# ON THE MODELLING OF OPTIMAL COORDINATED CHECKPOINT PERIOD IN SUPERCOMPUTERS

José A. Moríñigo, Manuel Rodríguez-Pascual, Rafael Mayo-García
Dept. of Technology, CIEMAT
Avda. Complutese 40, Madrid 28840, SPAIN
E-mail: josea.morinigo@ciemat.es

## ABSTRACT

This work revises current assumptions adopted in the checkpointing modelling and evaluates their impact on the attained prediction of the optimal coordinated single-level checkpoint period. An accurate a priori assessment of the optimal checkpoint period for a given computing facility is necessary as it drives the incurred overhead due to frequent checkpointing and, as a result, implies a drop in the resource steady-state availability. The present study discusses the impact of the order of approximation used in the single-level coordinated checkpoint modelling and follows on extending previous results of the optimal checkpoint period to explore the effects of the checkpoint rate on the cluster performance under total execution time and energy consumption policies, and in terms of resource availability. A consequence of a prescribed checkpoint rate with current technology is a critical size of the cluster above which the attained availability is too poor to become a cost-effective platform. Thus, some guidelines for the cluster sizing are indicated.

## 1. INTRODUCTION

High availability in supercomputers is an essential issue, even more at the coming of the exascale era, where higher failure rates in the environment are expected to occur as more and more components will shape the computing facility. It means that an adequate checkpoint protocol will be a “hot spot” to yield resource availability over an adequate threshold, such that a cost-effective high performance computing (HPC)

scenario can be achievable. But, how to correctly state the term “adequate” in a context of increasing overhead because of demanding checkpointing in computers with a extreme number of computing units?

It is clear the importance of assessing this overhead with accuracy as the trans-petascale computing is facing the exascale era. Complexity of applications is growing and the available higher HPC performance is driving the term ‘large’ even further. In particular, large-scale scientific applications are evolving incorporating finer resolution in time and space, as well as more detailed physics in the modelling. Typically, in strong and weak scaling scenarios the goal is to compute problem solutions as fast (prescribed size problem) or as big (prescribed computing time) as possible, respectively. But the integrity of the computations may be exposed to a too high failures rate in current and future supercomputers. Then significant or total lost of an amount of execution may occur at some point in the execution. This brings other dimension within the involved complexity: a two-fold algorithmic and system resilience must be added to fight against failures in the environment, such that they can be tolerated or minimized. One common technique in use is checkpointing & rollback, which guarantees the persistence of the computed data in the presence of failures. The computed data are copied someplace within the resource (local disk, remote disk, volatile memory, or a combination of these), so in case of a failure the affected part of the machine will re-start the computations out of the copied data, this leading to a smaller computational waste caused by a failure. Going to the extreme scale, the overall reliability of large computing resources can be too low for efficient usage even though component-level reliabilities may be very high (recent example of this have been reported for supercomputers Jaguar, Titan and others [1-4]). A consequence is that beyond certain size of HPC systems, it will be unfeasible to successfully execute parallel applications if frequent failures take place. It is known that modern HPC systems exhibit mean time between failures (*MTBF*) of about days to only hours due to hardware and software errors [5,6]. That means that the HPC availability for useful computing is closely linked to its physical size (in terms of number of individual components put together into the system) and data recovery capability implementation. This situation arises as a big challenge to overcome in the trans-petascale era.

Among the variety of recovery techniques explored by modeling, simulation and experiments at scale, the de-facto one in HPC is checkpointing and rollback, in which

the state of the parallel application is saved at successive time instant over the computing time. Being the single-level coordinated checkpoint scheme the most implemented one [7-9], other sophisticated versions of fault-tolerant protocols have been proposed during the last years, as it is the case of the multi-level (two-level and beyond) checkpointing [10-14] or the hierarchical approach among others [15-20]. Nevertheless there is still a lack of understanding of key aspects (like the effects of resource heterogeneity on the failures rate distribution modelling over the system lifetime or the inclusion of second- and higher-order formulations for improved accuracy). This work aims at filling this gap.

The present investigation provides the following contributions:

- Out of the analytical first-order single-level coordinated checkpoint model presented in [21], the expressions of the optimal checkpoint period (both to minimize the total execution time and energy consumption) are hereby modified by including a functional dependence with the checkpoint rate to take into account the foreseen evolution of the checkpoint technology, which it is expected to be implemented in future supercomputers.
- The inclusion of the blocking- / non-blocking effect into the coordinated checkpoint model is discussed within the context of first- and second-order formulations for accurate prediction of the optimal checkpoint period. A quantification is done.
- Guidelines for sizing the cluster (in number of nodes or sockets) using information from current clusters (taken as the reference case for comparison purposes) are provided to match a HPC availability criterion in the frame of an evolving checkpoint technology. With such a prediction, it will be easier to fix the cluster size to match an a priori prescribed availability for a given checkpoint overhead.

## 2. RELATED WORK

The high cost of checkpointing when using single-level schemes has motivated conducting research in more advanced recovery schemes capable of providing multiple levels of fault-tolerance, thus to achieve better performance than single-level

schemes. Such studies spread over the last two decades, mainly focusing on multi-level (mostly two-level) and hierarchical models to provide a better average performance overhead of failure recovery by combining saving data in various parts of the supercomputer, i.e., using local memory for those failures of greater probability of occurring; and using remote, stable storage for those less probable failures. In this direction some authors have recently [16] tackled the mathematical aspects from a standpoint of achieving unified checkpoint formulations.

The work of Vaidya [22] provides an analytical approach based on discrete Markov Chains (MCs) to evaluate the performance of two-level checkpoint schemes. In [10, 14] these two-level schemes have been revisited also following a MCs approach and stochastic renewal reward process-based methodologies. Interestingly, a major result is that two-level schemes exhibit superiority in scenarios of long-running applications, but are inferior on short-running ones. Both constitute limit scenarios with an optimal checkpoint solution for each. So if the workload changes over time, an adaptive checkpointing scheme might be seen as a compromise in between, making single-level checkpoint schemes relevant in the context to provide resilience in more general failure scenarios, and hence the interest of achieving their better modelling.

Next, the related work about single-level checkpoint (under focus in the present investigation) is detailed. Daly [23] refines the study conducted by Young [24] by deriving a higher-order analytical expression using the probability theory (PT) for quantifying the optimum restart interval that minimizes the total runtime. The second-order approximation clearly shows the improvement in the prediction of the optimized checkpoint period in those cases where it is of the order of the mean time between failures (*MTBF*). Precisely this is an approaching horizon of the coming exascale era. Gelenbe et al. [25] deduce an equation for the optimum checkpoint interval to maximize system availability in a context of short-running tasks. Their study generalizes the methodology for age-dependent processes [26] (both Weibull and Poisson failure rates are assumed and their results compared). Parameters as the system workload are included and it is assumed that failures do not occur during checkpoint and recovery. This last condition is relaxed in the work of Vaidya [27, 28] based on MCs, which presents a model for predicting the periodic checkpoint overhead taking into account the checkpoint latency and it shows that a large increase in latency is acceptable if it is accompanied by a reduction in overhead. His

study analyses the difference between blocking (sequential) and non-blocking (forked; that is, computation is overlapped with stable storage access) checkpoint. Ling et al. [29] formulate the optimal checkpointing scheduling using calculus of variations (CoV). They derive a closed-form equation which links optimal checkpointing frequency with a general failure rate; hence they derive a cost-optimal aperiodic checkpointing sequence. For workloads varying over time, they propose to subdivide the total time into segments and then to tune the Poisson arrival rate in each one. One major finding of their approach is that optimal checkpointing is equally spaced if and only if the system failure time is exponentially distributed in infinite-time horizons. It is demonstrated that this is not true for finite-time horizons, as stated in [30], which also gives algorithms for the optimum checkpoint frequency in cases of generic distribution functions.

With CoV, it has also been explored the assumption that failures obey to a Weibull distribution [8], but this issue remains controversial since several HPC resources exhibit non-Poisson, non-Weibull failure distributions [5, 30-32]. Hence, various authors point out to probabilistic models formulated using non-specific failure laws, solved numerically [20, 32]. Aupy et al. [21] provide a first-order analytical model following the PT to determine the optimal coordinated checkpoint period to minimize the execution time and the energy consumption and a discussion of both criteria is given. Their model incorporates a slowdown parameter which permits to adjust the intensity of the blocking effect (from full- to non-blocking operation) [9, 21] and it extends the first-order formulae of Young [24] to include it.

In summary, three major mathematical approaches have appeared in the literature to deal with the coordinated checkpointing modelling in the last decades. They correspond to models based on MCs, CoV, and PT. Table 1 shows their main constrains and applicability. It is noticed that some parameters explored in single-level formulations (i.e., blocking, workload intensity…) seems difficult to include in multi-level formulations due to the extra complexity.

**Table 1**. MC, CoV, and PT approaches to the optimal checkpoint modelling

| | Mathematical approach | | |
|---|---|---|---|
| | MCs | CoV | PT |
| Failure rate law | Poisson | general | Poisson |
| CKP interval | periodic | aperiodic [1] | periodic |
| Failure at $C$ and $R$ [2] | Yes | no ($C, R << T$) | some formulations |
| Non-blocking | yes [3] | no | yes |
| Other | - | based on renewal process theorem | - |

[1] Being periodic a particular case
[2] $C$: checkpoint time interval; $R$: rollback time interval; $T$: checkpoint period
[3] Controlled by a parameter

## 3. OPTIMAL CHECKPOINT PERIOD

This section provides some equations of the optimal checkpoint period following the deductions of [21] for single-level coordinated checkpoint schemes. The present study modifies the mentioned deductions by including a functional dependence with the checkpointing, rollback, and recovery rates. The proposed approximation serves to mimic the speed up of the checkpointing technology over the next years by means of a tuning parameter which links the evolution of the checkpointing technology with the size of the cluster. This functional dependence yields a generalization of the checkpointing overhead and allows to explore two situations of interest: a present-day scenario to quantify how a given cluster would behave when additional nodes are added under the current checkpoint technology; and a future scenario where at the same time of enlarging the platform with additional nodes, an evolved checkpoint technology is implemented. Both situations provide information which allows to better bound the cluster realizable size. Some particularities of heterogeneous clusters are also analysed in what follows.

### 3.1 Homogeneous Node Cluster

Let suppose a cluster built of *N* equal nodes (computing units). Assuming each one has an individual mean time between failures $MTBF_{unit}$, the hypothesis of a Poisson failure law for the whole *N*-size cluster leads to the definition $MTBF=MTBF_{unit}/N$. It is implicit within this assumption that all computing units behave the same, and that the failure rate attributed to the combination of system software and hardware granularity (i.e., having the cores grouped into multi-core sockets or in various CPUs per node) can be characterized at node-level. So the $MTBF_{unit}$ takes into account the system software; that is, it considers not only the reliability of individual cores (which run the application codes), but also the system software as an important source of failure. This assumption, albeit controversial according to the reliability theory and the experiments reported in the literature [5, 8, 30-33], is an approximation and applied here in the deductions that follow. The more complex case of an heterogeneous node cluster, which implies to tackle with a further degree of granularity, is introduced in the next subsection.

