On the Optimum Checkpoint Selection Problem*

Sam Toueg
Özalp Babaoğlu

TR 83-546
February 1983
(Revised November 1983)

Department of Computer Science
Cornell University
Ithaca, New York 14853

*This research was supported in part by the National Science Foundation under Grants MCS 81-03605 and MCS 82-10356.

ON THE OPTIMUM CHECKPOINT SELECTION PROBLEM*

SAM TOUEG† AND ÖZALP BABAOĞLU†

Abstract. We consider a model of computation consisting of a sequence of n tasks. In the absence of failures, each task i has a known completion time t_i. Checkpoints can be placed between any two consecutive tasks. At a checkpoint, the state of the computation is saved on a reliable storage medium. Establishing a checkpoint immediately before task i is known to cost s_i. This is the time spent in saving the state of the computation. When a failure is detected, the computation is restarted at the most recent checkpoint. Restarting the computation at checkpoint i requires restoring the state to the previously saved value. The time necessary for this action is given by r_i. We derive an $O(n^3)$ algorithm to select out of the $n-1$ potential checkpoint locations those that result in the smallest expected time to complete all the tasks. An $O(n^2)$ algorithm is described for the reasonable case where $s_i > s_j$ implies $r_i \geqslant r_j$. These algorithms are applied to two models of failure. In the first one, each task i has a given probability p_i of completing without a failure, i.e., in time t_i. Furthermore, failures occur independently and are detected at the end of the task during which they occur. The second model admits a continuous time failure mode where the failure intervals are independent and identically distributed random variables drawn from any given distribution. In this model, failures are detected immediately. In both models, the algorithm also gives the expected value of the overall completion time and we show how to derive all the other moments.

Key words. fault-tolerance, checkpoint, rollback-recovery, discrete optimization, renewal process

1. Introduction. A variety of hardware and software techniques have been proposed to increase the reliability of computing systems that are inherently unreliable. One such software technique is *rollback-recovery*. In this scheme, the program is *checkpointed* from time to time by saving its state on secondary storage and the computation is restarted at the most recent checkpoint after the detection of a failure [6]. Between the times when the failure is detected and the computation is restarted, the computation must be *rolled back* to the most recent checkpoint by restoring its state to the saved value. Obviously, rollback-recovery is an effective method only against transient failures. Examples of such failures are temporary hardware malfunctions, deadlocks due to resource contention, incorrect human interactions with the computation, and other external factors that can corrupt the computation's state. Persisting failures will block the computation no matter how many times it is rolled back. The ability to detect failures is an essential part of any fault-tolerance method including rollback-recovery. Examples of such failure detection methods are integrity assertion checking [8] and fail-stop processors [10].

In the absence of checkpoints, the computation has to be restated from the beginning whenever a failure is detected. It is clear that with respect to many objectives such as minimum completion time, minimum recovery overhead, maximum throughput, etc., the positioning of the checkpoints involves certain tradeoffs. A survey of an analytical framework for resolving some of these tradeoffs is presented by Chandy [1]. Young [11] and Chandy *et al* [3] addressed the problem of finding the checkpoint interval so as to minimize the time lost due to recovery for a never-ending program subject to failures constituting a Poisson process. Gelenbe and Derochette [5] and Gelenbe [4] have generalized this result to allow the possibility of external requests arriving during the establishment of a checkpoint or rollback-recovery.

* Received by the editors March 9, 1983, and in revised form September 13, 1983. This article was typeset by the authors at Cornell University on a UNIX™ system. Final copy was produced on November 3, 1983.

† Department of Computer Science, Cornell University, Ithaca, New York 14853. This research was supported in part by the National Science Foundation under Grants MCS 81-03605 and MCS 82-10356.

These requests are queued and serviced later. Consequently, these results for the optimum checkpoint intervals with respect to maximizing system availability and minimizing response time reflect the dependence on the rate of requests. More recently, Koren *et al* [7] have derived expressions for optimum parameters in a system employing a combination of instruction retry and rollback-recovery.

In the previous work described above, expressions for the optimum checkpoint interval were derived with the assumption that checkpoints could be placed at arbitrary points in the program at a cost (as measured in units of time) that is independent of their position. Recall that establishing a checkpoint implies the saving of the current program state on secondary storage. As the minimum amount of data required to specify the state of a program can vary greatly over time, it is unrealistic to assume that the time necessary to write it to secondary storage is a constant. Furthermore, some programs cannot be blocked to create a checkpoint during the execution of certain intervals. The transaction processing periods of a database system are examples of such intervals. Whether due to prohibitive costs or other practical considerations, most computations display only a discrete set of points where checkpoints can be placed. Informally, we will view these computations as a sequence of *tasks* such that the program state between two consecutive tasks is "compact" (i.e., incurs a reasonable cost to save). In other words, task boundaries define potential checkpoint locations. Statically, such a computation can be represented as a directed graph where the vertices denote the tasks and an edge (i,j) exists if and only if task i may be followed by task j in some execution. This computation model was used by Chandy and Ramamoorthy in the optimum checkpoint selection problem where the objective was to minimize the maximum and expected time spent in saving states [2].

