Towards Optimal Multi-Level Checkpointing

Anne Benoit, Aurélien Cavelan, Valentin Le Fèvre, Yves Robert, and Hongyang Sun

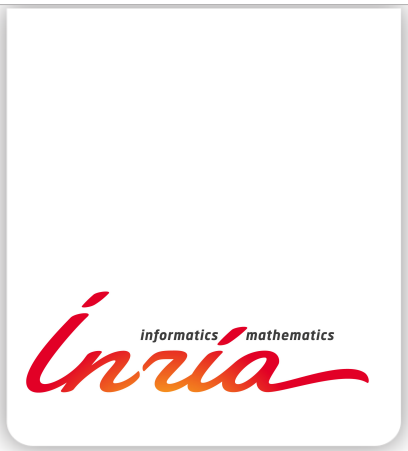
RESEARCH REPORT

N° 8930

June 2016

Project-Team ROMA

ISSN 0249-6399 ISRN INRIA/RR--8930--FR+ENG



Towards Optimal Multi-Level Checkpointing

Anne Benoit*†, Aurélien Cavelan*†, Valentin Le Fèvre*†, Yves Robert*†‡, and Hongyang Sun*†

Project-Team ROMA

Research Report n° 8930 — June 2016 — 23 pages

Abstract: We provide a framework to analyze multi-level checkpointing protocols, by formally defining a k-level checkpointing pattern. We provide a first-order approximation to the optimal checkpointing pattern, and show that the corresponding overhead is of the order of $\sum_{\ell=1}^{k} \sqrt{2\lambda_\ell C_\ell}$, where λ_ℓ is the error rate at level ℓ, and C_ℓ the checkpointing cost at level ℓ. This nicely extends the classical Young/Daly formula. Furthermore, we are able to fully characterize the shape of the optimal pattern (number and positions of checkpoints), and we provide a dynamic programming algorithm to determine which levels should be used. Finally, we perform simulations to check the accuracy of the theoretical study and to confirm the optimality of the subset of levels returned by the dynamic programming algorithm. The results nicely corroborate the theoretical study, and demonstrate the usefulness of multi-level checkpointing with the optimal subset of levels.

Key-words: resilience, fail-stop errors, multi-level checkpointing, optimal pattern.

* École Normale Supérieure de Lyon
† INRIA, France
‡ University of Tennessee Knoxville, USA

RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée
655 avenue de l'Europe Montbonnot
38334 Saint Ismier Cedex

Towards Optimal Multi-Level Checkpointing

Résumé : Ce travail analyse les techniques de checkpoint multi-niveaux. On étudie les schémas de calcul périodiques, où les différents niveaux de checkpoint sont imbriqués, et on caractérise le schéma optimal, i.e., celui dont le surcoût par unité de calcul est minimal. On montre que ce surcoût minimal est de l'ordre de $\sum_{\ell=1}^{k} \sqrt{2\lambda_\ell C_\ell}$, où λ_ℓ est le taux d'erreur au niveau ℓ, et C_ℓ le coût de checkpoint au niveau ℓ. Cette formule étend la célèbre formule de Young/Daly pour un seul niveau. On propose également un algorithme de programmation dynamique pour déterminer le meilleur sous-ensemble de niveuax à utiliser pour minimiser le surcoût global. Enfin, nous conduisons des simulations pour vérifier l'étude théorique, et confirmer l'optimalité du sous-ensemble déterminé par l'algorithme de programmation dynamique. Les résultats corroborent bien l'étude théorique, et montrent toute l'utilité d'une approche multi-niveaux basée sur le sous-ensemble de niveaux optimal.

Mots-clés : résilience, erreurs fatales, checkpoint multi-niveaux, schéma optimal.

1 Introduction

Checkpointing is the de-facto standard resilience method for HPC platforms at extreme-scale. However, the traditional single-level checkpointing method suffers from significant overhead, and multi-level checkpointing protocols now represent the state-of-the-art technique. These protocols allow for different levels of checkpoints to be set, each with a different checkpoint overhead and recovery ability. Typically, each level corresponds to a specific fault[1] type, and is associated to a storage device that is resilient to that type. For instance, a two-level system would deal with (i) transient memory errors (level 1) by storing key data in main memory; and (ii) node failures (level 2) by storing key data in stable storage (remote redundant disks).

We consider a very general scenario, where the platform is subject to k levels of faults, numbered from 1 to k. Level ℓ is associated with an error rate λ_ℓ, a checkpointing cost C_ℓ, and a recovery cost R_ℓ. A fault at level ℓ destroys all the checkpoints of lower levels (from 1 to $\ell - 1$ included) and implies a roll-back to a checkpoint of level ℓ or higher. Similarly, a recovery of level ℓ will restore data from all lower levels. Typically, fault rates are decreasing and checkpoint/recovery costs are increasing when we go to higher levels: $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k$, $C_1 \leq C_2 \leq \cdots \leq C_k$, and $R_1 \leq R_2 \leq \cdots \leq R_k$.

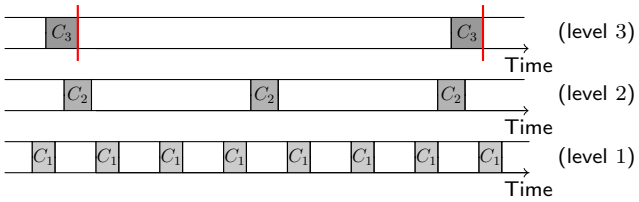


Figure 1: Independent checkpointing periods for three levels of faults: no synchronization between checkpoint levels.

The idea of multi-level checkpointing is that checkpoints are taken for each level of faults, but at different periods. Intuitively, the less frequent the faults, the longer the checkpointing period: this is because the risk of a failure striking is lower when going to higher levels; hence the expected re-execution time is lower too; one can safely checkpoint less frequently, thereby reducing failure-free overhead (checkpointing is useless in the absence of a fault). There are several natural approaches to implement multi-level checkpointing. The first option is to use independent checkpointing periods for each level, as illustrated in Figure 1 with $k = 3$ levels. This option raises several difficulties, the most prominent one being overlapping checkpoints. Typically, we need to checkpoint different levels in sequence (e.g., writing into memory before writing onto disk), so we would need to delay some checkpoints, which might be not possible in some environments, and which would introduce irregular periods. The second option is to synchronize all checkpoint levels by nesting them inside a periodic pattern that repeats over time, as illustrated in Figure 2(c). In this figure, the pattern has five computational segments, each followed by a level 1 checkpoint. The second and fifth level-1 checkpoints are followed by a level-2 checkpoint. Finally, the pattern ends with a level-3 checkpoint. When using patterns, a checkpoint at level ℓ is always preceded by checkpoints at all lower levels 1 to $\ell-1$, which makes good sense in practice (e.g., with two levels, main memory and disk, one writes the data into memory before transferring it to disk). In this context, the checkpointing

[1] We use the terms *fault*, *failure* and *error* indifferently.

cost C_ℓ at level ℓ is the additional cost paid to save data when going from level $\ell - 1$ to ℓ.

Using patterns simplifies the orchestration of checkpoints at all levels. In addition, repeatedly using the same pattern is optimal for on-line scheduling problems, or for jobs running a very long (even infinite) time on the platform. Indeed, in this scenario, we seek the best pattern, namely, the one whose overhead is minimal. The *overhead* of a pattern is the price to pay per work unit for resilience in the pattern; hence minimizing the overhead is equivalent to optimizing the throughput of the platform. For a pattern $\mathrm{P}(W)$ with W units of work (the cumulated length of all its segments), the overhead $H(\mathrm{P(W)})$ is defined as the ratio of the pattern's expected execution time $\mathbb{E}(\mathrm{P(W)})$ over its total work W minus 1:

$$H(\mathrm{P}(W)) = \frac{\mathbb{E}(\mathrm{P}(W))}{W} - 1. \tag{1}$$

If there were neither checkpoint nor fault, the overhead would be zero. Determining the optimal pattern (with minimal overhead), and then repeatedly using it until job completion, is the optimal approach with exponential failure distributions and long-lasting jobs. Indeed, once a pattern is successfully executed, the optimal strategy is to re-execute the same pattern. This is because of the memoryless property of exponential distributions: the history of failures has no impact on the solution, so if a pattern is optimal at some point in time, it stays optimal later in the execution, because we have no further information about the amount of work still to be executed.

The difficulty of characterizing the optimal pattern dramatically increases with the number of levels. How many checkpoints of each level should be used, and at which locations inside the pattern? What is the optimal length of each segment?

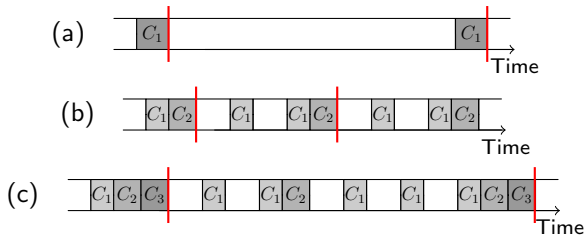


Figure 2: Computational patterns (highlighted using red bars) with $k = 1$, 2 and 3 levels.

With one single level (see Figure 2(a)), there is a single segment of length W, and the Young/Daly formula [16, 6] gives $W^{\mathrm{opt}} = \sqrt{\frac{2C_1}{\lambda_1}}$. The minimal overhead is then $H^{\mathrm{opt}} = \sqrt{2\lambda_1 C_1} + o(\sqrt{\lambda_1})$ [4].