A first-order extension of the Young's [24] and Daly's [23] formulae for optimal checkpointing in a cluster of *N* computing units is provided in [21] according to

$$T_{opt}^{min-time} = \sqrt{2C(1-\omega)[MTBF - (R + D + \omega C)]} \qquad (1)$$

where $T_{opt}^{min-time}$ is the optimum compute time between saving checkpoint data; that is, the execution is partitioned into periods of duration $T_{opt}^{min-time}$. In this expression, *C* is the time to save a checkpoint file, thus a checkpoint of length *C* is done every period. And *R* and *D* are the recovery and downtime lengths, respectively, in time units. Checkpoints are taken at regular intervals. The superscript "*min-time*" refers to apply a criterion (operational policy) of minimizing the total execution time. The parameter $\omega$ handles the degree of non-blocking: $\omega$=0 corresponds to fully blocking; and $\omega$=1 corresponds to fully non-blocking; that is, a checkpoint completely overlapped with computations, such that provides the particular case $T_{opt}^{min-time} = 0$. It is seen that the shorter the optimal period, the less work to re-execute after a failure, but it also means that a higher overhead caused by frequent checkpoints in a

failure-free execution scenario occurs. It is noticed that for $\omega$=0 and $R<<C$, $D<<C$, equation (1) collapses into the well known Young's and Daly's equation

$$T = \sqrt{2C \cdot MTBF}$$

Similarly, an expression for the optimal checkpointing period under the policy of minimizing the energy consumption of the running application may be deduced. To accomplish it, all the energy cost contributions over a generic interval of time $T$, are summed up to build the expression for the total energy consumed. The energy cost contributions are mainly four, defined as follows

- $P_{static}$: static power consumption, which corresponds to the base power consumed when the cluster is switched on. That is, the consumption during each time-step of the execution. It includes, for example, the cooling system power consumption.
- $P_{cal}$: calculation power consumption, which is the CPU power overhead when the cluster is active; it takes place in addition to $P_{static}$.
- $P_{I/O}$: power consumption overhead due to I/O operations. It is driven by checkpointing tasks and when recovery from a failure.
- $P_{down}$: power consumption overhead when one of more nodes are down and need to be rebooted (for coordinated checkpointing, when one node fails, the rest of the cluster stays idle).

In a generic non-blocking scenario ($\omega \neq 0$), $P_{cal}$ and $P_{I/O}$ consumptions are overlapped when checkpointing. The three last power contributions may be expressed as fractions of $P_{static}$ by introducing the parameters $\alpha$, $\beta$ and $\gamma$, hence they read

$$P_{cal}=\alpha P_{static}\,,\ P_{I/O}=\beta P_{static} \text{ and } P_{down}=\gamma P_{static}.$$

The total energy consumed $E_{total}$ may be derived as the summatory of these four power contributions

$$E_{total} = T_{cal}P_{cal} + T_{I/O}P_{I/O} + T_{down}P_{down} + T_{total}P_{static}$$

where $T_{cal}$, $T_{I/O}$, $T_{down}$ and $T_{total}$ are the times during which the respective power is used. After some math (see [21]), taking the first derivative of $E_{total}$ and setting it equal to zero leads to a quadratic polynomial over $T_{opt}$

$$A_{energy}{T_{opt}}^2 + B_{energy}T_{opt} + C_{energy} = 0 \qquad (2)$$

whose only positive root is the optimum time interval $T_{opt} = T_{opt}^{min-energy}$. The polynomial coefficients in eqn. (2) read

$$A_{energy} = \frac{\alpha\omega C + \beta R + \gamma D}{2MTBF^2} + \frac{b}{2MTBF} + \frac{a - \beta C}{4MTBF^2} + \frac{1}{2MTBF}$$

$$B_{energy} = \frac{(\beta C - a)b}{MTBF} - \frac{[\alpha(1-\omega) - \beta]C^2}{2MTBF^2}$$

$$C_{energy} = -\frac{ab(\alpha\omega C + \beta R + \gamma D + \mu)}{MTBF} - \beta C b^2 + \left(\frac{b}{2MTBF} + \frac{a}{4MTBF^2}\right)[\alpha(1-\omega) - \beta]C^2 \qquad (3)$$

with

$$a = (1-\omega)C, \quad b = 1 - \frac{D+R+\omega C}{MTBF}$$

The details of casting eqns. (1) – (3) can be found in [21] and not repeated here. In this work, it is assumed that the node rebooting cost is negligible compared to $P_{static,}$, then $\gamma \approx 0$ ($P_{down}=0$). In addition, the ratio of power consumption contributions

$$\rho = \frac{P_{static}+P_{I/O}}{P_{static}+P_{cal}} = \frac{1+\beta}{1+\alpha} \qquad (4)$$

can be built by inserting empirical values of the static, I/O and computing power budgets. This ratio serves to compare both optimal checkpoint criteria in different scenarios of energy consumption.

Furthermore, eqns. (1) and (2) are useful to estimate the optimal checkpoint period and to identify the maximum feasible size of a cluster for a given implemented checkpointing technology which automatically drives the order of magnitude of the overhead due to the *R*, *D* and *C* time intervals contributions in the equations ("feasible" here means a number of nodes that guarantee a steady-state operation above a prescribed threshold of resource availability).

To scale the cost of checkpointing of an application which spreads over *n* computing units, the following functional dependence is a wide accepted description of the checkpoint cost [34]

$$C_{ref}(n) = a_{ref} + b_{ref}/n + d_{ref} \cdot n \quad (5)$$

where $C_{ref}$ is a characteristic time for saving a checkpoint on *n* computing units in a reference cluster; and $a_{ref}$, $b_{ref}$ and $d_{ref}$ are fixed parameters into the definition of $C_{ref}$. The three terms in eqn. (5) relate to the following contributions. Let $V_{data}$ be the total volume of bytes to checkpoint. In a weak-scalable scenario, it can be written as the application memory footprint per computing unit (*Mem*) times *n*, that is $V_{data}=n \cdot Mem$. For strong-scalable cases, $V_{data}$ is a constant. In both cases each computing unit has $V_{data}/n$ bytes. Considering that $bw_{io}$ is the available I/O bandwidth, the overhead penalty on eqn. (5) results to be $\frac{V_{data}}{bw_{io}n}$, which leads to the functional dependence stated by the second term in eqn. (5), with $b_{ref}=V_{data}/bw_{io}$. In the evaluation of the checkpoint time $C_{ref}(n)$, a fixed start-up delay ($a_{ref,1}$) must be included to take into account the time needed to move the data from the computing unit or to proceed to the I/O storage. This delay is one of the two contributions to $a_{ref}$ in eqn. (5). In addition, the incurred time due to synchronization tasks among computing units to accomplish coordinated checkpoint may be modelled as another constant delay ($a_{ref,2}$) plus a term proportional to the number of participating computing units ($d_{ref} \cdot n$), that is $a_{ref,2}+d_{ref} \cdot n$. Grouping these contributions, it is $a_{ref}=a_{ref,1}+a_{ref,2}$ and the functional dependence indicated by eqn. (5) is satisfied.

Evolution of the checkpoint technology over time implies that checkpoint parameters $a_{ref}$, $b_{ref}$ and $d_{ref}$ in eqn. (5) will change accordingly. Being *N* the size of a cluster, let introduce the notation for general checkpoint parameters linked to their counterparts in a reference cluster

$$a_N=a_{ref}\cdot f_a(N)\ ,\ \ b_N=b_{ref}\cdot f_b(N)\ \ \ and\ \ d_N=d_{ref}\cdot f_d(N) \qquad (6)$$

where functions $f_i$ ($i=a,b,d$), satisfying $f_i(N_{ref})=1$, model the evolved checkpoint technology with its implementation in a $N$-size cluster. The generic expression for the checkpoint cost reads

$$C_N(n,N) = a_N + b_N/n + d_N\cdot n = a_{ref}\cdot f_a(N) + b_{ref}\cdot f_b(N)/n + d_{ref}\cdot f_d(N)\cdot n \qquad (7)$$

which conceptually includes two effects on the total checkpoint cost: the cost due to the size of the application running on $n$ nodes of the cluster. And the one because of the evolution of the checkpoint technology when implemented in a $N$-size cluster. Quicker coordinated checkpoint in the forthcoming bigger clusters (when available) is mimicked here by assuming decreasing $f_i$ functions over $N$. A first approximation to these functions in the present analysis is to assume a simple decreasing law over $N$ of the type

$$f_i \sim (N_{ref}/N)^{p_i} \ \ for\ i=a,\ b,\ d \qquad (8)$$

which is further simplified by assuming $p = p_a \sim p_b \sim p_d$, thus $f(N) = f_a \sim f_b \sim f_d = (N_{ref}/N)^p$ *and* $C_N=f(N)\cdot C_{ref}$. Exponent $p\geq 0$ tunes the hypothetical scenarios to come: from $p\sim 0$ (which means that future checkpoint rate stagnates at present values as indicated by eqn. (1), so a quick penalty will occur to the cluster availability at moderate values of $N$), to $p\sim 1$ (the resulting law $\sim 1/N$ approaches the same rate of drop than *MTBF*) or even a higher value. In resume, factor p adjusts the smoothness of how coordinated checkpoint technology will improve over years in bigger clusters. With this simplification and introducing a reference cluster to compare to, the new overhead intervals in eqn. (1) read

$$C = C_{ref}\left(\frac{N_{ref}}{N}\right)^p \ \text{ and } \ D + R + \omega C = (D_{ref} + R_{ref} + \omega C_{ref})\left(\frac{N_{ref}}{N}\right)^p \qquad (9)$$

where *ref* subscript corresponds to a reference cluster representative of high-end current ones ($N_{ref}=10^6$ nodes, $MTBF_{unit}=125$ years, $R_{ref}=C_{ref}$ and $D_{ref}=C_{ref}/10$. A realistic checkpoint interval in current clusters is about $C_{ref}=10$ minutes. Some figures

have been plotted with the more aggressive value $C_{ref}$=1 minute, characteristic of foreseen future checkpoint durations. The corresponding $C_{ref}$ value is indicated in each figure footprint).

Factor $p$=0 just recalls eqn. (1); that is, the scenario of having a fixed checkpoint overhead while building a bigger cluster by simply hardware stacking. The scaling rule for a general $p$ is

$$T_{opt}^{min-time}(N) = \left(\frac{N_{ref}}{N}\right)^{p} \sqrt{\frac{MTBF_{ref}\left(\frac{N_{ref}}{N}\right)^{1-p} - \left(D_{ref}+R_{ref}+\omega C_{ref}\right)}{MTBF_{ref}\frac{N_{ref}}{N} - \left(D_{ref}+R_{ref}+\omega C_{ref}\right)}} T_{opt}^{min-time}(N_{ref}) \qquad (10)$$

Taking orders of magnitude in this expression, it is seen that if for typical current HPC clusters with $O(10^4)$ to $O(10^5)$ nodes, the stacking of additional nodes to build a cluster ten times larger ($O(10^6)$ nodes), would imply to manage, roughly speaking, a three times higher checkpoint frequency during executions considering an scenario of $p$=1, which would provide a higher availability of the resource (under the assumption $C_{ref}$, $D_{ref}$ and $R_{ref}$ << $MTBF_{ref}$, the optimal checkpoint period scales as $(N_{ref}/N)^{p/2}$, so an increase of an order of magnitude in the size of the cluster impacts with a factor of about three in the frequency of optimal checkpoint). For a range of values of $p$, the higher checkpoint frequency is compensated by the shorter duration of the checkpoint events, thus the availability is not degraded. Nevertheless, under a p-threshold, availability will decay as $N$ increases.

In this path to build bigger clusters, factor p can be seen as a characterizing parameter of the checkpoint technology to match a feasible cluster size. A similar reasoning reads for $T_{opt}^{min-Energy}$ in eqn. (2) (using the definitions of eqn. (9)).

## 3.2 Heterogeneous Node Cluster

Current computer architectures are basically following two major trends [2, 3]: clusters based on heterogeneous nodes (CPUs with accelerators, multi-level memory); and clusters based on homogeneous nodes (groups of equal low-power cores with a single-level memory). One basic difference is that the first architecture has fewer nodes compared to the second one. Furthermore, to keep component counts for future systems within practical limits (< 1 million nodes), there will be

between $10^3$ to $10^4$ floating point computing units on a chip and it is expected on-chip parallelism to grow by a factor of 100x over the next decade. According to the reliability theory, this quite larger density of computing units per node (the so-called "fat" nodes) brings delicate issues regarding possible triggered chained-failures. In addition, it is common to enlarge homogeneous clusters by adding new nodes with more modern hardware, becoming a heterogeneous cluster.

