [9]. It served as a basis of many implementations of fault tolerant systems for high performance computing. Coordinated checkpointing requires that all processes periodically coordinate the construction of a consistent global state before writing the state of individual checkpoints in a stable storage. Based on simulation results, Elnozahy et al. [5] showed that this approach is the most effective fault tolerance mechanism in large-scale parallel

platforms. The aims of fault tolerance models is to find strategies that minimize the completion time of the execution. Young [8] proposed a checkpoint/restart model where the failures follow an exponential law. Moreover, checkpointing is assumed to be fault-free and to have a constant execution time. The basic periodic version of this model has been recently extended by Oliner et al. [7]. Daly proposed also in [4] an extension of Young's model where failures can occur during checkpointing and he derived a higher order of approximation. Both Young's and Daly's models are able to compute an optimal checkpoint interval that minimizes the completion time considering constant checkpoint times. Other works from Geist et al. [11], Plank et al. [12] studied stochastic models and determined an optimal checkpoint date that minimizes a cost function which corresponds to maximize the availability of the system. Yudan et al. [6] proposed a stochastic model and an optimal checkpoint interval which does not depend on the specific failure law, such as the Poisson's process used in [11] [12]. Moreover, in this model, it is possible to compute the optimal checkpoint interval that minimizes the expected wasting time in the protocol itself (checkpoint overhead, restart time and rollback time). Another variant of this model is proposed in [13], that modelizes the incremental checkpoint under Poisson's process.

We present an analytical model that expresses the expected completion time with respect to the distribution law of failures taking into account the checkpoint overheads. The most interesting differences in our model, is that it does not depend on a particular failure law and it reduces both of the checkpoint rate and total wast time. Finally it takes into account the variable size of checkpoint as an input of the problem and this issue is addressed in [10].

3 Probabilistic Execution Model

For modeling the execution of a parallel application in presence of failures and with Checkpoint/Restart mechanism, the scheme of modeling is organized as follows. Firstly, a basic model that describe the parallel application in presence of failures is presented. This first model allows to compute the expected completion time of the execution with a formal method. Finally, using this abstraction without checkpointing, we derive a global model that contains the Checkpoint/Restart mechanism. It allows, to establish a global formula that expresses the expected completion time of the execution.

3.1 Parallel Application Model

A parallel application is modeled by a residual computation work function denoted by ω_t which is the amount of work remaining at the time index t. If we draw this variation, the curve would be divide into two parts. The first part is characterized by a linear speed-up with a slope which depends on the number of computing processors m and overhead factor α (due to the parallelism). In this phase, all the processors are busy. The second part presents a less stiff curve. This means that there is not enough work for all the processors and some idle

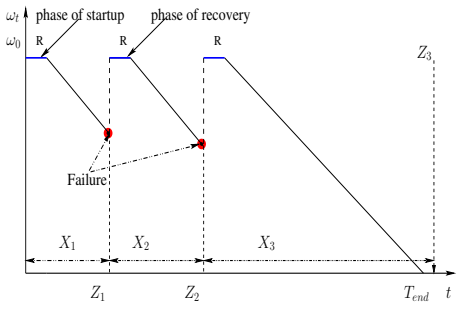

Fig. 1. Execution Model without checkpoint mechanism

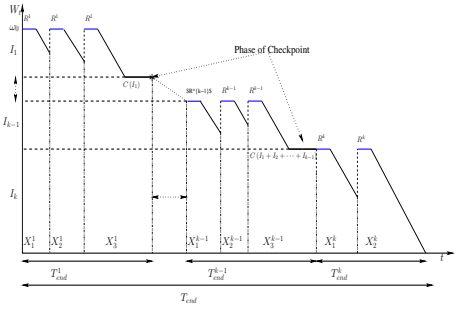

Fig. 2. Model of execution with the Checkpoint/Restart mechanism

time appear. These two parts will be determined by the amount of parallelism in the target application. In this work we consider only the first part with a linear speed-up until the completion of the application. This is not really a limitation for large scale applications. Under this hypothesis we have: $\omega_t = \omega_0 - \alpha m t$.

3.2 Execution Model without Checkpoint Mechanism

To establish the first fundamental theorem, we start by studying the execution without checkpoint mechanism, where the execution is restarted from the beginning after a failure (see Figure 1). This figure models the following renewal process. At the first step, there is a start-up in where the time is elapsed but the residual workload does not decrease. Then, the computation is started and the residual workload is decreased with a speed-up depending on the number of computing nodes. Finally, when a failure comes into one of these preview phases, all the computed work is lost. Thus, the application is restarted from the beginning after a duration R which denotes the restart cost.