With two levels, the pattern still has a simple shape, with N segments followed by a level-1 checkpoints, and ended by a level-2 checkpoint (see Figure 2(b)). Recent work [8] shows that all segments have same length in the optimal pattern, and provides mathematical equations that can be solved numerically to compute both the optimal length W^{opt} of the pattern and its optimal number of chunks. However, no closed-form expression is available, neither for W^{opt}, nor for the minimal overhead H^{opt}.

With three levels, no optimal solution is known. The pattern shape becomes quite complicated. Coming back to Figure 2(c), we identify two sub-patterns ending with a level-2 checkpoint. The first sub-pattern has 2 segments while the second one has 3. The memoryless property does not imply that all sub-patterns are identical, because the state after

completing the first sub-pattern is not the same as the initial state when beginning the execution of the pattern. In the general case with k levels, the shape of the pattern will be even more complicated, with different-shaped sub-patterns (each ended by a level $k-1$ checkpoint). In turn, each sub-pattern may have different-shaped sub-sub-patterns (each ended by a level $k-2$ checkpoint), and so on. The major contribution of this work is to provide an analytical characterization of the optimal pattern with an arbitrary number k of checkpoint levels, with closed-form formulas for the pattern length W^{opt}, the number of checkpoints at each level, and the optimal overhead H^{opt}. In particular, we obtain the following beautiful result:

$$H^{\text{opt}} = \sum_{\ell=1}^{k} \sqrt{2\lambda_\ell C_\ell} + \Theta(\Lambda), \tag{2}$$

where $\Lambda = \sum_{\ell=1}^{k} \lambda_\ell$. However, we point out that this analytical characterization relies on a first-order approximation, so it is valid only when resilience parameters C_ℓ and R_ℓ are small in front of the platform MTBF $\mu = 1/\Lambda$. Also, the optimal pattern has rational number of chunks, and we use rounding to derive a practical solution. Still, Equation (2) provides a lower bound of the optimal overhead, and this bound is met very closely in all our experimental scenarios.

Finally, in many practical cases, there is no obligation to use all available checkpoint levels. For instance with $k = 3$ levels, one may choose among four possibilities: level 3 only, levels 1 and 3, levels 2 and 3, and all levels 1, 2 and 3. Of course, we still have to account for all failure types, which translates into the following:

- level 3: use $\lambda_3 \leftarrow \lambda_1 + \lambda_2 + \lambda_3$;
- levels 1 and 3: use λ_1 and $\lambda_3 \leftarrow \lambda_2 + \lambda_3$;
- levels 2 and 3: use $\lambda_2 \leftarrow \lambda_1 + \lambda_2$ and λ_3;
- all levels: use λ_1, λ_2 and λ_3.

Our analytical characterization of the optimal pattern leads to a simple dynamic programming algorithm to select the optimal subset of levels.

The rest of the paper is organized as follows. Section 2 is the heart of the paper and shows how to compute the best pattern. To help the reader follow the derivation, we first show how to compute the optimal pattern with $k = 2$ levels in Section 2.1, before proceeding to the general case in Section 2.2. The algorithm to compute the optimal subset of levels is described in Section 2.3. Section 3 is devoted to simulations assessing the accuracy of the first-order approximation. We survey related work in Section 4. Finally, we give concluding remarks and hints for future work in Section 5.

2 Computing the optimal pattern

In the analysis, we make use of the following observation when dealing with the interplay of errors from different levels (recall that $\Lambda = \lambda_1 + \lambda_2 + \cdots + \lambda_k$):

Observation 1. *During the execution of a segment with length w, let X_ℓ denote the time when the first level ℓ error strikes. Thus, X_ℓ is a random variable following an Exponential distribution with parameter λ_ℓ, for all $\ell = 1, 2, \ldots, k$.*

(1) Let X denote the time when the first error (of any level) strikes. We have $X = \min\{X_1, X_2, \ldots, X_k\}$, which follows an Exponential distribution with parameter $\Lambda =$

$\sum_{\ell=1}^{k} \lambda_\ell$. *The probability of having an error (from any level) in the segment is therefore* $1 - e^{-\Lambda w}$.

(2) Given that an error (from any level) strikes during the execution of the segment, the probability that the error belongs to a particular level is proportional to the error rate of that level, i.e., $P(X = X_\ell | X \leq w) = \frac{\lambda_\ell}{\Lambda}$, *for all* $\ell = 1, 2, \ldots, k$.

Throughout the paper, we assume that errors only strike the computations, while checkpointing and recovery are error-free. It has been shown in [3] that removing this assumption does not change the first-order approximation of the pattern overhead.

We start with the particular case with $k = 2$ levels in Section 2.1, before proceeding to the general case in Section 2.2. Finally, the algorithm to compute the optimal subset of levels is described in Section 2.3.

2.1 Optimal two-level pattern

We start by analyzing the two-level pattern as shown in Figure 2(b). The goal is to determine a first-order approximation to the optimal pattern length W, the number n of level-1 checkpoints in the pattern, as well as the length $w_i = \alpha_i W$ of the i-th segment, for all $1 \leq i \leq n$, where $\sum_{i=1}^{n} \alpha_i = 1$. The following proposition shows the expected time to execute a two-level pattern when these parameters are fixed (recall that $\Lambda = \lambda_1 + \lambda_2$):

Proposition 1. *The expected execution time of a given two-level pattern is*

$$\mathbb{E}(P) = W + nC_1 + C_2 + \frac{1}{2}\left(\lambda_1 \sum_{i=1}^{n} \alpha_i^2 + \lambda_2\right) W^2 + O(\Lambda^2 W^3).$$

Proof. We first prove the following result (by induction) on the expected time $\mathbb{E}_i$ to execute the i-th segment of the pattern (without counting the checkpointing cost at the end of the segment):

$$\mathbb{E}_i = w_i + \frac{\lambda_1}{2} w_i^2 + \lambda_2 \left(\frac{w_i^2}{2} + \sum_{k=1}^{i-1} w_k w_i \right) + O(\Lambda^2 W^3). \tag{3}$$

For ease of analysis, we assume that there is a hypothetical segment at the beginning of the pattern with length $w_0 = 0$ (hence no need to checkpoint). For this segment, we have $\mathbb{E}_0 = w_0 = 0$, satisfying Equation (3). Suppose the claim holds up to $\mathbb{E}_{i-1}$. Then, $\mathbb{E}_i$ can be computed as follows:

$$\begin{aligned} \mathbb{E}_i = p_i \Bigg(& \mathbb{E}^{\text{lost}}(w_i, \lambda_1 + \lambda_2) + \frac{\lambda_1}{\lambda_1 + \lambda_2} (R_1 + \mathbb{E}_i) \\ & + \frac{\lambda_2}{\lambda_1 + \lambda_2} \Big(R_2 + R_1 + \sum_{j=1}^{i-1} \mathbb{E}_j + \mathbb{E}_i \Big) \Bigg) + (1 - p_i) w_i, \end{aligned} \tag{4}$$

where $p_i = 1 - e^{-(\lambda_1 + \lambda_2) w_i}$ denotes the probability of having a failure (either level-1 or level-2) during the execution of the segment, and $\mathbb{E}^{\text{lost}}(w_i, \lambda_1 + \lambda_2)$ denotes the expected time lost when such a failure occurs. In this case, if the failure belongs to level 1, which happens with probability $\frac{\lambda_1}{\lambda_1 + \lambda_2}$, we can recover from the latest level-1 checkpoint (R_1) and re-execute

only segment i. Otherwise, if the failure belongs to level 2, with probability $\frac{\lambda_2}{\lambda_1+\lambda_2}$, we need to recover from the last level-2 checkpoint (R_2) before restoring the corresponding level-1 checkpoint (R_1), and then re-execute all the segments (including segment i) that have been executed so far. Finally, if no error (of any level) strikes during the execution, which happens with probability $1-p_i$, the computation is done.

From [11, Equation (1.13)], we can get the following general formula on the expected time lost when executing a segment of length w with error rate λ:

$$\mathbb{E}^{\text{lost}}(w,\lambda) = \frac{1}{\lambda} - \frac{w}{e^{\lambda w}-1}. \tag{5}$$

Substituting Equation (5) into Equation (4), we get

$$\begin{aligned}
\mathbb{E}_i &= w_i + \frac{1}{2}\left(\lambda_1 w_i^2 + \lambda_2 w_i^2 + 2\lambda_2 w_i \sum_{k=1}^{i-1} \mathbb{E}_k\right) + O(\Lambda^2 W^3) \\
&= w_i + \frac{1}{2}\left(\lambda_1 w_i^2 + \lambda_2 w_i^2 + 2\lambda_2 w_i \sum_{k=1}^{i-1} \big(w_k + O(1)\big)\right) + O(\Lambda^2 W^3) \\
&= w_i + \frac{1}{2}\left(\lambda_1 w_i^2 + \lambda_2 \Big(w_i^2 + 2\sum_{k=1}^{i-1} w_i w_k\Big)\right) + O(\Lambda^2 W^3).
\end{aligned} \tag{6}$$