Let suppose the simplified scenario of an *N*-size heterogeneous cluster built with two different types of nodes (i.e., a combination of CPUs and GPUs; or even the situation of two different types of CPUs: recent and older ones) in quantity $N_1$ and $N_2=N-N_1$, respectively. Assuming that corresponding failure rates of each type are adequately modelled by Poisson distributions $e^{-\lambda_1 t}$, $e^{-\lambda_2 t}$ (being $\lambda_1=1/MTBF_1$ and $\lambda_2=1/MTBF_2$ the unitary failure rate of the nodes in the respective portions of the cluster), the entire cluster reliability function $R$ (i.e., the probability of no failure in the interval $(0,t]$) follows as

$$R(t) = R_1(t)R_2(t) = \prod_{i=1}^{N_1} R_i(t) \prod_{i=N_2}^{N} R_i(t) = \left(e^{-\lambda_1 t}\right)^{N_1}\left(e^{-\lambda_2 t}\right)^{N_2} = e^{-N_1\lambda_1 t - N_2\lambda_2 t} \tag{11}$$

and the average failure rate of the entire cluster results to be

$$\lambda_N = -\frac{dR/dt}{R} = N_1\lambda_1 + N_2\lambda_2 \tag{12}$$

which also yields a Poisson failure rate distribution, thus it retains the memoryless behavior, which is a consequence of the superposition of renewal processes [35]. Then, the *MTBF* of the entire system reads

$$MTBF = MTBF_1 \cdot MTBF_2 / (N_1 \cdot MTBF_2 + N_2 \cdot MTBF_1) \tag{13}$$

These expressions are still valid for nonhomogeneous (aging) Poisson processes, where it can be assumed that $\lambda_N$ exhibits a parametrical dependence with time, that is $d\lambda/dt = O(\lambda_N/T_{life})$, being $T_{life}$ the operational lifetime of the cluster. Then eqns. (12) and (13) hold as a first-order statistical description of the system.

A more complex situation arises when manufacture and integration variability of the nodes is considered in the modelling. Assume that both portions of the cluster are built using nodes supplied under a degree of variability in their unitary Poisson distributions. Hence, suppose that for the $N_1$ nodes portion, manufacture variability implies that its nodes have been stacked out of a population with a varying failure rate among two types: a proportion $q$ with failure rate $\lambda_{1,q}$ and a proportion $1-q$ with failure rate $\lambda_{1,1-q}$. The reliability function of the $N_1$ portion corresponds then to

$$R_1 = \prod_{i=1}^{N_1} R_{i,1} = \left(qR_{i.q} + (1-q)R_{i,1-q}\right)^{N_1} = \left(qe^{-\lambda_{1,q}\,t} + (1-q)e^{-\lambda_{1,1-q}\,t}\right)^{N_1} \quad (14)$$

Similar reasoning follows for the $N_2$ portion of the cluster (now $p$ and $1-p$ denote the respective variability proportions)

$$R_2 = \prod_{i=N_2}^{N} R_{i,2} = \left(pR_{i.p} + (1-p)R_{i,1-p}\right)^{N_2} = \left(pe^{-\lambda_{2,p}\,t} + (1-p)e^{-\lambda_{2,1-p}\,t}\right)^{N_2} \quad (15)$$

The average failure rate of the cluster is then

$$\begin{aligned}\lambda_N &= -\frac{d(R_1R_2)/dt}{R_1R_2} \\ &= N_1\frac{q\lambda_{1,q}e^{-\lambda_{1,q}t} + (1-q)\lambda_{1,1-q}e^{-\lambda_{1,1-q}t}}{qe^{-\lambda_{1,q}t} + (1-q)e^{-\lambda_{1,1-q}t}} \\ &\quad + N_2\frac{p\lambda_{2,p}e^{-\lambda_{2,p}t} + (1-p)\lambda_{2,1-p}e^{-\lambda_{2,1-p}t}}{pe^{-\lambda_{2,p}t} + (1-p)e^{-\lambda_{2,1-p}t}}\end{aligned} \quad (16)$$

from which setting $q$=$p$=1, eqn.(12) is recovered.

A significant difference compared to the first simplified scenario is that the included population variability ($q\neq0$, $p\neq0$) implies that the mean time between failures of the cluster becomes now time dependent. The average $\lambda_N = N_1\{q\cdot\lambda_{1,q} + (1-q)\cdot\lambda_{1,1-q}\} + N_2\{p\cdot\lambda_{2,p} + (1-p)\cdot\lambda_{2,2-p}\}$ holds only at the beginning of life ($t$=0). Hence, the Poisson distribution is lost in strict sense for $t$>0 and the failure rate of the resulting distribution shifts over time and decreases monotonically as $t \to \infty$.

This departure from Poisson law attributed to the nodes variability in the case of the heterogeneous cluster is also applicable to the homogeneous node cluster as well. And it agrees with the experimental observation that many clusters obey to non-Poisson distributions (sometimes modelled as Weibull or exponential failure processes, to include the time dependence) albeit their nodes are in a first approximation well represented by Poisson processes. However, the attained decrease over time does not explain the typically observed higher rate of interruptions because of system aging (i.e., the bath-tube plot), but fits with the behavior reported in some investigations which collect the failure rate observed in a variety of HPC systems [5]. The change of $\lambda_N$ over time is

$$\frac{d\lambda_N/dt}{\lambda_N} = -\frac{1}{\lambda_N}\frac{\frac{d^2R}{dt^2}R-\left(\frac{dR}{dt}\right)^2}{R^2} = \lambda_N + \frac{\frac{d^2R}{dt^2}}{\frac{dR}{dt}} \qquad (17)$$

Taking orders of magnitude, it follows

$$O\left(\frac{d\lambda_N/dt}{\frac{\lambda_N}{T_{life}}}\right) \sim \lambda_N T_{life} + O(1) \gg 1 \qquad (18)$$

(say $T_{life}$~ $10^8$s, corresponding to a lifetime of about 5 years). Since $d\lambda_N/dt \sim \lambda_N^2$ with $\lambda_N$ ~$10^{-1} - 10^{-4} s^{-1}$, eqn. (18) shows that the time variation of the failure distribution is significant and a large enough number of time segments must be considered within [0, $T_{life}$] under the age-dependent Poisson assumption, which has been suggested in [25, 29] to cite some. But the disparity of time scales makes this approach questionable as the number of time segments to take into account (each linked to an age-dependent Poisson process) results to be large and requires frequent update with failure information that may not be available for accurate results. This sub-section stresses some drawbacks of assuming a Poisson failure law for the cluster as has been done in the previous sub-section for homogeneous nodes clusters. Nevertheless, the mentioned assumption comes to simplify the involved math and to yield a first approximation to the problem under analysis.

### 3.3 Order of Approximation

An important issue of clusters is to operate with the highest possible availability, which implies to perform efficient checkpointing. From a predictive standpoint, it is necessary to develop accurate enough checkpoint models and to quantify how large the error can be when using them for the checkpoint period calculation (too coarse models impact negatively in predicting the available computing time of the resource). First- and higher-order checkpoint models have appeared in the literature to deal with this problem. Figure 1 depicts first- and second-order approximations to the optimal checkpoint period ($T_{opt}$) discussed in the literature by Young [24], Gelenbe et al. [25], Vaidya [27], Daly [23] and Aupy et al. [21] in the context of single-level coordinated checkpoint schemes. The comparison quantifies that first-order solutions depart very quickly from the second-order ones when the number of nodes is huge, which will be the case in the exascale era and the incurred departure is rather dependent on the checkpoint interval duration (plotted non-dimensionalized as $C/C_{ref}$). On the contrary, the curves plotted for $10^6$ nodes (high-end current supercomputers) show that both first- and second-order models almost coalesce when checkpoint is done at high to moderate rate ($C/C_{ref} < 0.1$). This is not the case for $10^8$ nodes, where it is visible that even for $C/C_{ref} \sim 10^{-3}$ (that is $C \sim 0.6$s) a departure of about a 20% still occurs between both approaches. It is seen in Figure 1 that first- and second-order predictions of $T_{opt}$ for an exascale cluster of $10^8$ nodes with a future rather optimistic $C \sim 1$min ($C/C_{ref} = 0.1$) differ by more than double or triple.

It is noticed that Figure 1 corresponds to full workload operation, so the region over $T_{opt}/MTBF = 1$ is of no applicability in cluster operation (optimal checkpoint solutions at partial workload operation exist for $T_{opt}/MTBF > 1$, as shown in [25]). The sensitivity to partial workload is shown in Figure 2, where it is visible that at very small checkpoint durations within an exascale scenario, the impact of shifting the workload from low (25%) to full (100%) is halving the optimal checkpoint interval. The asymptotic behavior $T_{opt}/MTBF \rightarrow 1$ with a slow checkpoint mechanism and full workload is captured by the second-order Vaidya's and Gelenbe at al. approximations. Besides, Daly's second-order equation closely follows the same tendency, but it drops and departs from it for quite slow checkpoint intervals. The effect of the duration of the checkpoint interval on the cluster availability is plotted in Figure 3 (full load case). It is

seen how critical will be to move from $10^6$ to $10^8$ nodes since the graphs depict a more abrupt drop at large checkpoint intervals. As a result, there is a smaller range of $C/C_{ref}$ with acceptable availability. For a fixed requirement of availability, this states a threshold and a range of feasible checkpoint rates.

Considering the full workload case, it is relevant to quantify the impact of including or not the non-blocking effect into the coordinated checkpoint protocol, as it is done in the Aupy et al. first-order formulation. This is shown in Figure 4, where Aupy's model is plotted for $\omega$=0 (full-blocking) and $\omega$=0.5 (half-blocking) superimposed with the Gelenbe et al. (full-blocking) second-order formulation for comparison purposes. Predictions for $10^6$ nodes look quite similar with both models for $\omega$=0, except at low checkpoint rate. The fork of computing and checkpointing following a 50 - 50% ratio in Aupy's model implies an increase of about 50% in the checkpoint frequency, which is large enough to be considered in the modelling. For the scenario of $10^8$ nodes the effect of $\omega$ is similar (of about 50% in terms of checkpoint frequency variation) but the comparison shows a definite departure of Gelenbe's plot from Aupy's plot for the entire $C/C_{ref}$ range. A similar result is obtained if Vaidya's second-order formulation is compared. The prediction provided by Aupy's with $\omega$=0 and $\omega$=0.5 over- and underestimates, respectively, the second-order prediction of the optimal checkpoint frequency at high checkpoint rates. Besides, their first-order solution provides an overprediction of the optimal period at moderate to slow $C/C_{ref}$. On the other side, at high-rate checkpoint both approaches yield a rather close value for the checkpoint frequency.

Current clusters with typical checkpoint durations (say, 10 minutes as a realistic value) show small differences between first and second-order performance predictions (see Figure 1), so the order of the approximation means a low impact from the users point of view. Thus, in the scenario of current clusters, the non-blocking property seems to be more important from the users perspective as it serves to reduce the execution time of their applications (by a 10% or even more according to estimations for petascale supercomputers) and, on the contrary, it is noticed that second-order $T_{opt}$ predictions introduce minor corrections to the cluster operation in this case.

This situation will change in the foreseen exascale systems, since much smaller checkpoint durations will be required to hold the overhead within limits and to operate

with good resource availability. To this regard, values of checkpoint duration at about 1 minute imply *C/Cref* ~0.1 in Figure 1, which represents a more sensitive region to the second-order prediction because of the impact of the checkpoints duration on the availability drop (this region of checkpoint rates shows that the departure between the first- and second-order plots implies an error in the $T_{opt}$ prediction of more than 100%). The notably smaller $T_{opt}$ predicted by the second-order approach makes the availability to drop, which means to accept greater average waiting times of the users queued jobs. In future clusters with workloads dominated by short-running applications, this constitutes a major issue, as important as the potential reduction of the execution time because of the implementation of a non-blocking checkpoint protocol. And to deal with it, a more accurate prediction of the $T_{opt}$ is mandatory since, as seen in Figure 1, it may introduce a more than a two-fold factor in the estimation of the checkpoint duration.

It can be concluded that second-order predictions deserve as much attention as the inclusion of non-blocking effects in the modelling because of the foreseen impact of future checkpoint rates on the resource availability, mostly when exascale computers are into the focus and considering the challenges of achieving better but probably moderate checkpoint rates. The inclusion of the penalty due to a smaller resource availability in addition to the attained speedup of the applications execution caused by a non-blocking checkpoint protocol, will permit a more accurate assessment of the mean overhead at system level.