In this paper, we model the execution of a program as a linear sequence of tasks and consider the optimum checkpoint selection problem with respect to minimizing the expected total execution time of the program subject to failures. In the next section, we introduce the formal program model along with the input parameters to the problem. § 3 describes an algorithm based on dynamic programming that generates the set of checkpoint locations so as to minimize the expected completion time. § 4 gives an improved version of this algorithm that is applicable when there is a certain relation between the costs for establishing checkpoints and the costs of rolling back the computation. In § 5, we present two possible failure models to which the algorithm could be applied. Solutions to some extensions of the original problem are presented in § 6. A discussion of the results concludes the paper.

2. The model of computation. Assume that a computation consists of the sequential execution of n tasks where a task may be a program, procedure, function, block, transaction, etc. depending on the environment. For our purposes, any point in the execution where the computation is allowed to block and where its state can be represented by a reasonable amount of information can delimit a task. Clearly, the decomposition of the computation into tasks is not unique -- any convenient one will suffice.

Let t_i denote the time required to complete task i in the absence of failures. We assume that these times are deterministic quantities and are known for all of the tasks. The boundary between two consecutive tasks $i-1$ and i determines the ith candidate checkpoint location. The *setup cost*, s_i, is defined to be the time required to establish a checkpoint at location i. This is the time necessary to save the state of the computation on secondary storage as it exists just before task i is executed. After a failure is detected, the state of the computation is restored to that of the most recent checkpoint. The *rollback cost*, r_i, is defined to be the time required to roll back the computation to the checkpoint at location i. After the rollback, the computation resumes with the execution of task i. We assume that a checkpoint is always established just before task 1. Initially, we also assume that the checkpoint setups and rollbacks are failure-free. We relax these last assumptions in § 6.

The optimization problem can now be stated as follows: Given t_i, s_i and r_i for $i=1,2,...,n$ and a suitable failure model, select the subset of the $n-1$ potential checkpoint locations such that the resulting expected total completion time (including the checkpoint setup and the rollback-recovery times) for the computation is minimized. The objective function we have selected is a reasonable one for computations that govern time-critical applications such as chemical process control, air traffic control, etc., in the presence of failures.

The following definitions will be used in the subsequent sections. Let $[i,j]$ denote the sequence of tasks i, $i+1,..., j$ where $j \geqslant i$. Note that $[1,n]$ is the entire computation. Let $T_{i,j}^m$ denote the minimum expected execution time for $[i,j]$ over all the possible checkpoint selections in $[i,j]$ with m or fewer checkpoints. Clearly, $T_{i,j}^0$ is the expected execution time of $[i,j]$ without any checkpoints. Among all the checkpoint selections in $[i,j]$ that achieve $T_{i,j}^m$, we consider those with the minimum number of checkpoints. These selections are called *m-optimal solutions* for $[i,j]$, and we denote them by $\mathbf{L}_{i,j}^m$. Note that a m-optimal solution $\mathbf{L}_{i,j}^m$ for $[i,j]$ contains at most m checkpoints. If $\mathbf{L}_{i,j}^m$ contains no checkpoints (i.e., it is the empty selection of checkpoints), it is written as $\mathbf{L}_{i,j}^m = <>$. If $\mathbf{L}_{i,j}^m$ contains k checkpoints $(1 \leqslant k \leqslant m)$, we represent it as the ordered sequence of the selected checkpoint locations $\mathbf{L}_{i,j}^m = <u_1, u_2, \ldots, u_k>$, where $i < u_1 < u_2 < \cdots < u_k \leqslant j$. The *rightmost checkpoint location* of $\mathbf{L}_{i,j}^m = <u_1, u_2, \ldots, u_k>$ is u_k, and the rightmost checkpoint location of $\mathbf{L}_{i,j}^m = <>$ is i. There may be more than one m-optimal solution for $[i,j]$. From now on, we will consider only those m-optimal solutions $\mathbf{L}_{i,j}^m$ that satisfy the following additional requirement: Either $\mathbf{L}_{i,j}^m = <>$ or the rightmost checkpoint location of $\mathbf{L}_{i,j}^m$ is greater than or equal to the rightmost checkpoint location of any other m-optimal solution for $[i,j]$. Henceforth, we reserve the notation $\mathbf{L}_{i,j}^m$ to denote only those m-optimal solutions for $[i,j]$ that satisfy this additional requirement.