Since checkpointing is assumed to be error-free, we can compute the expected execution time of the pattern as follows:

$$\begin{aligned}
\mathbb{E} &= \sum_{i=1}^{n} \mathbb{E}_i + nC_1 + C_2 \\
&= \sum_{i=1}^{n} w_i + nC_1 + C_2 \\
&\quad + \frac{1}{2}\left(\lambda_1 \sum_{i=1}^{n} w_i^2 + \lambda_2 \Big(\sum_{i=1}^{n} w_i^2 + 2\sum_{i=1}^{n}\sum_{k=1}^{i-1} w_i w_k\Big)\right) + O(\Lambda^2 W^3) \\
&= W + nC_1 + C_2 + \frac{1}{2}\left(\lambda_1 \sum_{i=1}^{n} \alpha_i^2 + \lambda_2\right) W^2 + O(\Lambda^2 W^3).
\end{aligned}$$

The last equation is because $\sum_{i=1}^{n} w_i^2 + 2\sum_{i=1}^{n}\sum_{k=1}^{i-1} w_i w_k = \left(\sum_{i=1}^{n} w_i\right)^2 = W^2$. □

Theorem 1. *A first-order approximation to the optimal two-level pattern is characterized by*

$$n^{opt} = \sqrt{\frac{\lambda_1}{\lambda_2} \cdot \frac{C_2}{C_1}}, \tag{7}$$

$$\alpha_i^{opt} = \frac{1}{n^{opt}} \quad \forall i = 1, 2, \ldots, n^{opt}, \tag{8}$$

$$W^{opt} = \sqrt{\frac{n^{opt} C_1 + C_2}{\frac{1}{2}\left(\frac{\lambda_1}{n^{opt}} + \lambda_2\right)}}, \tag{9}$$

where n^{opt} is the number of segments, $\alpha_i^{opt} W^{opt}$ is the length of the i-th segment, and W^{opt} is the pattern length. The optimal pattern overhead is

$$H^{opt} = \sqrt{2\lambda_1 C_1} + \sqrt{2\lambda_2 C_2} + O(\Lambda). \tag{10}$$

Proof. For a given pattern with a fixed number n of segments, $\sum_{i=1}^{n} \alpha_i^2$ is minimized subject to $\sum_{i=1}^{n} \alpha_i = 1$ when $\alpha_i = \frac{1}{n}$ for all $i = 1, 2, \ldots, n$. Hence, we can derive the expected execution overhead from Proposition 1 as follows:

$$H = \frac{nC_1 + C_2}{W} + \frac{1}{2}\left(\frac{\lambda_1}{n} + \lambda_2\right) W + O(\Lambda^2 W^2). \tag{11}$$

For a given n, the optimal work length can then be computed from Equation (11), and it is given by $W^{\text{opt}} = \sqrt{\frac{nC_1+C_2}{\frac{1}{2}\left(\frac{\lambda_1}{n}+\lambda_2\right)}} = \Theta(\Lambda^{-1/2})$. In that case, the execution overhead becomes

$$H = \sqrt{2\left(\frac{\lambda_1}{n} + \lambda_2\right)(nC_1 + C_2)} + O(\Lambda), \tag{12}$$

which is minimized as shown in Equation (10) when n satisfies Equation (7). Indeed, $2\left(\frac{\lambda_1}{n^{\text{opt}}} + \lambda_2\right)\left(n^{\text{opt}} C_1 + C_2\right) = 2\lambda_1 C_1 + 2\lambda_2 C_2 + 4\sqrt{\lambda_1 \lambda_2 C_1 C_2} = (\sqrt{2\lambda_1 C_1} + \sqrt{2\lambda_1 C_1})^2$. In practice, since the number of segments can only be a positive integer, the optimal solution is either $\max(1, \lfloor n^{\text{opt}} \rfloor)$ or $\lceil n^{\text{opt}} \rceil$, whichever leads to a smaller value of the convex function H as shown in Equation (12). □

Theorem 1 extends Young/Daly's classical formula [16, 6] to the two-level checkpointing scenario. When there is only one level, i.e., $\lambda_1 = 0$ and $C_1 = 0$, we retrieve their classical result.

2.2 Optimal k-level pattern

From the analysis of the two-level pattern, we observe that the overall execution overhead of a pattern comes from two distinct sources defined below:

Definition 1. *There are two types of execution overheads for a pattern:*

(1) Error-free overhead*, denoted as o_{ef}, is the total cost of all the checkpoints placed in the pattern. For a given set of checkpoints, the error-free overhead is completely determined regardless of their positions in the pattern.*

(2) Re-executed fraction overhead*, denoted as o_{re}, is the expected fraction of work that needs to be re-executed due to errors. The re-executed fraction overhead depends on both the set of checkpoints and their positions.*

For example, in the two-level pattern with n level-1 checkpoints and given values of α_i for all $i = 1, 2, \ldots, n$, the two types of overheads are given by $o_{\text{ef}} = nC_1 + C_2$ and $o_{\text{re}} = \frac{1}{2}\left(f_1 \sum_{i=1}^{n} \alpha_i^2 + f_2\right)$, where $f_\ell = \frac{\lambda_\ell}{\lambda_1 + \lambda_2}$ for $\ell = 1, 2$. Assuming that checkpoints at all levels have constant costs and that the error rates at all levels are in the same order, then both o_{ef} and o_{re} can be considered as constants, i.e., $o_{\text{ef}} = O(1)$ and $o_{\text{re}} = O(1)$.

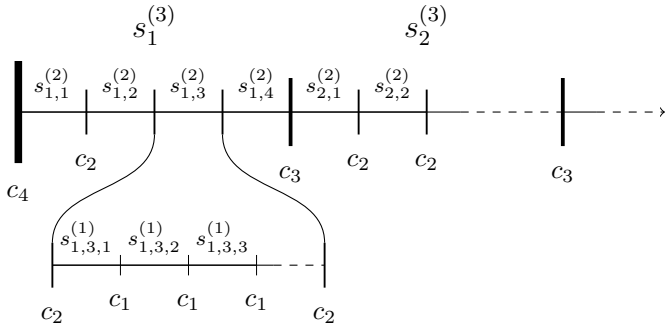


Figure 3: Pattern example with 4 levels. We let $c_i = C_1|C_2|\dots|C_i$ denote the succession of checkpoints from level 1 to level i.

A trade-off exists between these two types of execution overheads, since placing more checkpoints generally reduces the re-executed work fraction when an error strikes, but it can adversely increase the overhead when the execution is error-free. Therefore, in order to achieve the best overall overhead, a resilience algorithm must seek an optimal balance between o_{ef} and o_{re}.

For a given pattern with fixed overheads o_{ef} and o_{re}, we can make the following observation, which partially characterizes the optimal pattern.

Observation 2. *For a given pattern (with fixed o_{ef} and o_{re}), the expected execution time is given by*

$$\mathbb{E} = \underbrace{W + o_{\text{ef}}}_{\substack{\textit{error-free}\\ \textit{execution time}}} + \underbrace{\Lambda W}_{\substack{\textit{expected}\\ \textit{\# errors}}} \cdot \underbrace{o_{\text{re}} W}_{\substack{\textit{re-executed work}\\ \textit{in case of error}}} + O(\Lambda^2 W^3), \tag{13}$$

and the optimal pattern length and the resulting expected execution overhead of the pattern are

$$W^{opt} = \sqrt{\frac{o_{\text{ef}}}{\Lambda \cdot o_{\text{re}}}}, \tag{14}$$

$$H^{opt} = 2\sqrt{\Lambda \cdot o_{\text{ef}} \cdot o_{\text{re}}} + O(\Lambda). \tag{15}$$

In particular, Equation (15) shows that the trade-off between o_{ef} and o_{re} is manifested as the product of the two terms. Hence, in order to determine the optimal pattern, it suffices to find the pattern parameters (e.g., n and α_i) that minimize $o_{\text{ef}} \cdot o_{\text{re}}$. Furthermore, since both o_{ef} and o_{re} are constants, we obtain that $W^{\text{opt}} = \Theta(\Lambda^{-1/2})$and $H^{\text{opt}} = \Theta(\Lambda^{1/2})$.

We now extend the analysis to derive the optimal multi-level checkpointing patterns. Generally, for a k-level pattern, each computational chunk or segment $s^{(\ell)}_{i_{k-1},\dots,i_\ell}$ can be uniquely identified by the level ℓ it is in as well as its position $\langle i_{k-1}, \dots, i_\ell \rangle$ within the hierarchy. For instance, in a four-level pattern, the segment $s^{(2)}_{1,3}$ denotes the third level-2 segment inside the first level-3 segment of the pattern (see Figure 2.2). Note that a segment can contain multiple sub-segments at the lower levels (except for bottom-level segments) and is a sub-segment of a larger segment at a higher level (except for top-level segments). The entire pattern can be denoted as $s^{(k)}$, which is the only segment at level k.