## 4. CLUSTER SIZING

To assess the cluster operation policies (goodness of minimizing the total execution time or the energy consumption) according to the presented model, several figures are provided. Two figures of merit are depicted over the number of nodes in Figures 5, 6 and 7 showing the ratio of total execution time, following the definition Time($T_{opt}^{min-energy}$)/Time($T_{opt}^{min-time}$); and the ratio of energy consumption, defined as Energy($T_{opt}^{min-time}$)/Energy($T_{opt}^{min-energy}$). Both figures of merit are plotted in two checkpoint rate scenarios: for a current checkpoint rate (here a value $C_{ref}$=10 minute is set as realistic of present-day supercomputers); and a rather quicker rate $C_{ref}$=1 minute, which mimics the situation of future exascale.

Figure 7 depicts this information by means of contour maps. The information of a reference cluster with checkpoint rates $C_{ref}$=10min and $C_{ref}$=1min is used to fill up the plotted equations. Two power ratios (see eqn. (3)) $\rho$=5.5 & 7 are considered in the case of the energy policy as realistic [21]. Plots compare the results for three values of parameter $p$, namely $p$=0, 0.5 and 1. These figures of merit serve to quantify both cluster policies and, as a result, indicate the relative gain or loss incurred during applications execution in both scenarios of checkpoint rate ($C_{ref}$=10min and $C_{ref}$=1min).

Case $p$=0 in Figures 5 and 6 shows that an important saving of energy can be achieved, namely up to 30% for an execution time overhead of about 15% ($C_{ref}$=10min) and 12% ($C_{ref}$=1min). This saving corresponds to the visible maximum located in the plots ($N$~$3 \cdot 10^5$ nodes for $C_{ref}$=10min and $N$~$5 \cdot 10^6$ nodes for $C_{ref}$=1min), attained at a notably higher number of nodes in the exascale scenario.

The exascale scenario reveals that both ratios tend to one as the cluster size reaches its critical number of nodes $N_{critical}$~$7 \cdot 10^7$ nodes (that is, the maximum size which corresponds to enforce $T_{opt}$=0 in eqns. (1) and (2) under the respective policies at $p$=0). On the contrary, at $C_{ref}$=10 min the ratio of total execution time exhibits a flat value over a wide number of nodes, which indicates the insensitivity of the attained overhead with the size of the cluster. Interestingly, this flat zone of the plot is followed by a quick increase of the overhead in execution time, caused by the penalty of such a large $C_{ref}$ in eqn. (1).

For $p$=0.5 and $C_{ref}$=1min the maxima shift to higher $N$, with $N_{critical}$ ~$3 \cdot 10^9$ nodes. However, for $C_{ref}$=10min the behaviour reverses and the maxima shift to a lower $N$ (the critical size of the cluster shift to higher number of nodes: $N_{critical}$ ~$2 \cdot 10^7$ nodes). In terms of the ratios of execution time and energy consumption, this maxima shift which occurs for $C_{ref}$=10min implies a rather small impact on the figures of merit because the curves are rather flat in this zone.

It is visible that the energy saving and execution time overhead at $p$=0.5 are quite similar to their counterparts at $p$=0; but the advantage at $p$=0.5 is that there is a smaller rate of variation of the resource availability over the number of nodes and the cluster can be sized to achieve higher availability for the same number of nodes when its size is big enough.

According to subsection 3.1, case $p$=1 implies that the overhead due to $C$, $R$ and $D$ intervals scales as *~1/N*, so the figures of merit stagnate at constant values (i.e., for $C_{ref}$=1min the ratios result to be: Time($T_{opt}^{min-energy}$)/Time($T_{opt}^{min-time}$)~1.08 and Energy($T_{opt}^{min-time}$)/Energy($T_{opt}^{min-energy}$)~1.22). Interestingly, the energy saving and execution time overhead (which at $p$=0 show percentages of about 30% and 12%), exhibit for $p$=1 a lower bound: 22% and 8%, respectively, which quantifies the effect of the mentioned policies in a very optimistic scenario of checkpointing. The optimal checkpoint period is also plotted in Figures 5 and 6.

Another figure of merit is the resource availability $A_R$, defined as

$$A_R = 1 - \frac{\left(D_{ref}+R_{ref}+\omega C_{ref}\right)\left(\frac{N_{ref}}{N}\right)^p + \frac{T}{2}}{MTBF} \qquad (19)$$

The rate of $A_R$ over $N$ satisfying the criterion of optimal checkpoint period $T_{opt}$ yields

$$\frac{\partial A_R}{\partial N} = -\frac{1}{2}\frac{\partial}{\partial N}\left(\frac{T_{opt}}{MTBF}\right) - (1-p)\frac{D_{ref}+R_{ref}+\omega C_{ref}}{MTBF_{ref}}\frac{N_{ref}^{p-1}}{N^p} \qquad (20)$$

The sign of both terms on the right side of this equation is important in order to understand how the availability evolves over $N$. The sign of the first term depends on $N$. When $N$ is less than the number of nodes which yields $(T_{opt}/MTBF)_{max}$ (see Figures 5 and 6), its contribution becomes negative, otherwise positive. When it is positive, then it palliates to some extent the net availability drop over $N$ since it compensates part of the drop caused by the second term in eqn. (20), which is negative for $p \leq 1$.

Let $A_{R,ref}$ be the availability attained for a reference cluster of $N_{ref}$ nodes. The drop of the availability $\Delta A_R$ when shifting from $N_{ref}$ to certain $N^* > N_{ref}$ nodes can be evaluated directly from eqn. (19)

$$\Delta A_R = A_R^* - A_{R,ref} = \frac{1}{2}\left[\left(\frac{T_{opt}}{MTBF}\right)_{ref} - \left(\frac{T_{opt}}{MTBF}\right)^*\right] \qquad (21)$$

Availability drop is plotted in Figures 8 and 9 and can be used to guide the sizing of a cluster out of the reference data. Let assume $\Delta A_R = A^*_R - A_{R,ref}$ is the accepted net drop in availability for an enlarged version of the reference cluster. Then $N^*$ nodes are needed to match that constrain results from solving eqn. (21) for a prescribed checkpoint technology ($p$=0 if it is the same as in the reference cluster; or an improved one characterized by some $p$>0). In particular, for $p$=0 eqn. (21) gives

$$\frac{1}{2}\left(\frac{T_{opt}}{MTBF}\right)^* + \left(\frac{D_{ref}+R_{ref}+\omega C_{ref}}{MTBF_{ref}}\right)\frac{N^*}{N_{ref}} = -\Delta A_{R,ref} + \frac{1}{2}\left(\frac{T_{opt}}{MTBF}\right)_{ref} + \frac{D_{ref}+R_{ref}+\omega C_{ref}}{MTBF_{ref}} \quad (22)$$

where the right term is known and the equation can be solved for $N^*$. The expressions are valid for both the total execution time and energy consumption policies using the corresponding value of $T_{opt}$ from eqns. (1) and (2). For the energy consumption policy, results of $\rho$=5.5 and 7 are outlined in the figures.

As $N$ increases for a given $p$, the availability decreases at a rate which speeds down for $N$ greater than the number of nodes which yields $(T_{opt}/MTBF)_{max}$ and becomes zero at the end of the curves, which states the critical size $N_{critical}$ of the cluster. The comparison of the performance at both checkpoint rates shows that the slope of the availability curve is steeper over $N$ for $C_{ref}$=10min compared to $C_{ref}$=1min, which points out to the difficulties of enlarging current clusters by stacking additional hardware under the current checkpoint rate constrains. Besides, it is also visible that the starting-point availability (say, at $10^5$ nodes according to Figure 8) is worse for $C_{ref}$=10min. This is explained by the larger penalty caused by the $T_{opt}/MTBF$ contribution in eqn. (19), which is about three times its counterpart for $C_{ref}$=1min.

As expected, higher $p$ implies smaller $T_{opt}/MTBF$ in the region of large $N$, so a higher availability can be attained. Figures 5 and 6 shows that the curve which corresponds to the energy policy, irrespective of $\rho$, provides larger values of $T_{opt}$ than the execution time policy for all $p$ values. The comparison of both policies shows that the energy consumption policy gives a smaller availability over nodes as a result of its

smaller checkpoint frequency (higher $T_{opt}/MTBF$ in the entire range of number of nodes). Furthermore, the availability exhibits a higher rate of decay for the energy consumption policy at the beginning, which is explained by the behavior of the term $\partial(T_{opt}/MTBF)/\partial N$ in eqn. (20). On the contrary, the availability for the execution time policy shows a smooth drop over $N$ at the beginning but exhibits a rather speed up of the decay rate compared to the energy consumption policy.

The quantification of the availability drop reads out of these plots and the threshold line for, say, a 50% availability is drawn in Figure 9, which provides an estimation of the feasible maximum number or nodes for a given checkpoint $p$-value. The results have been obtained for multi-core architectures under the hypothesis $D_{ref} \sim C_{ref}/10$, which seems too optimistic for those architectures based on many-core chips. Whereas $D_{ref}$=1min or smaller sounds reasonable for multi-core based CPUs of the exascale era, it is not the case for current many-core chips (i.e. Intel KNLs/KNHs) where a more realistic value of the downtime interval is about 6 minutes according to recent experiments [36]. As a result, downtime interval $D_{ref}$ dominates in eqn. (1), hence a penalty is paid and the realizable cluster size shrinks (up to a 25% in current architectures and within a 4 to 5 times shrinkage using a number of computing units equivalent to the total number of cores using multi-core CPUs); the availability for the realizable cluster exhibits a significant drop as well. This is clearly shown in Figures 10 and 11.

## 5. CONCLUSIONS

This study assesses the impact of the optimal coordinated single-level checkpoint period and current modelling assumptions on the cluster performance under total execution time and energy consumption policies. A consequence of the prescribed checkpoint rate is the identification of a critical size of the cluster above which its availability is too poor.

To improve predictions of optimal checkpoint periods in forthcoming exascale computers, both high-order and non-blocking effects must be adequately included in the coordinated single-level checkpoint models. This is not easy because of the inherent mathematical difficulties of having into account all participating effects, but it

is important to know the real contribution of each effect on the cluster performance. While it makes little difference to perform second-order predictions of the optimal checkpoint frequency in current petascale platforms, it seems not to be the case in future exascale computers, as both effect will have similar impact on the performance.

Then, for accurate results, the development of second-order checkpoint formulations will lead to an improvement in their predictive capacity since the non-linearity over the checkpoint rate modifies the cluster availability in a significant amount (caused by the higher checkpoint frequency) when the number of nodes is huge. Consequently, its interplay with the acceleration of jobs executions of the users (because of using non-blocking checkpoint) must be correctly evaluated in the modelling.

In addition, the checkpoint overhead under criteria of minimizing total execution time and energy consumption has been compared for the case of homogeneous nodes clusters. Besides these performance metrics (which are driving vectors in the frame of cost and computational efficiency), another major factor to consider is cluster availability. The undertaken analysis revels that the total execution time policy delivers a higher availability of the resource for a prescribed cluster size and checkpoint technology (characterized by the checkpoint rate and its scalability in future platforms) than the energy consumption policy. Since resource availability is of paramount importance during the cluster lifetime, an adequate cluster sizing must be accomplished out of accurate estimations of the overhead provided by the checkpoint technology in use. To this respect, the variation of the performance metrics and availability over the number of nodes (for a given checkpoint characterization) provides guiding information in the decision making about how big a cluster might be.

## ACKNOWLEGMENT

This work was supported by the COST Action NESUS (IC1305) and partially funded by the Spanish Ministry of Economy and Competitiveness project CODEC2 (TIN2015-63562-R) with FEDER funds, the RICAP Network (517RT0529) with CYTED funds, and EU H2020 project HPC4E (grant agreement nº 689772).

## REFERENCES

[1] Cappello F., Geist A., Gropp W., Kale S., Kramer B., Snir M.: Towards Exascale Resilience: 2014 Update, *Int. J. Supercomputing Frontiers and Innovations* 1(1), pp. 5-28, 2014.

[2] Geist A., Reed D.A.: A Survey of High-performance Computing Scaling Challenges*, Int. J. of High Performance Computing Applications*, pp. 1-10, 2015.