For modeling the failures process, we consider mainly failures that affect the hardware or software system. Also, we assume the reliability of the failures detection tool and the time of failures detection is negligible. Then, let us suppose that failures do not propagate hence, the time between failures on different processors in the system are independent and identically distributed. From the above assumptions, we deduce that $\{X_i\}$ process (which denote the i^{th} inter-arrival of failures) is a renewal process and $\{X_i\}$ are positive independent and identically distributed random variables with distribution function $F(t)$ and probability density function $f(t)$. We recall that in this case the recovery process is considered as a deterministic process with duration R. From this abstraction, we deduce an analytical model that expresses the expected completion time T_{end}.

Let us consider a number of failures N during the execution such as: $\{N = n\} = \{X_n \geqslant \frac{\omega_0}{\alpha m} + R\}$. Consequently N is a stopping time process associated to the process $\{X_i\}$. Using Wald's equation [14] we can establish the following theorem.

Theorem 1. *Let T_{end} be the completion time of the execution and p the probability of failure during the execution with $p = F(\frac{\omega_0}{\alpha m} + R)$ then:*

$$\mathbb{E}(T_{end}) = \frac{1}{1-p}\mathbb{E}(X_1) + \frac{\omega_0}{\alpha m} + R - \mathbb{E}(X_N) = \frac{1}{1 - F(\frac{\omega_0}{\alpha m} + R)} \int_0^{\frac{\omega_0}{\alpha m}+R} 1 - F(x)dx.$$

proof in [10].

3.3 Execution Model with Checkpoint Mechanism

The next step consists in inserting the Checkpoint/Restart mechanism to the basic model. Let us divide the initial amount of work into k intervals denoted $I_1, I_2, \cdots, I_k$ such as each T_{end}^j denote the completion time of amount of work I_j. Therefore, the global execution model is drawn in Figure 2. At the first step, there is a start-up or a restart phase in where the time is elapsed but the residual workload does not decrease. Then, the computation is started and the residual workload is decreased with a speed-up depending on the number of computing nodes. After an interval of work denoted by I_j a checkpoint is triggered. Finally, when a failure comes into one of these preview phases, all the computed work after the last checkpoint will be lost. Thus, the application is restarted from the last checkpoint. In this model we consider that the checkpoint process is a deterministic process and its cost depends only on the amount of the work already done. We briefly recall that based on the design of the coordinated checkpoint mechanism, the data to save are the current state of each process and the computing data results associated to this one. More precisely, the amount of this data will depend on the work already done. Therefore, let s_j be the amount of work already done, where $s_j = \sum_{i=1}^{j} I_i$, the checkpoint cost function is denoted by $C(s_j)$. Also, the restart process is considered as a deterministic process and its cost depends on the amount of data saved in the last checkpoint. It is denoted by $R(r_j)$ where $r_j = \sum_{i=1}^{j-1} I_i$ is the work already done before the last checkpoint. Finally, we consider that the phases of checkpoint and recovery may also suffer from failures. From the above assumptions, we deduce that $\{X_i^j\}$ process (which denotes the i^{th} inter-arrival of failures in the j^{th} interval of checkpoint) is a renewal process, thus $\{X_i^j\}$ are positive independent and identically distributed random variables with distribution function $F(t)$ and probability density function $f(t)$.

Then, the main idea is to use the basic model (without checkpoint mechanism) for each I_j, as I_j is the amount of initial workload that should be done between the checkpoint number $j-1$ and j. Using this abstraction and under the these hypothesis we establish the following theorem:

Theorem 2. *Let k be the number of checkpoints and η_j be the amount of work between each checkpoint where $\eta_j = I_j + C(s_j) + R(r_j)$. Such as $\sum_{j=1}^{k} I_j = \frac{\omega_0}{\alpha m}$ then:*

$$\mathbb{E}(T_{end}) = \mathbb{E}(X_1) \sum_{j=1}^{k} \frac{1}{1-p_j} + \sum_{j=1}^{k} \eta_j - \mathbb{E}(X_N^j), \quad \textit{where } p_j = F(\eta_j).$$

Proof in [10]. Hence, Theorem 2 expresses the expected value of the completion time. It is worth noting it does not depend on the type of failures law which confirms the generality and flexibility of the proposed model. Therefore, in order to find the optimal amount of work between each checkpoint $(\hat{I_1}, \hat{I_2}, \cdots, \hat{I_k})$, we have to minimize the equation of Theorem 2.