For any segment $s^{(\ell)}_{i_{k-1},\ldots,i_\ell}$ at level ℓ, where $1 \le \ell \le k$, let $w^{(\ell)}_{i_{k-1},\ldots,i_\ell}$ denote its size or amount of work. Hence, we have $w^{(\ell+1)}_{i_{k-1},\ldots,i_{\ell+1}} = \sum_{i_\ell} w^{(\ell)}_{i_{k-1},\ldots,i_\ell}$ and $w^{(k)} = W$. Also, let $n^{(\ell)}_{i_{k-1},\ldots,i_\ell}$ denote the number of sub-segments contained by $s^{(\ell)}_{i_{k-1},\ldots,i_\ell}$ at the lower level $\ell - 1$. We have $n^{(1)}_{i_{k-1},\ldots,i_1} = 1$ for all $i_{k-1}, \ldots, i_1$. For convenience, we further define

$$\alpha^{(\ell)}_{i_{k-1},\ldots,i_\ell} = \frac{w^{(\ell)}_{i_{k-1},\ldots,i_\ell}}{W}$$

as the fraction of the size of segment $s^{(\ell)}_{i_{k-1},\ldots,i_\ell}$ inside the pattern, and define N_ℓ to be the total number of level-ℓ segments in the entire pattern. Therefore, we have $N_k = 1$, $N_{k-1} = n^{(k)}$, and in general

$$N_\ell = \sum_{i_{k-1},\ldots,i_{\ell+1}} n^{(\ell+1)}_{i_{k-1},\ldots,i_{\ell+1}}.$$

The following proposition shows the expected time to execute a given k-level pattern.

Proposition 2. *The expected execution time of a given k-level pattern is*

$$\mathbb{E} = W + \sum_{\ell=1}^{k-1} N_\ell C_\ell + C_k + \frac{W^2}{2}\left(\sum_{\ell=1}^{k} \lambda_\ell \sum_{i_{k-1},\ldots,i_\ell} \left(\alpha^{(\ell)}_{i_{k-1},\ldots,i_\ell}\right)^2\right) + O(\Lambda^2 W^3).$$

Proof. We show that the expected time to execute any segment $s^{(h)}_{i_{k-1},\ldots,i_h}$ at level h, where $1 \le h \le k$, satisfies the following:

$$\begin{aligned}\mathbb{E}^{(h)}_{i_{k-1},\ldots,i_h} = w^{(h)}_{i_{k-1},\ldots,i_h} &+ \frac{W^2}{2}\left(\sum_{\ell=1}^{h} \lambda_\ell \sum_{i_{h-1},\ldots,i_\ell} \left(\alpha^{(\ell)}_{i_{k-1},\ldots,i_\ell}\right)^2\right) \\ &+ \lambda_{[h+1,k]}\left(\frac{\left(w^{(h)}_{i_{k-1},\ldots,i_h}\right)^2}{2} + w^{(h)}_{i_{k-1},\ldots,i_h} \sum_{j_h=1}^{i_h-1} E^{(h)}_{\langle i_{k-1},\ldots,j_h\rangle}\right) \\ &+ w^{(h)}_{i_{k-1},\ldots,i_h} \sum_{\ell=h+2}^{k} \lambda_{[\ell,k]} \sum_{j_{\ell-1}=1}^{i_{\ell-1}-1} \mathbb{E}^{(\ell-1)}_{\langle i_{k-1},\ldots,j_{\ell-1}\rangle} + O(\Lambda^2 W^3), \end{aligned} \tag{16}$$

where $\lambda_{[x,y]} = \sum_{\ell=x}^{y} \lambda_\ell$. Then, the proposition can be proven by setting $\mathbb{E} = \mathbb{E}^{(k)} + \sum_{\ell=1}^{k-1} N_\ell C_\ell + C_k$, since checkpoints are assumed to be error-free.

We now prove Equation (16) by induction on the level h. For the base case, i.e., when $h = 1$, consider a segment $s^{(1)}_{i_{k-1},\ldots,i_1}$ at the first level. Following the proof of Proposition 1 (in

particular, Equation (4)), we can express its expected execution time as

$$\begin{aligned}
\mathbb{E}^{(1)}_{i_{k-1},\ldots,i_1} = p^{(1)}_{i_{k-1},\ldots,i_1} \Bigg(& \mathbb{E}^{\text{lost}}\big(w^{(1)}_{i_{k-1},\ldots,i_1}, \Lambda\big) \\
& + \frac{\lambda_1}{\Lambda}\Big(R_1 + \mathbb{E}^{(1)}_{i_{k-1},\ldots,i_1}\Big) \\
& + \frac{\lambda_2}{\Lambda}\Big(\sum_{j=1}^{2} R_j + \sum_{j_1=1}^{i_1} \mathbb{E}^{(1)}_{i_{k-1},\ldots,j_1}\Big) \\
& + \frac{\lambda_3}{\Lambda}\Big(\sum_{j=1}^{3} R_j + \sum_{j_2=1}^{i_2-1} \mathbb{E}^{(2)}_{i_{k-1},\ldots,j_2} + \sum_{j_1=1}^{i_1} \mathbb{E}^{(1)}_{i_{k-1},\ldots,j_1}\Big) \\
& \qquad \vdots \\
& + \frac{\lambda_k}{\Lambda}\Big(\sum_{j=1}^{k} R_j + \sum_{j_{k-1}=1}^{i_{k-1}-1} \mathbb{E}^{(k-1)}_{j_{k-1}} + \sum_{j_{k-2}=1}^{i_{k-2}-1} \mathbb{E}^{(k-2)}_{i_{k-1},j_{k-2}} \\
& \qquad\qquad + \cdots + \sum_{j_1=1}^{i_1} \mathbb{E}^{(1)}_{i_{k-1},\ldots,j_1}\Big)\Bigg) \\
& + \big(1 - p^{(1)}_{i_{k-1},\ldots,i_1}\big) w^{(1)}_{i_{k-1},\ldots,i_1}. \qquad (17)
\end{aligned}$$

where $\Lambda = \sum_{\ell=1}^{k} \lambda_\ell$ denotes the total rate of all error sources, and $p^{(1)}_{i_{k-1},\ldots,i_1} = 1 - e^{\Lambda \cdot w^{(1)}_{i_{k-1},\ldots,i_1}}$ denote the probability of having an error (from any level) during the execution of the segment. Simplifying Equation (17) and solving for $\mathbb{E}^{(1)}_{i_{k-1},\ldots,i_1}$ we get:

$$\begin{aligned}
\mathbb{E}^{(1)}_{i_{k-1},\ldots,i_1} &= w^{(1)}_{i_{k-1},\ldots,i_1} + \frac{W^2}{2}\lambda_{[1,k]}\left(\alpha^{(1)}_{i_{k-1},\ldots,i_1}\right)^2 \\
&+ w^{(1)}_{i_{k-1},\ldots,i_1} \sum_{\ell=2}^{k} \lambda_{[\ell,k]} \sum_{j_{\ell-1}=1}^{i_{\ell-1}-1} \mathbb{E}^{(\ell-1)}_{i_{k-1},\ldots,j_{\ell-1}} + O(\sqrt{\lambda}),
\end{aligned}$$

which satisfies Equation (16).

Suppose Equation (16) holds up to any segment $s^{(h)}_{i_{k-1},\ldots,i_h}$ at level h. Following the proof of Proposition 1 (in particular, Equation (6)), we can show by induction that $\mathbb{E}^{(h)}_{i_{k-1},\ldots,i_h} =$

$w^{(h)}_{i_{k-1},\ldots,i_h} + O(1)$. Hence, for segment $s^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}$ at level $h+1$, we have:

$$
\begin{aligned}
\mathbb{E}^{(h+1)}_{i_{k-1},\ldots,i_{h+1}} &= \sum_{i_h} \mathbb{E}^{(h)}_{i_{k-1},\ldots,i_h} \\
&= \sum_{i_h} w^{(h)}_{i_{k-1},\ldots,i_h} + \frac{W^2}{2}\left(\sum_{\ell=1}^{h}\lambda_\ell \sum_{i_h,\ldots,i_\ell}\left(\alpha^{(\ell)}_{i_{k-1},\ldots,i_\ell}\right)^2\right) \\
&\quad + \lambda_{[h+1,k]}\sum_{i_h}\left(\frac{\left(w^{(h)}_{i_{k-1},\ldots,i_h}\right)^2}{2} + w^{(h)}_{i_{k-1},\ldots,i_h}\sum_{j_h=1}^{i_h-1} w^{(h)}_{i_{k-1},\ldots,j_h}\right) \\
&\quad + \sum_{i_h} w^{(h)}_{i_{k-1},\ldots,i_h}\sum_{\ell=h+2}^{k}\lambda_{[\ell,k]}\sum_{j_{\ell-1}=1}^{i_{\ell-1}-1}\mathbb{E}^{(\ell-1)}_{i_{k-1},\ldots,j_{\ell-1}} + O(\sqrt{\lambda}) \\
&= w^{(h+1)}_{i_{k-1},\ldots,i_{h+1}} + \frac{W^2}{2}\left(\sum_{\ell=1}^{h}\lambda_\ell \sum_{i_h,\ldots,i_\ell}\left(\alpha^{(\ell)}_{i_{k-1},\ldots,i_\ell}\right)^2\right) \\
&\quad + \lambda_{[h+1,k]}\frac{\left(w^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)^2}{2} \\
&\quad + w^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\sum_{\ell=h+2}^{k}\lambda_{[\ell,k]}\sum_{j_{\ell-1}=1}^{i_{\ell-1}-1}\mathbb{E}^{(\ell-1)}_{i_{k-1},\ldots,j_{\ell-1}} + O(\Lambda^2 W^3) \\
&= w^{(h+1)}_{i_{k-1},\ldots,i_{h+1}} + \frac{W^2}{2}\left(\sum_{\ell=1}^{h+1}\lambda_\ell \sum_{i_h,\ldots,i_\ell}\left(\alpha^{(\ell)}_{i_{k-1},\ldots,i_\ell}\right)^2\right) \\
&\quad + \lambda_{[h+2,k]}\left(\frac{\left(w^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)^2}{2} + w^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\sum_{j_{h+1}=1}^{i_{h+1}-1}\mathbb{E}^{(h+1)}_{i_{k-1},\ldots,j_{h+1}}\right) \\
&\quad + w^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\sum_{\ell=h+3}^{k}\lambda_{[\ell,k]}\sum_{j_{\ell-1}=1}^{i_{\ell-1}-1}\mathbb{E}^{(\ell-1)}_{i_{k-1},\ldots,j_{\ell-1}} + O(\Lambda^2 W^3).
\end{aligned}
$$