[3] Shalf J., Dosanjh S., Morrison, J.: Exascale Computing Technology Challenges, *in Procs. Int. Conf. on High Computing for Computational Science -VECPAR 2010, Lecture Notes in Computer Science*, pp.1-25, vol. 6449, Springer, 2011.

[4] Geist A.: How to Kill a Supercomputer: Dirty Power, Cosmic Rays and Bad Solder - Will Future Exascale Supercomputers Be Able to Withstand the Steady Onslaught of Routine Faults?, *in IEEE Spectrum* (http://spectrum.ieee.org/computing/hardware), Feb 23, 2016.

[5] Schroeder B., Gibson G.A.: A Large-scale Study of Failures in High-performance Computing Systems, IEEE Transactions on Dependable and Secure Computing Vol.7 (4), pp. 337-350, 2010.

[6] Hacker Th.J., Romero F., Carothers C.D.: An Analysis of Clustered Failures on Large Supercomputing Systems, *J. Parallel and Distributed Computing*, 69, pp. 652-665, 2009.

[7] Hérault Th., Robert Y. (Eds.): Fault-tolerance Techniques for High-performance Computing, pp. 3-85, Switzerland, Springer, 2015.

[8] Hiroyama Sh., Dohi T., Okamura H.: Comparison of Aperiodic Checkpoint Placement Algorithms, in *Procs. Advanced Computer Science and Information Technology*, AST20120, Miyazaki, Japan, June 23-25, 2010.

[9] Buntinas D., Coti C., Hérault Th., Lemarinier p., Pilard L., Rezmerita a., Rodríguez E., Cappello F.: Blocking vs. Non-blocking Coordinated Checkpointing for Large-scale Fault Tolerant MPI, *Future Generation Computing Systems* 24(1), pp. 73-84, 2008.

[10] Naruse K., Umemura Sh., Nakagawa S.: Optimal Checkpointing Interval for Two-Level Recovery Schemes*, Computers and Mathematics with Applications* 51, pp. 371-376, 2006.

[11] Li H., Pang L., Wang Zh.: Two-level Incremental Checkpoint Recovery Scheme for Reducing System Total Overheads, *Plos One* 9(8), doi:19.1371/journal.pone.01045912014, 2014.

[12] Benoit A., Cavelan A., Robert Y., Sun H.: Optimal Resilience Patterns to Cope with Fail-stop and Silent Errors, *Research report RR-8786*, LIP-ENS Lyon <hal-01215857>, 2015.

[13] Di S., Robert Y., Vivien F., Cappello F.: Toward an Optimal Online Checkpoint Solution under a Two-level HPC Checkpoint Model*, IEEE Transactions on Parallel and Distributed Systems*, Vol.28 (1), pp. 244-259, 2017.

[14] Mohror K., Moody A., Bronevetsky G., Supinski B.R.: Detailed Modelling and Evaluation of a Scalable Multilevel Checkpointing System, *IEEE Transactions on Parallel and Distributed Systems* , Vol.25 (9), pp. 2255-2263, 2014.

[15] Ferreira K.B., Widener P., Levy S., Arnold D., Hoefler T.: Understanding the Effect of Communication and Coordination on Checkpointing at Scale, *Supercomputing Conference (SC14)*, Nov.16-21, New Orleans, USA, 2014.

[16] Bolsica G., Bouteiller A., Brunet E., Cappello F., Dongarra J., Guermouche A., Hérault Th., Robert Y., Vivien F., Zaidouni D.: Unified Model for Assessing Checkpointing Protocols at Extreme-scale, *Concurrency and Computation*: *Practice and Experience* 26, pp. 2772-2791, 2014.

[17] Bouguerra M.S., Gainaru A., Gómez L.B., Cappello F., Matsuoka S., Maruyama N.: Improving the Computing Efficiency of HPC Systems Using a Combination of Proactive and Preventive Checkpointing, *in Procs. IEEE 27th Int. Symp. on Parallel and Distributed Processing*, May 20-24 Boston (MA), USA, 2013.

[18] Gottumukkala N.R., Nassar R., Paun M., Leangsuksun Ch.B., Scott S.L.: Reliability of a System of K Nodes for High Performance Computing Applications, *IEEE Transactions on Reliability*, 59(1), pp. 162-169, 2010.

[19] Paun M., Naksinehaboon N., Nassar R., Leangsuksun Ch., Scott S.L., Taerat N.: Incremental Checkpoint Schemes for Weibull Failure Distribution, *Int. J. Foundations of Computer Science* 21(3), pp. 329-344, 2010.

[20] Bouguerra M.S., Gautier Th., Trystram D., Vincent J.M.: A Flexible Checkpoint/Restart Model in Distributed Systems, *PPAM 2009*, Part I, LNCS 6067, pp. 206-215, 2010.

[21] Aupy G., Benoit A., Hérault Th., Robert Y., Dongarra J.: Optimal Checkpointing Period: Time vs. Energy, *in Procs. Benchmarking and Simulation of High Performance Computer Systems*, Supercomputing Conference (SC13), Nov.17-22, Denver, USA, 2013.

[22] Vaidya N.H.: A Case for Two-level Recovery Schemes, *IEEE Transactions on Computers* 47(6), pp. 656-666, 1998.

[23] Daly J.T.: A Higher Order Estimate of the Optimum Checkpoint Interval for Restart Dumps, *Future Generation Computer Systems* 22, pp.303-312, 2006.

[24] Young W.: A First Order Approximation to the Optimum Checkpoint Interval, *Communications of the ACM*, Vol.17 (9), pp.530-531, 1974.

[25] Gelenbe E., Hernández M.: Optimum Checkpoints with Age Dependent Failures*, Acta Informatica* 27, pp. 517-531, 1990.

[26] Cox D.R., Miller H.D.: The Theory of Stochastic Processes, Chapman and Hall Ltd, London, 1972.

[27] Vaidya N.H.: Impact of Checkpoint Latency on Overhead Ratio of a Checkpointing Scheme, *IEEE Transactions on Computers*, 46(8), 1997.

[28] Vaidya N.H.: On Checkpoint Latency, Tex[12] Vaidya N.H.: A Case for Two-level Recovery Schemes, *IEEE Transactions on Computers* 47(6), pp. 656-666, 1998.
as A&M University, *Report 95015*, 1995.

[29] Yibei Ling, Jie Mi, Xiaola Lin: A Variational Calculus Approach to Optimal Checkpoint Placement, *IEEE Transactions on Computers*, 50(7), pp. 699-708, 2001.

[30] Ozaki T., Dohi T., Okamura H., Kaio N.: Distribution-free Checkpoint Placement Algorithms Based on Min-Max Principle, *IEEE Transactions on Dependable and Secure Computin*g, Vol.3, Nº2, 2006.

[31] Plank J.S., Elwasif W.R.: Experimental Assessment of Workstation Failures and Their Impact on Checkpointing Systems, *28th Annual Int. Symp. on Fault-tolerant Computing*, Munich, Germany, pp.48-57, 1998 (also as Univ. Tennessee Technical Report UT CS 97379, 1997).

[32] Liu Y., Nassar R., Leangsuksun Ch.B., Naksinehaboon N., Paun M., Scott S.L.: An Optimal Checkpoint/Restart Model for a Large Scale High Performance Computing System, *in Procs. IEEE Int. Symp. Parallel and Distributed Processing*, Miami, FL, pp. 1-9, 2008.

[33] Ozaki T., Dohi T., Kaio N.: Numerical Computation Algorithms for Sequential Checkpoint Placement, *Performance Evaluation* 66, pp. 311-326, 2009.

[34] Herault T., Robert Y. (Eds.): Fault-tolerance Techniques for High-Performance Computing, chapter 1, Computer Communications and Networks series, Springer, 2015.

[35] Kella O., Stadje W.: Superposition of Renewal Processes and an Application to Multi-server Queues, *Statistics & Probability Letters* 76, pp.1914-1924, 2006.

[36] Private communication, SLURM User Group Meeting, Sept. 26-27, Athens, Greece, 2016.

# LIST OF FIGURES

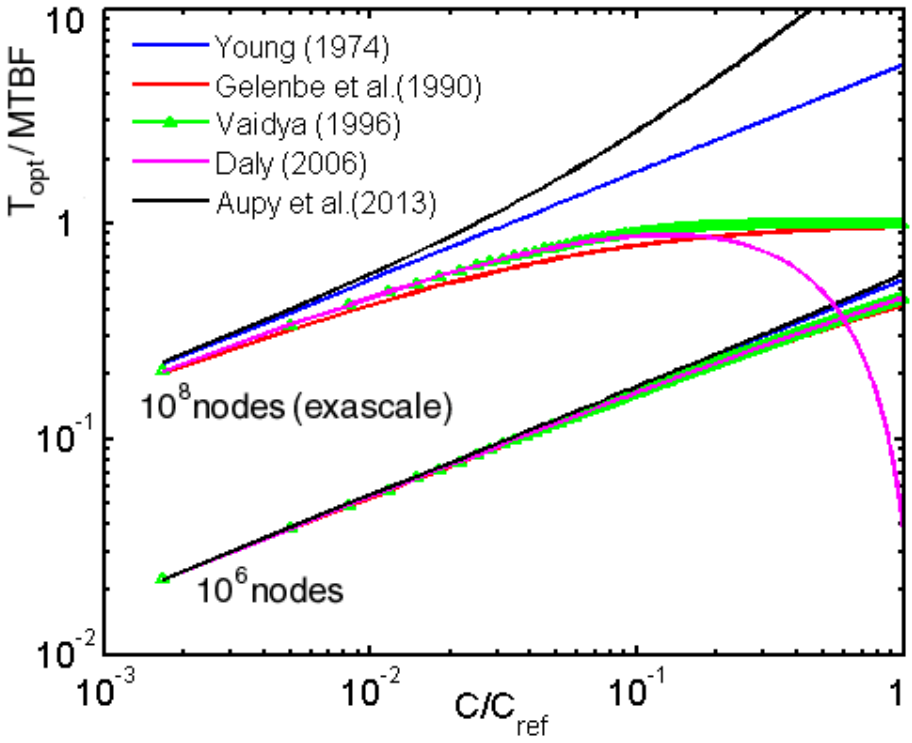


**Fig. 1:** Comparison of the nondimensional optimal checkpoint interval proposed by several authors in the literature (see references). All correspond to blocking checkpointing and $C_{ref}$=10min.

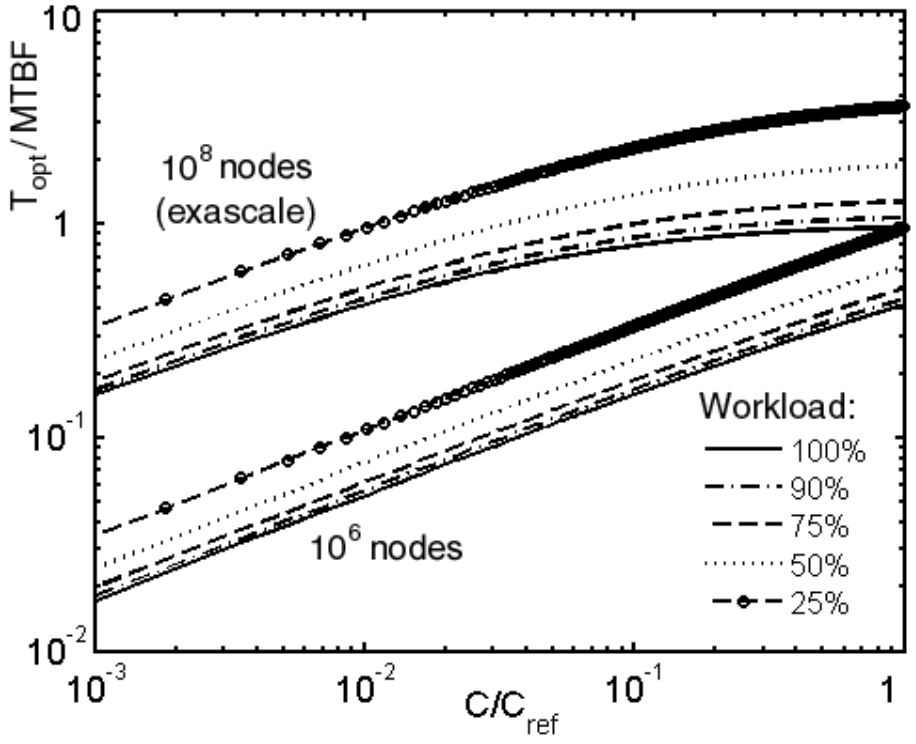


**Fig. 2:** Nondimensional optimal checkpoint interval proposed by Gelenbe *et al.* [25] where the effect of the % workload is shown for current ($10^6$ nodes) and future exascale ($10^8$ nodes) supercomputers. It is assumed Poisson distribution, negligible reloading time ($R$=0) of checkpointed data; and full processing of lost time interval (reference checkpoint interval: $C_{ref}$=10min).