4 Case Studies

In this section, we propose two case studies. In the first one, we express the optimal checkpoint interval when the failures process is modeled by a Poisson's process. Then, in the second one we also express the suboptimal solution when the failures followed a Weibull's law.

4.1 Poisson's Process Failures

One of the most common method to model the failures on computer devices, is the Poisson's process with a constant rate denoted by λ. In this case the expected completion time is given by this expression.

$$\mathbb{E}(T_{end}) = \frac{1}{\lambda}\sum_{j=1}^{k}[e^{\lambda\eta_j} - 1] \text{ where } \eta_j = I_j + C(s_j) + R(r_j). \tag{1}$$

Then, considering the cost of the checkpoint barrier as a constant cost such that $C(s) = C$ and $R(r) = R$, the following lemma is established.

Lemma 1. *When the cost of Checkpoint and restart is constant, then the optimal interval between each checkpoint is* $\frac{\omega_0}{\alpha mk}$ *where* k *is the checkpoint number.*

Proof in [10]. Then after finding the optimal interval between each checkpoint using Lemma 1, Equation (1) becomes:

$$\mathbb{E}(T_{end}) = \frac{k}{\lambda}(e^{\lambda(\frac{\omega_0}{\alpha mk}+C+R)} - 1). \tag{2}$$

Finally, to minimize this equation we use the Lambert's function denoted by $\mathcal{W}$. Then, the optimal interval between checkpoint is:

$$\frac{\omega_0}{\alpha mk} = 1 + \mathcal{W}(-e^{-1-\lambda\,(C+R)}) \tag{3}$$

Notice that Lambert's function could be computed numerically using Taylor's series, (more information about the proof can be found in [10]).

4.2 Using Weibull's Law

In this second case study, a Weibull's law with shape parameter β and scale parameter λ is used to model the failure distribution occurrences. Then, the expected completion time is given by:

$$\mathbb{E}(T_{end}) = \sum_{j=1}^{k} e^{(\lambda\eta_j)^\beta} \int_0^{\eta_j} e^{-(\lambda x)^\beta} dx. \tag{4}$$

Considering the case where a constant cost is spend to checkpoint the global application. Consequently, the following lemma can be established:

Lemma 2. *If the failure distribution is a Weibull's law and the checkpoint/ restart cost is constant, the suboptimal interval between each checkpoint is* $\frac{\omega_0}{\alpha m k}$*, where* k *is the number of checkpoints.*

Proof details in [10]. Then, using Lemma 2 the new expression of the expected completion time becomes:

$$k e^{(\lambda(\frac{\omega}{\alpha m k}+C+R))^{\beta}} \int_0^{\frac{\omega}{\alpha m k}+C+R} e^{-(\lambda x)^{\beta}} dx \tag{5}$$

Hence, to minimize Equation (5) in order to find $\hat{k}$, the Brent's numerical method from the GNU scientific library is used to find the minimum. More informations about the numerical method source code are in [10].

5 Simulations

5.1 Simulations Scenario

In this simulation we propose 3 different scenario to validate and to compare the proposed model. In the first and the second scenario a Poisson's process is used to generate random failures time with respect to different failure rates. Then, in the last scenario we use a Weibull's law to generate the failures time using the same value for the parametres λ and β described in [6]. These Weibull's parameters are the result of a statical analysis in [6] of failures log-files from the LLNL ASC White Computer system during the one year period June, 2003 to May, 2004. The best fitted distribution to the data is the Weibull distribution with the shape parameter of $\beta = 0.5094$ and the scale parameter of $\lambda = \frac{1}{20.584}$.

Then, to compare the model, we consider the Daly's model [4] in the first and in the second scenario then, in the 3^{th} scenario we consider the model proposed in [6] to compare our model when the failures follow a Weibull distribution. The main objective of these simulations is to study the behaviors of the model with respect to the variation of one form the different parameters i,e, (Failure rate, Checkpoint cost, Initial amount of work). Therefore, as scenario, we suppose that one parameter varies in a given interval and the others are constant. The different scenario configurations are in Table 1. For instance, in the first scenario the failure rate is increased in $[\frac{1}{2}, 96]$ by step of 5 while the checkpoint cost and the initial amount of work are constant. Then, for each setup we perform 10^4 simulations of execution with random failures. We notice that the same simulation is performed for both constant checkpoint cost 15 and 10 minutes only for scenario 3. Finally, It should be noted that the confidence interval reported in all simulations are at 95%.