Hence, Equation (16) also holds for any segment at level $h+1$. This completes the proof of the proposition. □

Proposition 2 shows that, for a given k-level checkpointing pattern, the error-free overhead o_{ef} and the re-executed fraction overhead o_{re} are given as follows:

$$o_{\text{ef}} = \sum_{\ell=1}^{k-1} N_\ell C_\ell + C_k, \tag{18}$$

$$o_{\text{re}} = \frac{1}{2}\sum_{\ell=1}^{k} f_\ell \sum_{i_{k-1},\ldots,i_\ell}\left(\alpha^{(\ell)}_{i_{k-1},\ldots,i_\ell}\right)^2, \tag{19}$$

where $f_\ell = \frac{\lambda_\ell}{\Lambda}$. According to Observation 2, it remains to find parameters of the pattern such that $o_{\text{ef}} \cdot o_{\text{re}}$ is minimized.

To derive the optimal pattern, we first consider the case where o_{ef} is fixed, that is, the set of checkpoints is given. The following proposition shows the optimal value of o_{re}.

Proposition 3. *For a k-level checkpointing pattern, suppose the number N_ℓ of checkpoints at each level ℓ is given, i.e., the error-free overhead o_{ef} is fixed (as in Equation (18)). Then, the optimal value of the re-executed work overhead is given by*

$$o_{\text{re}}^{opt} = \frac{1}{2}\left(\sum_{\ell=1}^{k-1}\frac{f_\ell}{N_\ell} + f_k\right), \tag{20}$$

and it is obtained when all the checkpoints of each level are equally spaced in the pattern.

Proof. According to Equation (19), which shows the value of o_{re} for the entire pattern, we can define the corresponding overhead for each level-h segment $s^{(h)}_{i_{k-1},\ldots,i_h}$ recursively as follows:

$$o_{\text{re}}\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right) = \frac{f_h}{2}\cdot\left(\alpha^{(h)}_{i_{k-1},\ldots,i_h}\right)^2 + \sum_{i_{h-1}} o_{\text{re}}\left(s^{(h-1)}_{i_{k-1},\ldots,i_{h-1}}\right),$$

with $o_{\text{re}}\left(s^{(0)}_{i_{k-1},\ldots,i_0}\right) = 0$ by definition.

For each segment $s^{(h)}_{i_{k-1},\ldots,i_h}$, we also define $N_\ell\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right)$ to be the total number of level-ℓ segments it contains, with $\ell \le h$. We will show that the optimal value $o^{\text{opt}}_{\text{re}}\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right)$ for the segment satisfies:

$$o^{\text{opt}}_{\text{re}}\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right) = \frac{1}{2}\left(\sum_{\ell=1}^{h}\frac{f_\ell}{N_\ell\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right)}\right)\left(\alpha^{(h)}_{i_{k-1},\ldots,i_h}\right)^2, \tag{21}$$

and it is achieved when its level-ℓ checkpoints are equally spaced, for all $\ell \le h-1$. The proposition can then be proven by setting $o^{\text{opt}}_{\text{re}} = o^{\text{opt}}_{\text{re}}\left(s^{(k)}\right)$, since $N_\ell\left(s^{(k)}\right) = N_\ell$, $N_k = 1$, and $\alpha^{(k)} = 1$.

Now, we prove Equation (21) by induction on the level h. For the base case, i.e., when $h = 1$, we have $o_{\text{re}}\left(s^{(1)}_{i_{k-1},\ldots,i_1}\right) = \frac{f_1}{2}\cdot\left(\alpha^{(1)}_{i_{k-1},\ldots,i_1}\right)^2$ by definition, and it satisfies Equation (21), because $N_1\left(s^{(1)}_{i_{k-1},\ldots,i_1}\right) = 1$. Suppose Equation (21) holds for any segment $s^{(h)}_{i_{k-1},\ldots,i_h}$ at level h. Then, for segment $s^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}$ at level $h+1$, we have:

$$\begin{aligned} o_{\text{re}}\left(s^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right) &= \frac{f_{h+1}}{2}\cdot\left(\alpha^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)^2 + \sum_{i_h} o^{\text{opt}}_{\text{re}}\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right) \\ &= \frac{f_{h+1}}{2}\cdot\left(\alpha^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)^2 + \frac{1}{2}y, \end{aligned} \tag{22}$$

where $y = \sum_{i_h} x^{(h)}_{i_{k-1},\ldots,i_h}\cdot\left(\alpha^{(h)}_{i_{k-1},\ldots,i_h}\right)^2$, and $x^{(h)}_{i_{k-1},\ldots,i_h} = \sum_{\ell=1}^{h}\frac{f_\ell}{N_\ell\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right)}$. To minimize $o_{\text{re}}\left(s^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)$ as shown in Equation (22), it suffices to solve the following minimization

problem:

$$\begin{aligned} \text{minimize} \quad & y = \sum_{i_h} x^{(h)}_{i_{k-1},\ldots,i_h} \cdot \left(\alpha^{(h)}_{i_{k-1},\ldots,i_h}\right)^2 \\ \text{subject to} \quad & \sum_{i_h} \alpha^{(h)}_{i_{k-1},\ldots,i_h} = \alpha^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}. \end{aligned}$$

Since y is clearly a convex function of $\alpha^{(h)}_{i_{k-1},\ldots,i_h}$, we can readily get, using Lagrange multiplier, the minimum value of y as follows:

$$y_{\min} = \frac{1}{\sum_{i_h} 1/x^{(h)}_{i_{k-1},\ldots,i_h}} \cdot \left(\alpha^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)^2, \tag{23}$$

which is obtained at

$$\widetilde{\alpha}^{(h)}_{i_{k-1},\ldots,i_h} = \frac{1/x^{(h)}_{i_{k-1},\ldots,i_h}}{\sum_{j_h} 1/x^{(h)}_{i_{k-1},\ldots,j_h}} \cdot \alpha^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}. \tag{24}$$

Let us define $z = \sum_{i_h} 1/x^{(h)}_{i_{k-1},\ldots,i_h}$. We now need to solve the following maximization problem:

$$\begin{aligned} \text{maximize} \quad & z = \sum_{i_h} \frac{1}{\sum_{\ell=1}^{h} \frac{f_\ell}{N_\ell\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right)}} \\ \text{subject to} \quad & \sum_{i_h} N_\ell\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right) = N_\ell\left(s^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right) \\ & \qquad\qquad \text{for all} \quad \ell = 1,\ldots,h. \end{aligned}$$

Again, z is a convex function of $N_\ell\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right)$, and it can be shown to be maximized when

$$N_\ell\left(s^{(h)}_{i_{k-1},\ldots,i_h}\right) = \frac{N_\ell\left(s^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)}{n^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}}, \quad \forall \ell = 1,\ldots,h,$$

which gives $\widetilde{\alpha}^{(h)}_{i_{k-1},\ldots,i_h} = \frac{1}{n^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}} \alpha^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}$ according to Equation (24). This implies that all level-ℓ checkpoints are also equally spaced inside segment $s^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}$, for all $\ell \leq h$. The maximum value of z in this case is

$$z_{\max} = \frac{1}{\sum_{\ell=1}^{h} \frac{f_\ell}{N_\ell\left(s^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)}},$$

and the optimal value of $y_{\min}$ according to Equation (23) is then given by

$$\begin{aligned} y^{\text{opt}}_{\min} &= \frac{1}{z_{\max}} \left(\alpha^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)^2 \\ &= \left(\sum_{\ell=1}^{h} \frac{f_\ell}{N_\ell\left(s^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)}\right) \left(\alpha^{(h+1)}_{i_{k-1},\ldots,i_{h+1}}\right)^2. \end{aligned}$$

Substituting $y_{\min}^{\text{opt}}$ into Equation (22), we get the optimal value of $o_{\text{re}}\left(s_{i_{k-1},\ldots,i_{h+1}}^{(h+1)}\right)$ as follows:

$$\begin{aligned} o_{\text{re}}^{\text{opt}}\left(s_{i_{k-1},\ldots,i_{h+1}}^{(h+1)}\right) &= \frac{f_{h+1}}{2}\cdot\left(\alpha_{i_{k-1},\ldots,i_{h+1}}^{(h+1)}\right)^2+\frac{1}{2}y_{\min}^{\text{opt}} \\ &= \frac{1}{2}\left(\sum_{\ell=1}^{h+1}\frac{f_\ell}{N_\ell\left(s_{i_{k-1},\ldots,i_h}^{(h+1)}\right)}\right)\left(\alpha_{i_{k-1},\ldots,i_{h+1}}^{(h+1)}\right)^2. \end{aligned}$$

This shows that Equation (21) also holds for segment $s_{i_{k-1},\ldots,i_{h+1}}^{(h+1)}$ at level $h+1$ and, hence, completes the proof of the proposition. □

We are now ready to characterize the optimal k-level pattern. The result is stated in the following theorem.