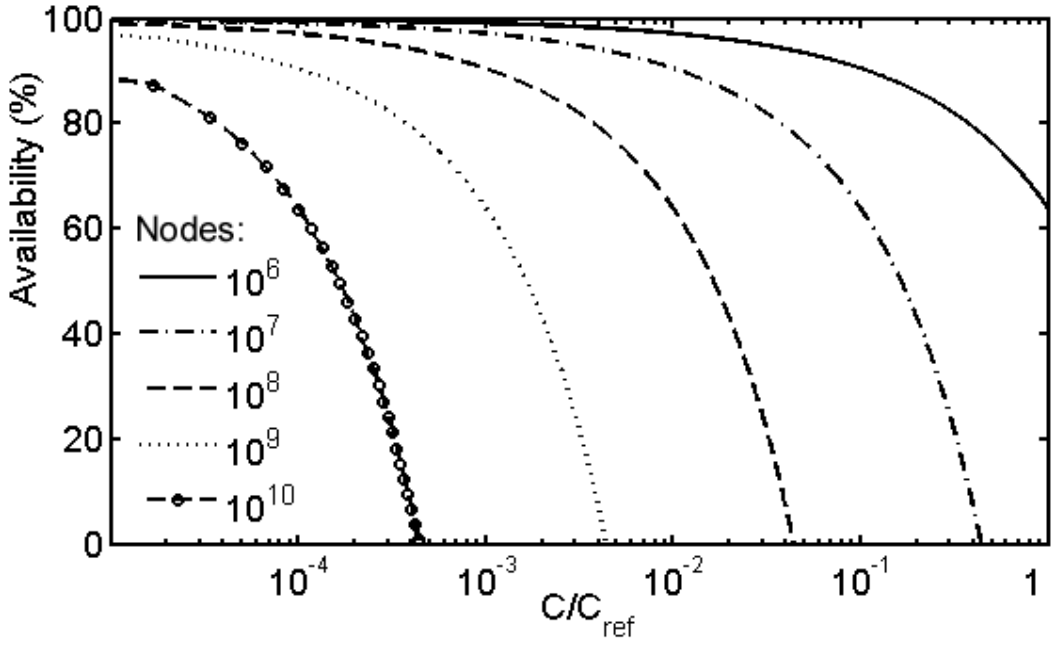


**Fig. 3:** Availability versus checkpointing duration for increasing facility scale (reference checkpoint interval: $C_{ref}$=10min).

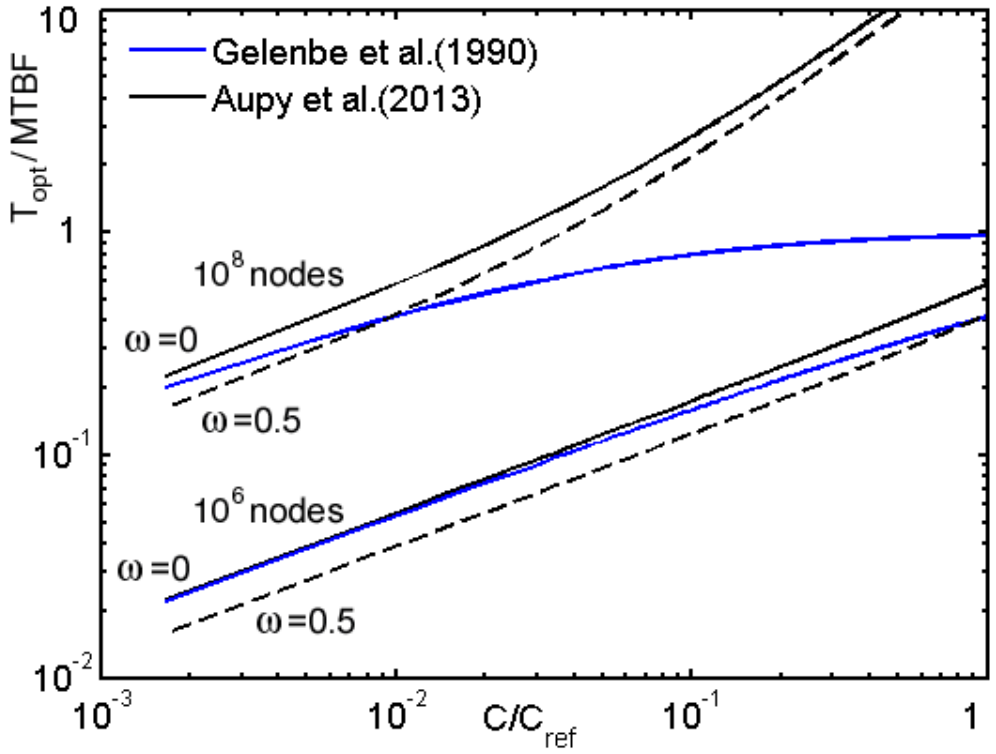


**Fig. 4:** Nondimensional optimal checkpoint interval according to Gelenbe *et al.* full-blocking model and Aupy *et al.* (this last for full- and half-blocking checkpointing: $\omega$=0 and $\omega$=0.5, respectively). Reference checkpoint interval: $C_{ref}$=10min.

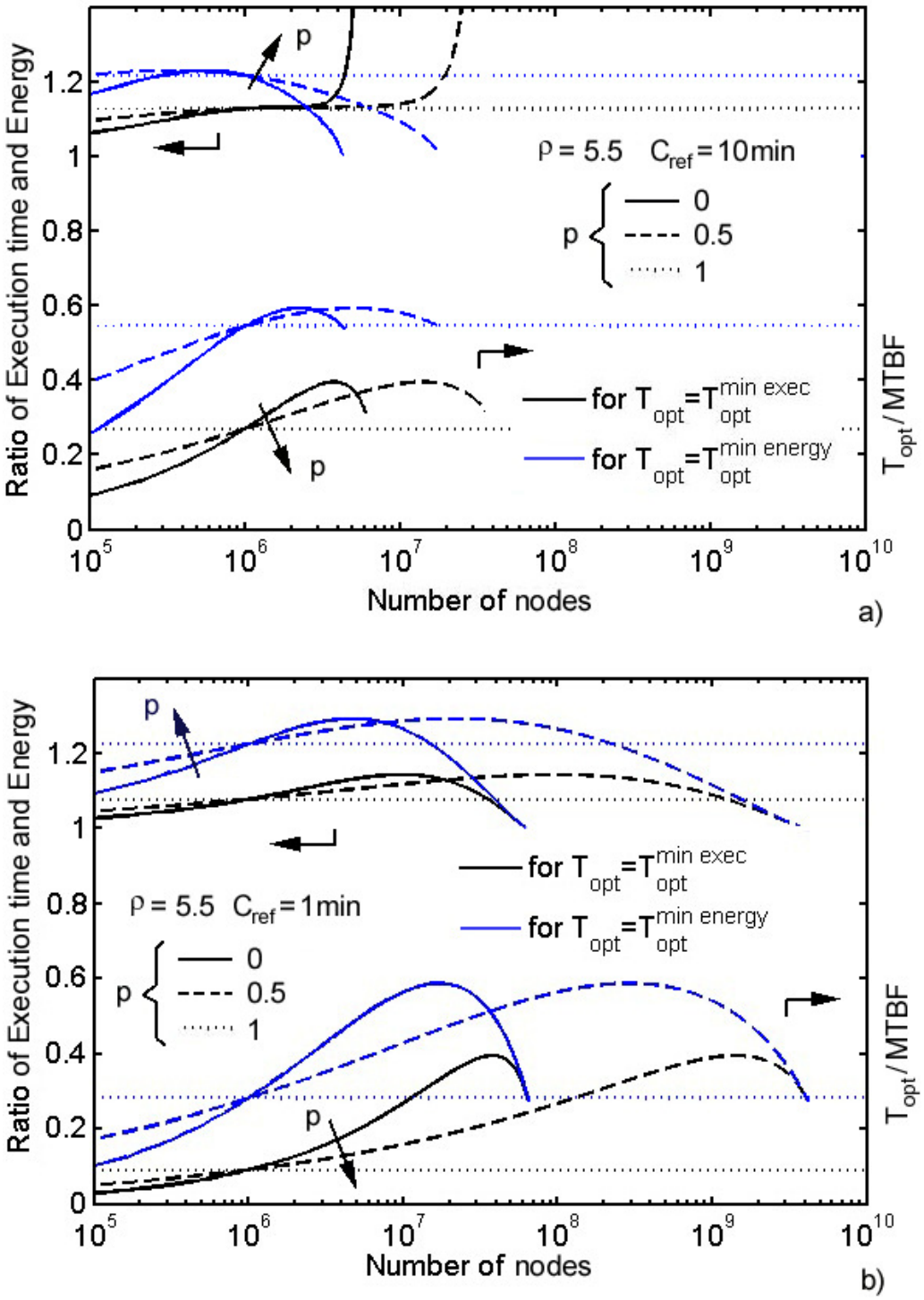


**Fig. 5:** Execution time and power consumption figure of merit comparison for $\rho$=5.5: ratio $T_{total}(T_{opt}t^{min\text{-}exec})/T_{total}(T_{opt}{}^{min\text{-}energy})$ (in black) and $E_{total}(T_{opt}t^{min\text{-}energy})/E_{total}(T_{opt}t^{min\text{-}exec})$ (in blue) is shown for three values of the checkpoint rate parameter ($p$=0, 0.5, 1) and $\omega$=0.5. Two scenarios of the checkpoint rate are depicted: a) Mimicking current checkpoint rate at $C_{ref}$=10min; and b) Improved (exascale) checkpoint rate at $C_{ref}$=1min. (nondimensional checkpoint period is shown in the lower portion of the graphs).