Table 1. Scenario configurations

Scenario	Failure Law	Failure Rate (per day)	Checkpoint Cost (mins)	Amount Of Work
1	Poisson's Process	$\lambda \in [1/2, 96]$	$C = 10$	$\omega_0/\alpha m = 10\ days$
2	Poisson's Process	$\lambda = 1/2$	$C \in [1/2, 96]$	$\omega_0/\alpha m = 7\ days$
3	Weibull	$\lambda = 1/20.584, \beta = 0.5094$	$C = 10, C = 15$	$\omega_0/\alpha m \in [100, 2500]\ hrs$

5.2 Results and Discussions

Figure 3 reports the variation of the average completion time versus the variation of the failures rate, with respect to the configuration of scenario 1. As it can be seen in this figure both models perform the same average completion time and the same confident interval. Consequently, our model has the same performance in term of completion time as Daly's model when λ highly increases. The second result with configuration 1 is presented in Figure 4, that shows the variation of the checkpoint rate versus the failure rate. This figure reveals that our model outperforms Daly's model in term of checkpoint rate when λ becomes very large. Indeed, the proposed model has the same average completion time as Daly's model even though it reduces the checkpoint rate by to 20%. It should be noted that decreasing in the checkpoint rate leads to lower I/O request and communication on the system in terms of network communication and stable storage writing as well.

Secondly, to explain more why both models have the same completion time and FCM outperforms Daly's model in term of checkpoint rate. The average overhead due to failure and the average overhead due to checkpoint are depicted versus the variation of the checkpoint cost in Figure 5, for FCM in contrast to Daly's model. Hence, as it can be seen in this figures that the proposed model performs less checkpoint (discontinuous curve with square marker) than Daly's model (discontinuous curve without marker) but FCM loses more work due to failure then Daly's model. That means both models have the same wast time (Checkpoint overhead in addition to re-exectude work due to failures), consequently they have the same average completion time. These outcomes show that the proposed model (FCM) outperforms the Daly's model in terms of system I/O overhead while the completion time performance dose not change which is very interesting features. In fact, this is due to the assumption on the checkpoint cost that should be low in Daly's model. As a conclusion of the performances analysis, let us emphasize that both models have the same average completion time and confident interval which determines the stability of the model. Indeed, FCM reduces very well the checkpoint rate by to 20% and it is independent from the type of the failures law which make it flexible while Daly's model considers only the Poisson's process. Finally in the 3^{th} scenario, we intend to compare the average wast time versus the variation of the initial amount of work with the model proposed in [6]. The result of this simulation is depicted in Figure 6, in which the same scenario is repeated for two values of checkpoint cost. This figure reveals for the both values of checkpoint cost, that FCM reduces by a factor of 3 the total wast time and reduces as well the confident intervals especially when the initial amount of work is relatively large. This is due to the point-fixe method

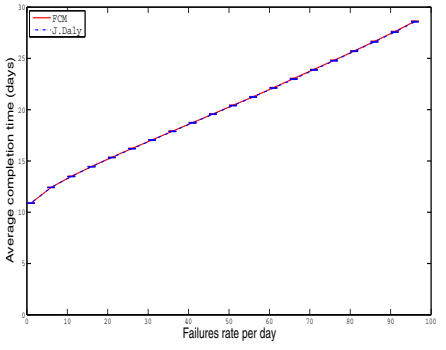

Fig. 3. Variation of the average completion times (Scenario 1)

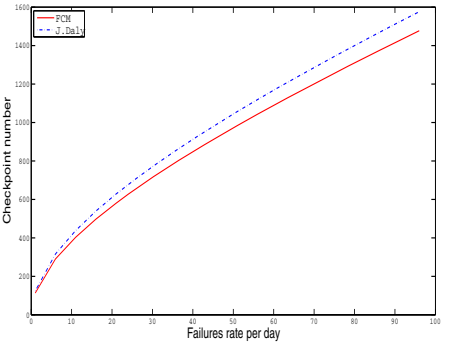

Fig. 4. Variation of the optimal checkpoint number (Scenario 1)

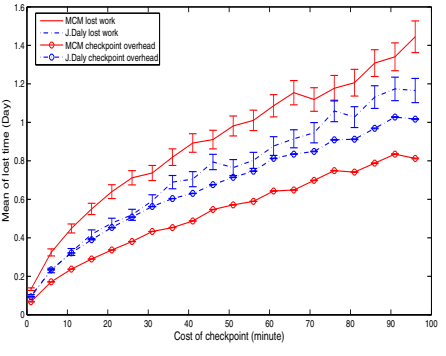

Fig. 5. Average Checkpoint / Lost work overhead (Scenario 2)