Theorem 2. *A first-order approximation to the optimal k-level checkpointing pattern is characterized by*

$$W^{opt} = \sqrt{\frac{2\left(\sum_{\ell=1}^{k-1} N_\ell^{opt} C_\ell + C_k\right)}{\sum_{\ell=1}^{k-1}\frac{\lambda_\ell}{N_\ell^{opt}}+\lambda_k}}, \tag{25}$$

$$N_\ell^{opt} = \sqrt{\frac{\lambda_\ell}{C_\ell}\cdot\frac{C_k}{\lambda_k}}, \qquad \forall \ell = 1,2,\ldots,k-1. \tag{26}$$

The optimal pattern overhead is given by

$$H^{opt} = \sum_{\ell=1}^{k}\sqrt{2\lambda_\ell C_\ell} + O(\Lambda). \tag{27}$$

Proof. From Observation 2, Equation (18) and Proposition 3, we know that the optimal pattern can be obtained by minimizing the following function:

$$F = o_{\text{ef}}\cdot o_{\text{re}}^{\text{opt}} = \frac{1}{2}\left(\sum_{\ell=1}^{k-1} N_\ell C_\ell + C_k\right)\left(\sum_{\ell=1}^{k-1}\frac{f_\ell}{N_\ell}+f_k\right). \tag{28}$$

We first compute the optimal number of checkpoints at each level using a *two-phase iterative* method. Towards this end, let us define

$$\begin{aligned} o_{\text{ef}}(h) &= \sum_{\ell=h}^{k-1} N_\ell C_\ell + C_k, \\ o_{\text{re}}^{\text{opt}}(h) &= \frac{1}{2}\left(\sum_{\ell=h}^{k-1}\frac{f_\ell}{N_\ell}+f_k\right). \end{aligned}$$

In the first phase, we set initially $F(1) = o_{\text{ef}}(1)\cdot o_{\text{re}}^{\text{opt}}(1)$ as shown in Equation (28). The

optimal value of N_1 that minimizes $F(1)$ can then be obtained by setting

$$\begin{aligned}\frac{\partial F(1)}{\partial N_1} &= C_1 o_{\text{re}}^{\text{opt}}(1) - o_{\text{ef}}(1)\frac{f_1}{2N_1^2} \\ &= C_1\left(\frac{f_1}{2N_1} + o_{\text{re}}^{\text{opt}}(2)\right) - (N_1C_1 + o_{\text{ef}}(2))\,\frac{f_1}{2N_1^2} \\ &= C_1 o_{\text{re}}^{\text{opt}}(2) - o_{\text{ef}}(2)\frac{f_1}{2N_1^2} = 0,\end{aligned}$$

which gives $N_1^{\text{opt}} = \sqrt{\frac{f_1}{C_1}\cdot\frac{o_{\text{ef}}(2)}{2o_{\text{re}}^{\text{opt}}(2)}}$. Substituting it into $F(1)$ and simplifying, we can get the value of F after the first iteration as

$$F(2) = \frac{1}{2}\left(\sqrt{f_1C_1} + \sqrt{o_{\text{ef}}(2)\cdot o_{\text{re}}^{\text{opt}}(2)}\right)^2.$$

Repeating the above process, we can get the optimal value of F after $k-1$ iterations as

$$F^{\text{opt}} = F(k) = \frac{1}{2}\left(\sum_{\ell=1}^{k}\sqrt{f_\ell C_\ell}\right)^2, \tag{29}$$

and the optimal value of N_ℓ as

$$N_\ell^{\text{opt}} = \sqrt{\frac{f_\ell}{C_\ell}\cdot\frac{o_{\text{ef}}(\ell+1)}{2o_{\text{re}}^{\text{opt}}(\ell+1)}}, \quad \forall \ell = 1, 2, \ldots, k-1. \tag{30}$$

In the second phase, we first get $N_{k-1}^{\text{opt}} = \sqrt{\frac{f_{k-1}}{C_{k-1}}\cdot\frac{C_k}{f_k}} = \sqrt{\frac{\lambda_{k-1}}{C_{k-1}}\cdot\frac{C_k}{\lambda_k}}$ from Equation (30). Substituting it into N_{k-2}^{opt} we obtain

$$\begin{aligned}N_{k-2}^{\text{opt}} &= \sqrt{\frac{f_{k-2}}{C_{k-2}}\cdot\frac{o_{\text{ef}}(k-1)}{2o_{\text{re}}^{\text{opt}}(k-1)}} \\ &= \sqrt{\frac{\lambda_{k-2}}{C_{k-2}}\cdot\frac{N_{k-1}^{\text{opt}}C_{k-1}+C_k}{\frac{\lambda_{k-1}}{N_{k-1}^{\text{opt}}}+\lambda_k}} \\ &= \sqrt{\frac{\lambda_{k-2}}{C_{k-2}}\cdot\frac{\sqrt{\frac{\lambda_{k-1}}{\lambda_k}C_{k-1}C_k}+C_k}{\sqrt{\lambda_{k-1}\lambda_k\frac{C_{k-1}}{C_k}}+\lambda_k}} \\ &= \sqrt{\frac{\lambda_{k-2}}{C_{k-2}}\cdot\frac{C_k}{\lambda_k}}.\end{aligned}$$

Repeating the above process iteratively, we can compute the optimal values of N_ℓ^{opt}, for $\ell = k-3, \ldots, 2, 1$, as given in Equation (26) by using values of $N_{k-1}^{\text{opt}}, \ldots, N_{\ell+1}^{\text{opt}}$.

The optimal pattern length, according to Equation (14), can be expressed as $W^{\text{opt}} = \sqrt{\frac{o_{\text{ef}}}{\Lambda\cdot o_{\text{re}}^{\text{opt}}}}$, which turns out to be Equation (25) with the optimal values of N_ℓ^{opt}.

The optimal overhead, according to Equations (15) and (29), can be expressed as $H^{\text{opt}} = 2\sqrt{\Lambda\cdot F^{\text{opt}}} + O(\Lambda)$, which gives rise to Equation (27). This completes the proof of the theorem. □

Corollary 1. *In a k-level checkpointing pattern, the optimal number of level-ℓ checkpoints between any two consecutive level-$(\ell+1)$ checkpoints is*

$$n_\ell^{opt} = \frac{N_\ell^{opt}}{N_{\ell+1}^{opt}} = \sqrt{\frac{\lambda_\ell}{\lambda_{\ell+1}} \cdot \frac{C_{\ell+1}}{C_\ell}}, \tag{31}$$

for all $\ell = 1, \ldots, k-1$.

2.3 Optimal subset of levels

While Theorem 2 characterizes the optimal pattern by using k levels of checkpoints, this section addresses the problem of selecting the optimal subset of levels in order to minimize the overall execution overhead.

First, we show that the optimal solution does not necessarily use all the levels available. Consider the simple example with $k = 2$ levels. Define $\alpha = \frac{\lambda_2}{\lambda_1}$ and $\beta = \frac{C_2}{C_1}$. Equation (27) suggests that the optimal solution uses both levels if and only if the following condition holds:

$$\begin{aligned} & \sqrt{2\lambda_1 C_1} + \sqrt{2\lambda_2 C_2} < \sqrt{2\left(\lambda_1 + \lambda_2\right) C_2}, \\ \Leftrightarrow\ & 4\alpha\beta < (\beta - 1)^2, \end{aligned}$$

which is not true when $\alpha = 0.5$ and $\beta = 2$. For a general k-level pattern, the optimal subset of levels to use could well depend on the relative checkpointing costs and error rates of different levels. The following theorem presents a dynamic programming algorithm to solve this problem. The solution is particularly useful when the number k of levels is large.

Theorem 3. *Suppose there are k levels of checkpoints available. Then, the optimal subset of levels to use can be obtained by dynamic programming in $O(k^2)$ time.*

Proof. Let $\mathcal{S}^{\text{opt}}(h) \subseteq \{0, 1, \ldots, h\}$ denote the optimal subset of levels used by a pattern that is capable of handling errors up to level h, and let $H^{\text{opt}}(h)$ denote the corresponding optimal overhead (ignoring lower-order terms) incurred by the pattern.

Define $\mathcal{S}^{\text{opt}}(0) = \emptyset$ and $H^{\text{opt}}(0) = 0$. Recall that $\lambda_{[x,y]} = \sum_{\ell=x}^{y} \lambda_\ell$. We can compute $H^{\text{opt}}(h)$ using the following dynamic programming formulation:

$$H^{\text{opt}}(h) = \min_{0 \le \ell \le h-1} \left\{ H^{\text{opt}}(\ell) + \sqrt{2\lambda_{[\ell+1,h]} C_h} \right\}, \tag{32}$$

and the optimal subset is $\mathcal{S}^{\text{opt}}(h) = \mathcal{S}^{\text{opt}}(\ell^{\text{opt}}) \bigcup \{h\}$, where ℓ^{opt} is the value of ℓ that yields the minimum $H^{\text{opt}}(h)$.