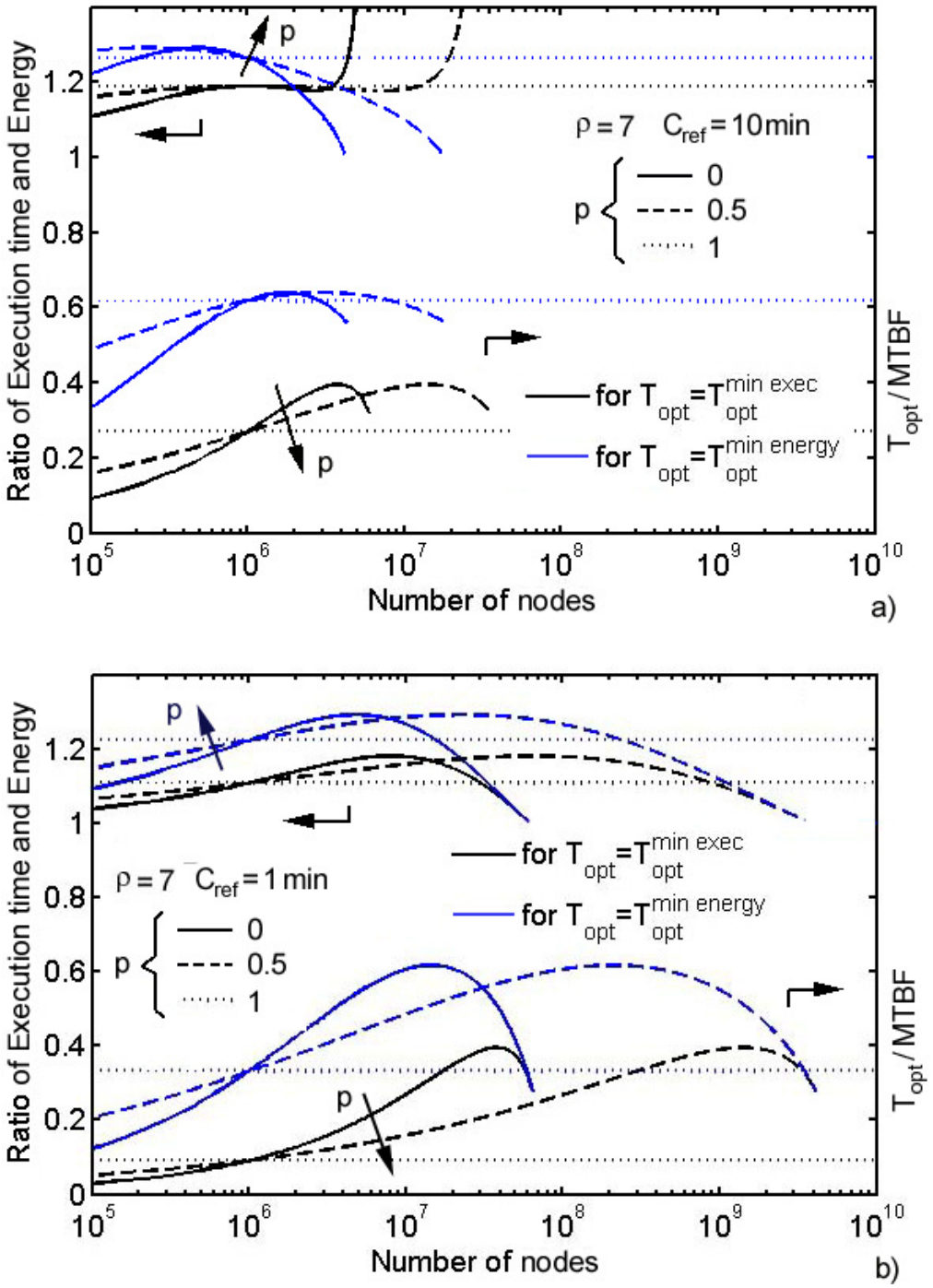


**Fig. 6:** Execution time and power consumption figure of merit comparison for $\rho$=7: ratio $T_{total}(T_{opt}{}^{min\text{-}exec})/T_{total}(T_{opt}{}^{min\text{-}energy})$ (in black) and $E_{total}(T_{opt}{}^{min\text{-}energy})/E_{total}(T_{opt}{}^{min\text{-}exec})$ (in blue) is shown for three values of the checkpointint rate parameter ($p$=0, 0.5, 1) and $\omega$=0.5. Two scenarios of the checkpoint rate are depicted: a) Mimicking current checkpoint rate at $C_{ref}$=10min; and b) Improved (exascale) checkpoint rate at $C_{ref}$=1min (nondimensional checkpoint period is shown in the lower portion of the graphs).

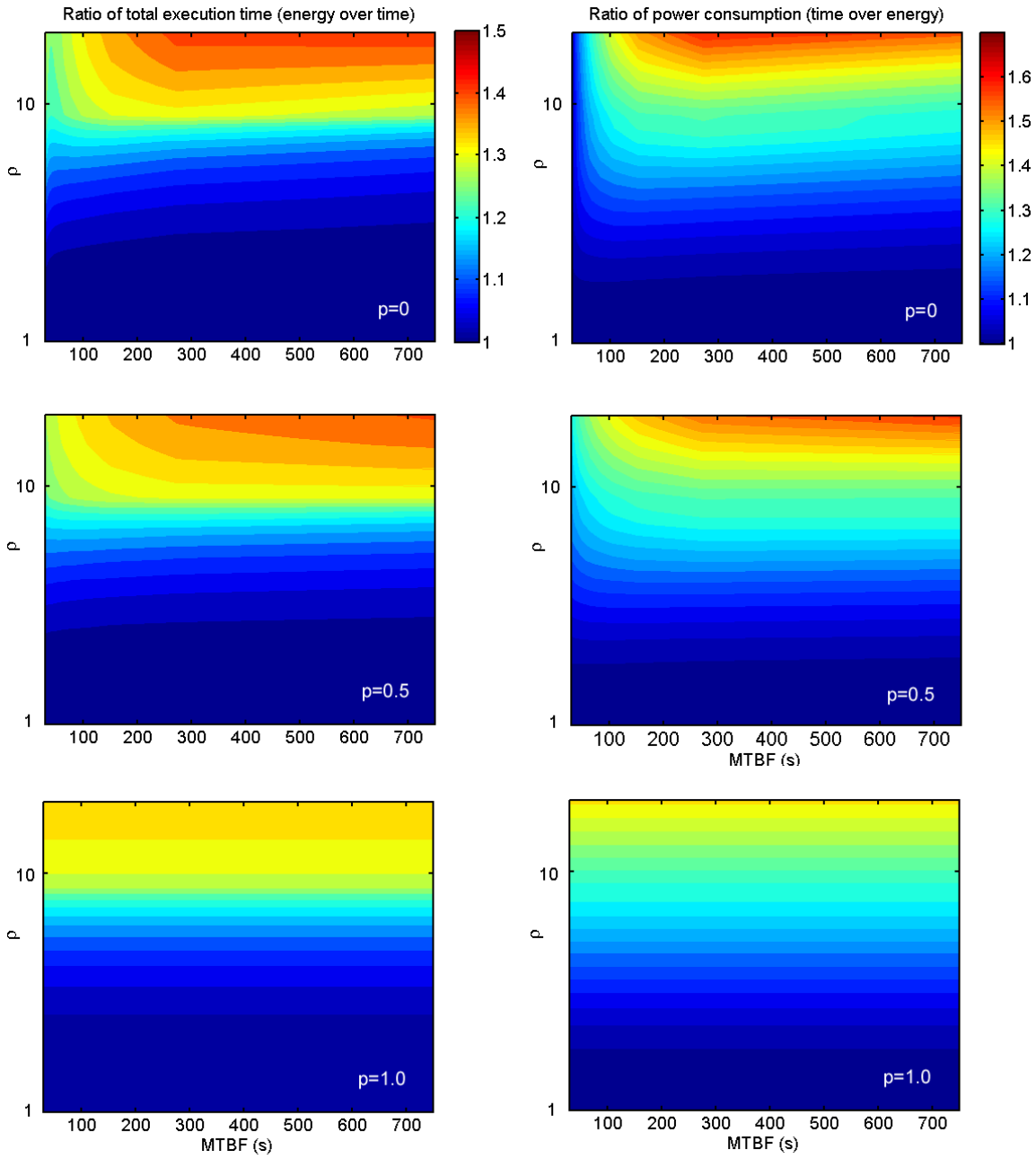


**Fig. 7:** Contour maps of the execution time and power consumption figures of merit: ratio $T_{total}(T_{opt}t^{min\text{-}exec})/T_{total}(T_{opt}^{min\text{-}energy})$ (left column) and $E_{total}(T_{opt}t^{min\text{-}energy})/E_{total}(T_{opt}t^{min\text{-}exec})$ (right) is shown for three values of the checkpoint rate parameter ($p$=0, 0.5, 1). Parameters: $\omega$=0.5, $C_{ref}$=10min.

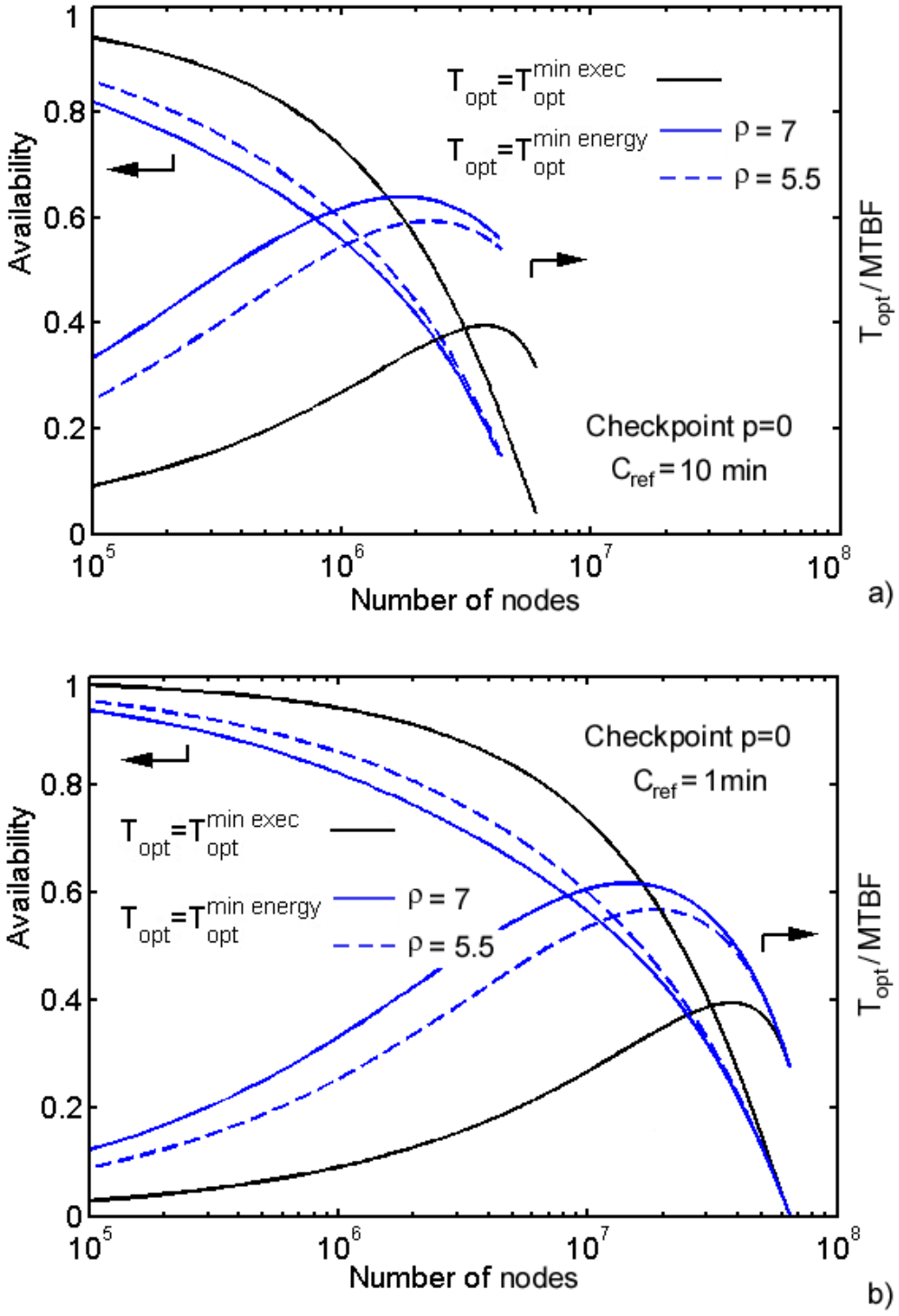


**Fig. 8:** Availability as a function of the cluster size and nondimensional optimal checkpoint period versus number of nodes depicted for checkpoint parameter $p$=0. Both total execution time and energy consumption policies are included in two scenarios of checkpoint rate: a) Mimicking current checkpoint rate at $C_{ref}$=10min; and b) Improved (exascale) checkpoint rate at $C_{ref}$=1min (results correspond to $\rho$=5.5 and 7 with $\omega$=0.5).