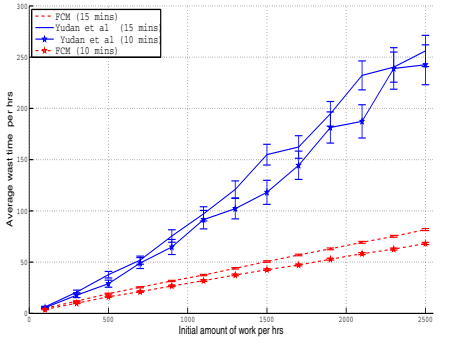

Fig. 6. Variation of the average wast time (Scenario 3)

used by authors to predict the time between the last checkpoint and the next failures. Consequently, FCM reduces the average completion time by reducing the total wast time and it outperforms the model in [6] in terms of stability by reducing the confident intervals.

6 Conclusions

We presented in this paper a new flexible mathematical model for checkpointing applications in large-scale computing platforms. The main result was to establish a general analytical model which expresses the expected completion time of the application in distributed systems. We showed that it is possible to use this model with various probability laws that modelize the failures process, e.g, the Poisson's process or the Weibull's law. Moreover, this model is derived, for determining the optimal interval between checkpoints considering the checkpoint cost and

the wast time. A performance analysis with others existing models revealed that our model reduces the checkpoint rate, the strain on I/O bandwidth and the total wast time in terms of expectation and variance.

We are currently working on implementing this mechanism in a real system. For this purpose, it would be interest to take into account a variable number of processors (instead of unbounded number of processors) since, it is not reasonable to assume that enough number of spare processors are always available.

References

1. Adiga, N., et al.: An Overview of the BlueGene/L Supercomputer. In: ACM/IEEE 2002 Conference on Supercomputing, p. 60 (2002)
2. Schroeder, B., Gibson, G.A.: A large-scale study of failures in high-performance computing systems. In: DSN 2006: Proceedings of the International Conference on Dependable Systems and Networks, Washington, DC, USA, pp. 249–258 (2006)
3. Hacker, T.J., Romero, F., Carothers, C.D.: An analysis of clustered failures on large supercomputing systems. J. Parallel Distrib. Comput. 69(7), 652–665 (2009)
4. Daly, J.T.: A higher order estimate of the optimum checkpoint interval for restart dumps. Future Generation Computer Systems 22(3), 303–312 (2006)
5. Elnozahy, E.N., Plank, J.S.: Checkpointing for peta-scale systems: A look into the future of practical rollback-recovery. IEEE Trans. Dependable Secur. Comput. 1(2), 97–108 (2004)
6. Liu, Y., Nassar, R., Leangsuksun, C., Naksinehaboon, N., Paun, M., Scott, S.: An optimal checkpoint/restart model for a large scale high performance computing system. In: IEEE International Symposium on Parallel and Distributed Processing, pp. 1–9 (2008)
7. Oliner, A.J., Rudolph, L., Sahoo, R.K.: Cooperative checkpointing: a robust approach to large-scale systems reliability. In: Proceedings of The 20th Annual International Conference on Supercomputing, pp. 14–23. ACM, New York (2006)
8. Young, J.W.: A first order approximation to the optimum checkpoint interval. ACM Commun. 17(9), 530–531 (1974)
9. Chandy, K.M., Lamport, L.: Distributed snapshots: determining global states of distributed systems. ACM Trans. Comput. Syst. 3(1), 63–75 (1985)
10. Bouguerra, M.S., Gautier, T., Trystram, D., Vincent, J.M.: A new flexible checkpoint/restart model. Technical report, RR-6751, INRIA (2008)
11. Geist, R., Reynolds, R., Westall, J.: Selection of a checkpoint interval in a critical-task environment. IEEE Transactions on Reliability 37, 395–400 (1988)
12. Plank, J.S., Thomason, M.G.: The average availability of parallel checkpointing systems and its importance in selecting runtime parameters. In: 29th International Symposium on Fault-Tolerant Computing, pp. 250–259 (1999)
13. Naksinehaboon, N., Liu, Y., Leangsuksun, C., Nassar, R., Paun, M., Scott, S.: Reliability-Aware Approach: An Incremental Checkpoint/Restart Model in HPC Environments. In: IEEE International Symposium on Cluster Computing and the Grid, pp. 783–788 (2008)
14. Tijms, H.C.: A First Course in Stochastic Models. John Wiley, Chichester (2003)