The optimal subset of levels to handle all k levels of errors is then given by $\mathcal{S}^{\text{opt}}(k)$ with the optimal overhead $H^{\text{opt}}(k)$. The complexity is clearly quadratic in the total number of levels. □

3 Simulations

In this section, we conduct a set of simulations whose goal is twofold: (i) to check the accuracy of the theoretical study; and (ii) to confirm the optimality of the subset of levels found by the dynamic programming algorithm. We instantiate the model with two scenarios. The first scenario uses a set of real values measured on a medium-sized HPC system at Lawrence

Set	From	Level	1	2	3	4
(A)	Moody et al. [12]	C (s)	0.5	4.5	1051	-
		MTBF (s)	5.00e6	5.56e5	2.50e6	-
(B)	Balaprakash et al. [1]	C (s)	10	20	20	100
		MTBF (s)	3.60e4	7.20e4	1.44e5	7.20e5

Table 1: Sets of parameters (A) and (B), used as input for simulations.

Livermore National Laboratory (LLNL). In the second scenario, we run simulations for a large petascale HPC application based on a set of values that were used on the BG/Q platform Mira at Argonne National Lab (ANL). The simulator code is publicly available at *http://graal.ens-lyon.fr/~yrobert/multilevel.zip*, so that interested readers can experiment with it and build relevant scenarios of their choice.

3.1 Simulation setup

Checkpoint and recovery costs both depend on the volume of data to be saved, and are mostly determined by the hardware resource used at each level. As such, we assume that recovery cost for a given level is equivalent to the corresponding checkpointing cost, i.e. $R_i = C_i$ for $1 \leq i \leq k$. This a common assumption [12, 7], even though in practice the recovery cost can be expected to be *somewhat* smaller than the checkpoint cost [7, 8].

The simulator is given a platform with k levels of errors and their MTBFs $\mu_i = 1/\lambda_i$, and the resilience parameters C_i and R_i. For each of the 2^{k-1} possible subsets of levels (the last level is always included), we do the following:

- take the optimal pattern from Section 2;
- fix the total amount of work to $100W^{\text{opt}}$;
- try all possible roundings (floor and ceiling) of the (rational) optimal number of checkpoints at each level. Each experiment is run 10000 times and results are averaged;
- return the two solutions with minimal and maximal overhead, and compare them with the theoretical bound.

3.2 Medium HPC system

3.2.1 Platform settings

The target platform is Coastal, a medium-sized HPC system of 1104 compute nodes at LLNL, whose parameters are given as set (A) in Table 1. The first row of Table 1 presents the checkpoint costs, and the corresponding MTBF used in the first scenario.

The Coastal platform has been used to evaluate the Scalable Checkpoint/Restart (SCR) library by Moody et al. [12], who provide accurate measurements for μ and C using real applications. There are $k = 3$ levels of checkpoint. First level checkpoints are written to the local RAM of the node, and this is the fastest method ($0.5s$). Second level checkpoints are also written to local RAM, but small sets of nodes collectively compute and store parity redundancy data, which takes a little while longer ($4.5s$). Lastly, Lustre is used to store third-level checkpoints onto the parallel file system, which takes $1051s$. Failures were analyzed in [12], and the error rates at each level are those reported in Table 1. Note that the error rate at level 2 is higher than that of level 1 and 3.

Levels	N_1	N_2	N_3	W^{opt}	Sim. Ov.	Th. Ov.	L. b.
{3}	-	-	1	2.96e4	7.75e-2	7.11e-2	7.11e-2
{1,3}	14	-	1	3.09e4	7.44e-2	6.85e-2	6.85e-2
	13	-	1	3.09e4	7.40e-2	6.85e-2	
{2,3}	-	35	1	7.27e4	**3.41e-2**	**3.33e-2**	**3.33e-2**
	-	34	1	7.25e4	3.42e-2	**3.33e-2**	
{1,2,3}	33	33	1	7.27e4	3.45e-2	3.35e-2	3.35e-2
	32	32	1	7.24e4	3.45e-2	3.35e-2	

Table 2: Simulation results using set of parameters (A) for all possible roundings.

3.2.2 Results

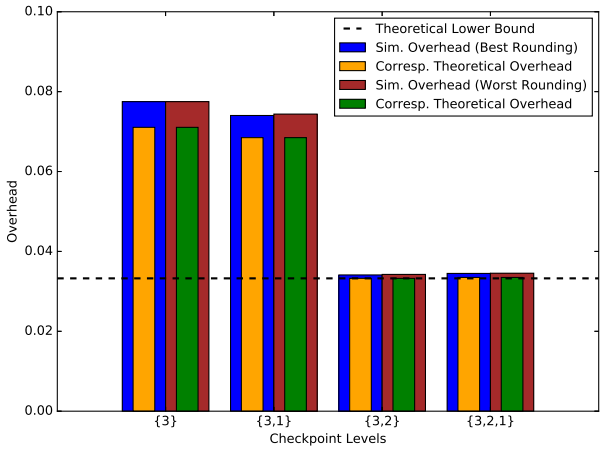


Figure 4: Simulated and (corresponding) theoretical overhead for best and worst rounding using set of parameters (A).

Figure 4 and Table 2 present the results of the simulation. With $k = 3$, there are four possible level subsets. First, the difference between simulated and theoretical overhead is very small, with a difference $< 1\%$ in overhead values, and a relative error ranging from $\approx 2\%$ (levels 2 and 3) to $\approx 9\%$ (level 3), which shows the accuracy of the first-order approximation for this set of values. The simulated overhead is always higher than the theoretical one, which is expected, because the first-order approximation is ignoring some lower order terms. Next, we observe that the difference between the best and worst integer roundings for the number of checkpoints at each level is almost negligible. All roundings yield similar overhead for this platform. Finally, we observe that the best subset (levels 2 and 3) improves the overhead by over 50% compared to using only a single level 3 checkpoint.

3.3 Petascale HPC system

3.3.1 Platform settings

Here, we run simulations for the large BG/Q platform Mira running LAMMPS application at ANL [1], whose parameters correspond to set (B) in Table 1. Four checkpoint levels are provided by the FTI library [2] ($k = 4$): Local checkpoint; Local checkpoint + Partner-

copy; Local checkpoint + Reed-Solomon coding; and PFS-based checkpoint. The MTBFs correspond to a default failure rate commonly used for petascale HPC applications [2, 12, 7].

3.3.2 Results

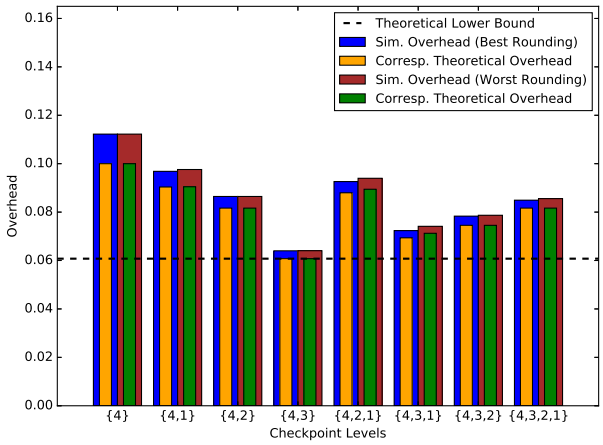


Figure 5: Simulated and (corresponding) theoretical overhead for best and worst rounding using set of parameters (B).

Figure 5 presents the results for the set of parameters (B). There are 8 possible subsets. As before, we observe that the theoretical overhead is always slightly smaller that the simulated one, with a difference $< 2\%$ in overhead values, and a relative error of $\approx$ 5-7%, demonstrating the accuracy of the model. For this platform, the simulated overheads vary from **6.4% (optimal subset with levels 4,3)** to 11.2% (single level 4 checkpoint). For a given level subset, the rounding does not play a significant role, as W^{opt} is also increased, or decreased in consequence. For instance, we observe that for subset $4, 3, 2, 1$, twice fewer checkpoints of levels 1 and 2 are used for the second rounding in Table 3, but W^{opt} is also reduced by 33%, so that for the same amount of work, the number of checkpoints does not change by much. Then, we can see that the pattern length W^{opt} for the smallest overhead is around $12000s$, but only $2000s$ for the largest overhead. Actually, the largest pattern length is obtained when using all the 4 levels. This is because using more checkpoints both increases the error-free overhead and reduces the time lost due to re-executions upon errors. As a consequence, and to mitigate the aforementioned overhead, the size of the pattern length W^{opt} increases (e.g. $W^{\text{opt}} = 1.47e4s$ for $4, 3, 2, 1$ and $N_1 = 20, N_2 = 10, N_3 = 5$ and $N_4 = 1$). And the reciprocal is true: when using fewer checkpoints, the error-free overhead decreases and the time lost upon re-execution increases. In order to compensate, W^{opt} decreases (e.g. $W^{\text{opt}} = 9.80e3s$ for $4, 3, 2, 1$ and $N_1 = 10, N_2 = 5, N_3 = 5$ and $N_4 = 1$).