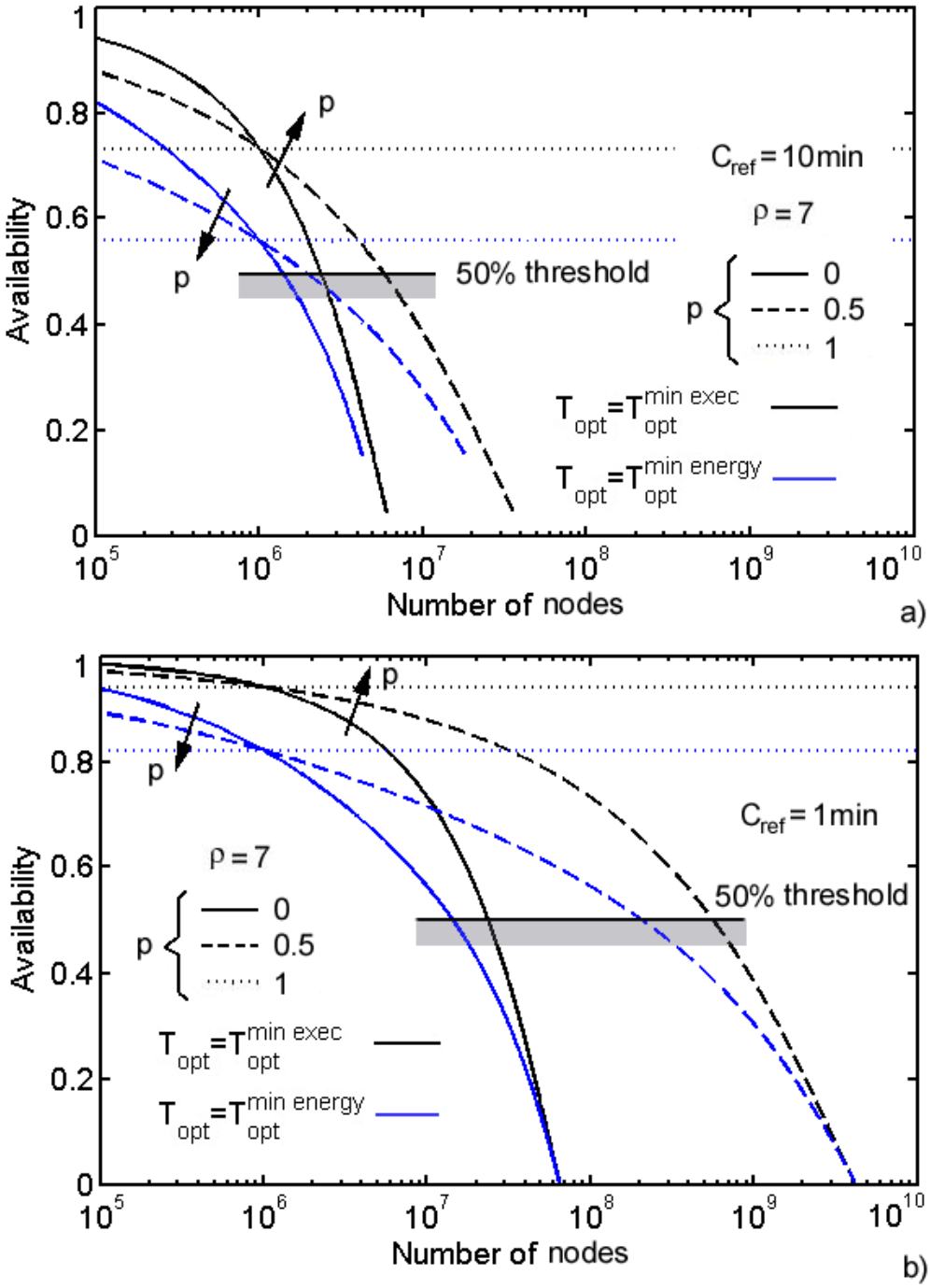


**Fig. 9:** Availability as a function of the cluster size for checkpoint parameters $p$=0, 0.5 and 1. Both total execution time and energy consumption policies are plotted (parameters: $\rho$=7, $\omega$=0.5 and two checkpoint rates: a) $C_{ref}$=10min; and b) $C_{ref}$=1min).

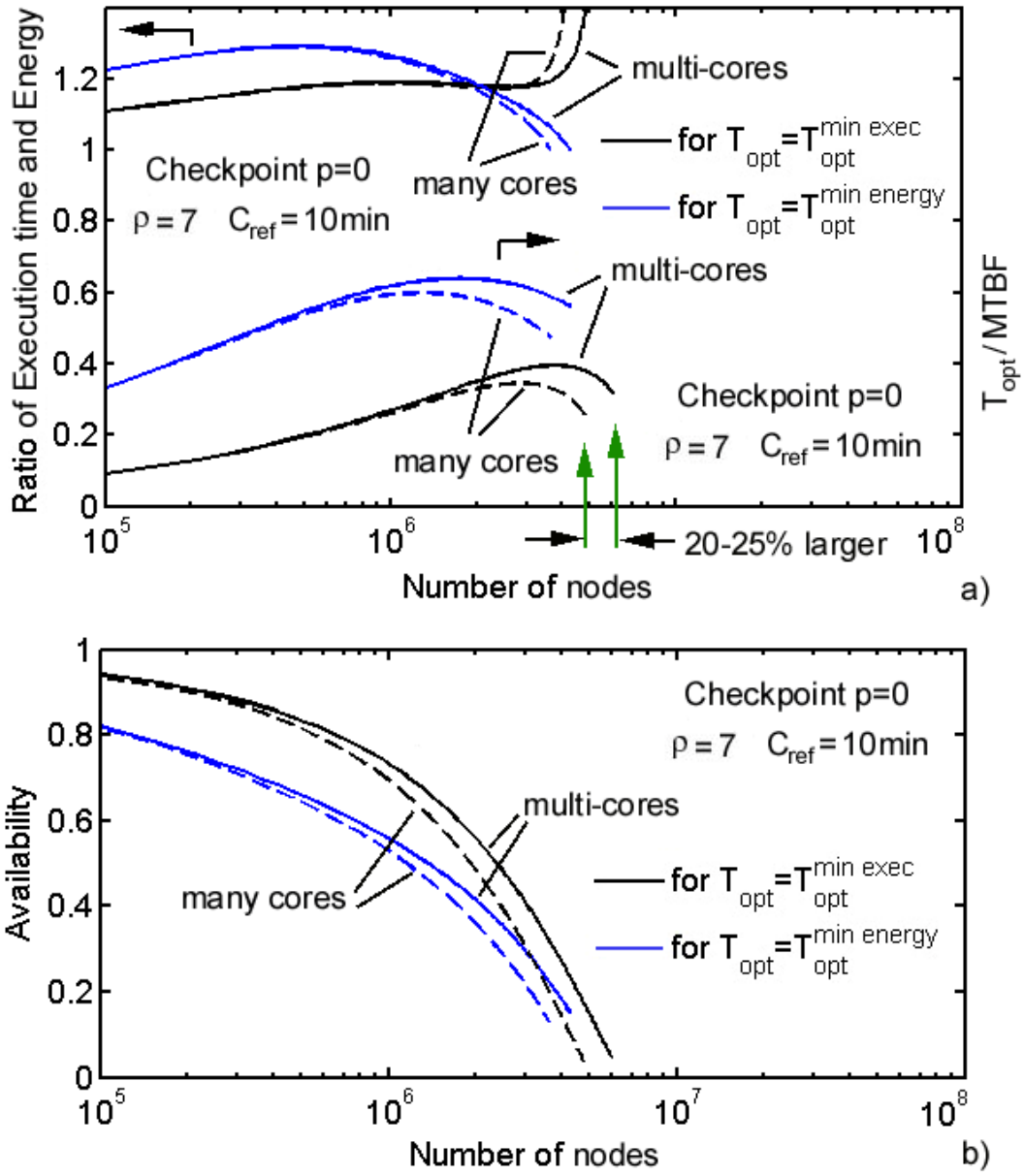


**Fig. 10:** Comparison of two architectures based on multi-cores characterized by $C_{ref}$=$R_{ref}$=10min and $D_{ref}$=1min; and many-cores ($C_{ref}$=$R_{ref}$=10min; $D_{ref}$=6min): a) Figures of merit of the total execution time and energy consumption ratios (above) and optimal checkpoint period (bottom); and b) Cluster availability over the number of nodes. The variation of the critical size of the cluster is indicated by green arrows (parameters: $\rho$=7 and $\omega$=0.5).

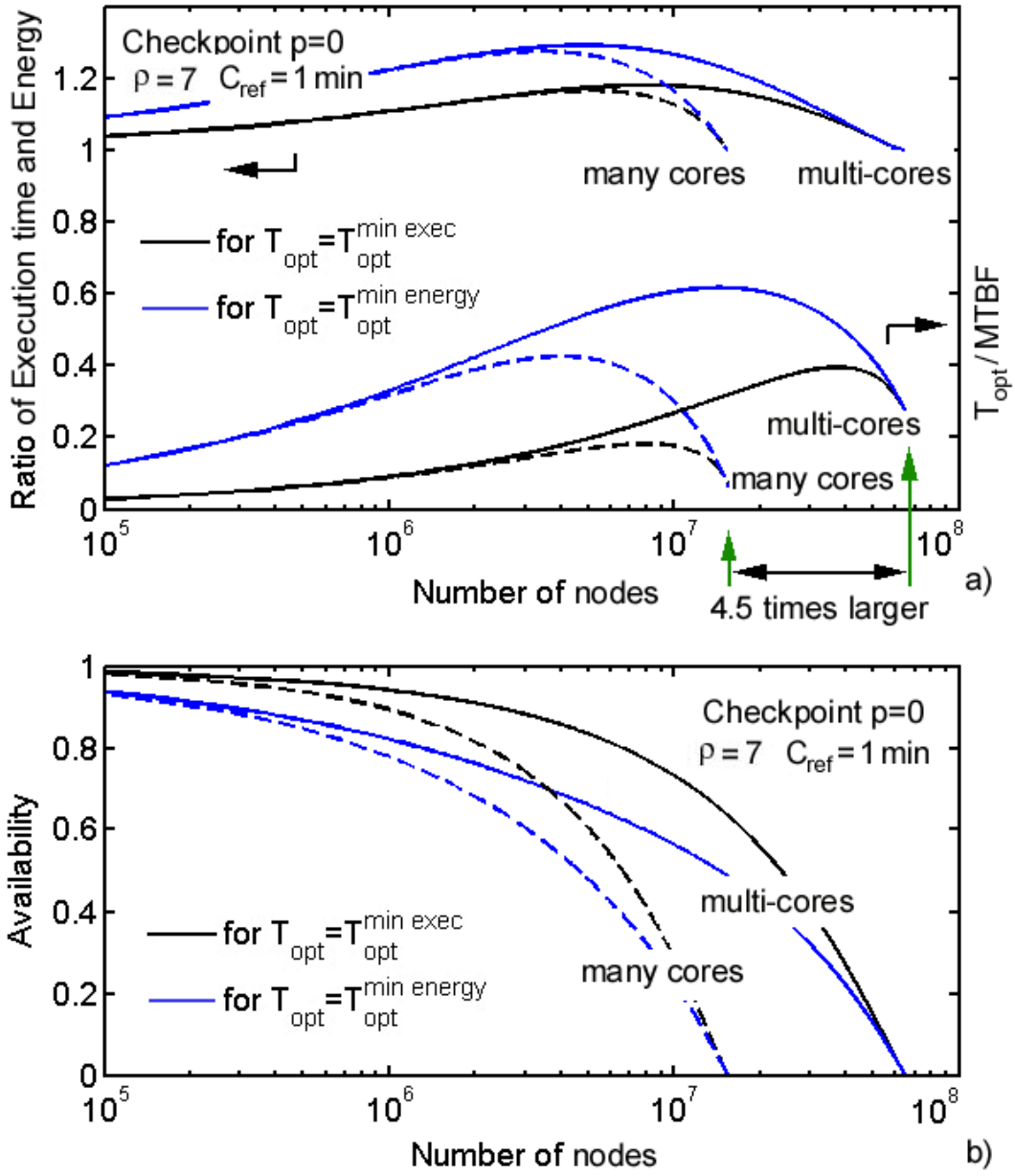


**Fig. 11:** Comparison of two architectures based on multi-cores characterized by $C_{ref}$=$R_{ref}$=1min and $D_{ref}$=0.1min; and many-cores ($C_{ref}$=$R_{ref}$=1min; $D_{ref}$=6min): a) Figures of merit of the total execution time and energy consumption ratios (above) and optimal checkpoint period (bottom); and b) Cluster availability over the number of nodes. The variation in critical size of the cluster is indicated by green arrows (parameters: $\rho$=7 and $\omega$=0.5).