3.4 Summary of results

From the simulation results, we conclude that the first-order approximation for the multi-level pattern provides an accurate performance model for systems where the λ_i's stay reasonably small. Hence, the results corroborate the analytical study and show the benefits of using only a few levels among all the available levels. In our case, the best thing is to consider a

Levels	N_1	N_2	N_3	N_4	W^{opt}	Sim. Ov.	Th. Ov.	L. b.
{4}	-	-	-	1	2.00e3	1.13e-1	1.00e-1	1.00e-1
{1,4}	4	-	-	1	3.10e3	9.74e-2	9.04e-2	9.02e-2
	3	-	-	1	2.87e3	9.74e-2	9.05e-2	
{2,4}	-	5	-	1	4.90e3	8.65e-2	8.17e-2	8.17e-2
{3,4}	-	-	14	1	1.25e4	**6.40e-2**	**6.08e-2**	**6.08e-2**
	-	-	13	1	1.18e4	6.41e-2	**6.08e-2**	
{1,2,4}	6	3	-	1	5.00e3	9.26e-2	8.80e-2	8.80e-2
	4	2	-	1	4.02e3	9.40e-2	8.94e-2	
{1,3,4}	18	-	9	1	1.32e4	7.27e-2	6.95e-2	6.91e-2
	16	-	8	1	1.21e4	7.24e-2	6.94e-2	
	9	-	9	1	1.04e4	7.36e-2	7.09e-2	
	8	-	8	1	9.54e3	7.41e-2	7.13e-2	
{2,3,4}	-	15	5	1	1.34e4	7.87e-2	7.45e-2	7.42e-2
	-	10	5	1	1.07e4	7.83e-2	7.45e-2	
{1,2,3,4}	20	10	5	1	1.47e4	8.49e-2	8.17e-2	8.05e-2
	10	5	5	1	9.80e3	8.56e-2	8.16e-2	

Table 3: Simulation results using set of parameters (B) for all possible roundings.

2-level equivalent pattern with $C_1 = 20$, $C_2 = 100$, $\mu_1 = 20,570$, $\mu_2 = 720,000$ achieving an overhead of 6.4% instead of 8.2% with the 4 initial levels described in Table 1. It also shows the efficiency of this new model of checkpointing (instead of using only one level of checkpoint) as the overhead is almost divided by 2 in our example.

4 Related work

Given the checkpointing cost and platform MTBF, classical formulas due to Young [16] and Daly [6] are well known to determine the optimal checkpointing period in the single-level checkpointing scheme. However, this method suffers from the intrinsic limitation that the cost of checkpointing/recovery grows with failure probability, and becomes unsustainable at large scale [9, 5] (even with diskless or incremental checkpointing [13]).

To reduce the I/O overhead, various two-level checkpointing protocols have been studied. Vaidya [15] proposed a two-level recovery scheme that tolerates a single node failure using a local checkpoint stored on a parter node. If more than one failure occurs during any local checkpointing interval, the scheme resorts to the global checkpoint. Silva and Silva [14] advocated a similar scheme by using memory to store local checkpoints, which is protected by XOR encoding. Di et al. [8] analyzed a two-level computational pattern, and proved equal-length segments in the optimal solution. They also provided mathematical equations that can be solved numerically to compute the optimal pattern length and number of segments. Benoit et al. [4] relied on disk checkpoints to cope with fail-stop failures and used memory checkpoints coupled with error detectors to handle silent data corruptions. They derived first-order approximation formulas for the optimal pattern length as well as the number of memory checkpoints between two disk checkpoints.

Some authors have also generalized two-level checkpointing to account for an arbitrary number of levels. Moody et al. [12] implemented this approach in a three-level Scalable Checkpoint/Restart (SCR) library. They relied on a rather complex Markov model to recursively compute the efficiency of the scheme. Bautista-Gomez et al. [2] designed a four-level

checkpointing library, called Fault Tolerance Interface (FTI), in which partner-copy and Reed-Solomon encoding are employed as two intermediate levels between local and global disks. Based on FTI, Di et al. [7] proposed an iterative method to compute the optimal checkpointing interval for each level with prior knowledge of the application's total execution time. Hakkarinen and Chen [10] considered multi-level diskless checkpointing for tolerating simultaneous failures of multiple processors. Balaprakash et al. [1] studied the trade-off between performance and energy for general multi-level checkpointing schemes.

While all of these works relied on numerical methods to compute the checkpointing intervals at different levels, this paper appears to be the first one to provide explicit formulas on the optimal parameters in a multi-level checkpointing protocol (up to first-order approximation as in Young/Daly's classical result).

5 Conclusion

This work has studied multi-level checkpointing protocols, where different levels of checkpoints can be set; lower levels deal with frequent errors that can be recovered at low cost (for instance with a memory copy), while higher levels allow us to recover from all errors, such as node failures (for instance with a copy in stable storage). We consider a general scenario with k levels of faults, and we provide explicit formulas to characterize the optimal checkpointing pattern, up to first-order approximation. The overhead turns out to be of the order of $\sum_{\ell=1}^{k} \sqrt{2\lambda_\ell C_\ell}$, which elegantly extends Young/Daly's classical formula.

The first-order approximation to the optimal k-level checkpointing pattern uses rational numbers of checkpoints, and we prove that all segments should have equal lengths. We corroborate the theoretical study by a set of simulations, demonstrating that a solution greedily rounding rational values, leads to an overhead very close to the lower bound. Furthermore, we provide a dynamic programming algorithm to determine those levels that should be used, and simulations confirm the optimality of the subset of levels returned by the dynamic programming algorithm.

The problem of finding a first-order optimal pattern with an integer number of segments to minimize the overhead remains open. It may well be the case that such an integer pattern is not periodic at each level and uses different-length segments. However, the good news is the rounding of the rational solution provided in this paper seems quite efficient in practice; this should be confirmed in future work by further experimentations.

Acknowledgment

This research was funded in part by the European project SCoRPiO, by the LABEX MILYON (ANR-10-LABX-0070) of Université de Lyon, within the program "Investissements d'Avenir" (ANR-11-IDEX-0007) operated by the French National Research Agency (ANR), by the PIA ELCI project, and by the ANR Rescue project. Yves Robert is with Institut Universitaire de France.

References

[1] P. Balaprakash, L. A. B. Gomez, M.-S. Bouguerra, S. M. Wild, F. Cappello, and P. D. Hovland. Analysis of the tradeoffs between energy and run time for multilevel check-

pointing. In *Proc. PMBS'14*, 2014.

[2] L. Bautista-Gomez, S. Tsuboi, D. Komatitsch, F. Cappello, N. Maruyama, and S. Matsuoka. FTI: High performance fault tolerance interface for hybrid systems. In *Proc. SC'11*, 2011.

[3] A. Benoit, A. Cavelan, Y. Robert, and H. Sun. Optimal resilience patterns to cope with fail-stop and silent errors. Research report RR-8786, INRIA, 2015. Available at `graal.ens-lyon.fr/~yrobert/rr8786.pdf`. Short version appears in IPDPS'16.

[4] A. Benoit, A. Cavelan, Y. Robert, and H. Sun. Optimal resilience patterns to cope with fail-stop and silent errors. In *Proc. IPDPS'16*, 2016.

[5] G. Bosilca et al. Unified model for assessing checkpointing protocols at extreme-scale. *Concurrency and Computation: Practice and Experience*, 2013.

[6] J. T. Daly. A higher order estimate of the optimum checkpoint interval for restart dumps. *FGCS*, 22(3):303–312, 2006.

[7] S. Di, M. S. Bouguerra, L. Bautista-Gomez, and F. Cappello. Optimization of multi-level checkpoint model for large scale HPC applications. In *Proc. IPDPS'14*, 2014.

[8] S. Di, Y. Robert, F. Vivien, and F. Cappello. Toward an optimal online checkpoint solution under a two-level HPC checkpoint model. *IEEE Trans. Parallel & Distributed Systems*, 2016, preprint available on the IEEE digital library.

[9] K. Ferreira, J. Stearley, J. H. I. Laros, R. Oldfield, K. Pedretti, R. Brightwell, R. Riesen, P. G. Bridges, and D. Arnold. Evaluating the Viability of Process Replication Reliability for Exascale Systems. In *Proc. SC'11*, pages 44:1–44:12, 2011.

[10] D. Hakkarinen and Z. Chen. Multilevel diskless checkpointing. *IEEE Transactions on Computers*, 62(4):772–783, 2013.

[11] T. Hérault and Y. Robert, editors. *Fault-Tolerance Techniques for High-Performance Computing*, Computer Communications and Networks. Springer Verlag, 2015.

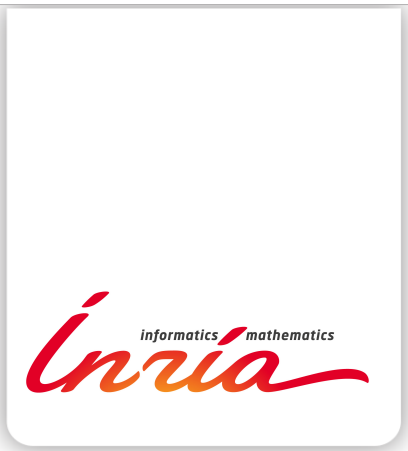
[12] A. Moody, G. Bronevetsky, K. Mohror, and B. R. d. Supinski. Design, Modeling, and Evaluation of a Scalable Multi-level Checkpointing System. In *Proc. SC'10*, 2010.

[13] J. Plank, K. Li, and M. Puening. Diskless checkpointing. *IEEE Trans. Parallel Dist. Systems*, 9(10):972–986, 1998.

[14] L. Silva and J. Silva. Using two-level stable storage for efficient checkpointing. *IEE Proceedings - Software*, 145(6):198–202, 1998.

[15] N. H. Vaidya. A case for two-level distributed recovery schemes. *SIGMETRICS Perform. Eval. Rev.*, 23(1):64–73, 1995.

[16] J. W. Young. A first order approximation to the optimum checkpoint interval. *Comm. of the ACM*, 17(9):530–531, 1974.



RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée

655 avenue de l'Europe Montbonnot

38334 Saint Ismier Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399