The next section describes an algorithm to determine an optimum checkpoint selection $\mathbf{L}_{1,n}^{n-1}$ and the corresponding minimum expected execution time $T_{1,n}^{n-1}$ for a given problem.

3. The basic algorithm. Consider $T_{1,j}^k$ and $T_{1,j}^{k-1}$ for some k and j such that $k \geqslant 1$ and $j \geqslant 2$. Note that either $T_{1,j}^k = T_{1,j}^{k-1}$ or $T_{1,j}^k < T_{1,j}^{k-1}$. Suppose $T_{1,j}^k < T_{1,j}^{k-1}$. In this case, any k-optimal solution for $[1,j]$ must contain exactly k checkpoints. Let h be the location of the rightmost checkpoint of a k-optimal solution for $[1,j]$. We must have

$$T_{1,j}^k = T_{1,h-1}^{k-1} + T_{h,j}^0 + s_h .$$

That is, up to $k-1$ checkpoints are optimally established in $[1,h-1]$ and a checkpoint is established at location h. No checkpoints are established in $[h,j]$. Let $\mathbf{L}_{1,h-1}^{k-1}$ be any $(k-1)$-optimal solution for $[1,h-1]$. Then

$$\mathbf{L}_{1,j}^k = \mathbf{L}_{1,h-1}^{k-1} \,||\, <h>$$

must be a k-optimal solution $[1,j]$ (the $||$ operator denotes concatenation of sequences, i.e., $<u_1,...,u_n> || <v> = <u_1,...,u_n,v>$).

for $i \leftarrow 1$ **until** n **do**
  for $j \leftarrow i$ **until** n **do** *compute* $T_{i,j}^{0}$;

for $k \leftarrow 1$ **until** $n-1$ **do**
  begin
    $T_{1,1}^{k} \leftarrow T_{1,1}^{0}$
    $\mathbf{L}_{1,1}^{k} \leftarrow < >$
  end;

for $k \leftarrow 1$ **until** $n-1$ **do**
  for $j \leftarrow n$ **step** -1 **until** 2 **do**
    begin
      $T \leftarrow \underset{1<i\leqslant j}{\mathrm{MIN}} \, (T_{1,i-1}^{k-1} + T_{i,j}^{0} + s_i)$
      Let h be the largest minimizing index above
      if $T < T_{1,j}^{k-1}$ **then do**
        begin
          $T_{1,j}^{k} \leftarrow T$
          $\mathbf{L}_{1,j}^{k} \leftarrow \mathbf{L}_{1,h-1}^{k-1} \,||\, <h>$
        end
      else do
        begin
          $T_{1,j}^{k} \leftarrow T_{1,j}^{k-1}$
          $\mathbf{L}_{1,j}^{k} \leftarrow \mathbf{L}_{1,j}^{k-1}$
        end;
    end;

FIG. 1. *Dynamic programming algorithm for computing the optimum checkpoint selection (and the corresponding expected execution time) for a n-task computation.*

From these observations, it is clear that we can compute $T_{1,j}^{k}$ and $\mathbf{L}_{1,j}^{k}$ as follows. Let

$$T = \underset{1<i\leqslant j}{\mathrm{MIN}} \, (T_{1,i-1}^{k-1} + T_{i,j}^{0} + s_i)$$

and let h be the largest index such that $T = T_{1,h-1}^{k-1} + T_{h,j}^{0} + s_h$. We must have $T_{1,j}^{k} = \mathrm{MIN}(T, T_{1,j}^{k-1})$. If $T_{1,j}^{k} = T_{1,j}^{k-1}$ then we have $\mathbf{L}_{1,j}^{k} = \mathbf{L}_{1,j}^{k-1}$; otherwise $\mathbf{L}_{1,j}^{k} = \mathbf{L}_{1,h-1}^{k-1} \,||\, <h>$.

We just showed that if $T_{i,j}^{0}$, $T_{1,i}^{k-1}$ and $\mathbf{L}_{1,i}^{k-1}$ are computed first (for all i's and j's such that $1\leqslant i\leqslant j$), then we can also derive $T_{1,j}^{k}$ and $\mathbf{L}_{1,j}^{k}$. This suggests a dynamic programming algorithm to compute $T_{1,n}^{n-1}$ and $\mathbf{L}_{1,n}^{n-1}$. The algorithm is described in detail (in "Pidgin Algol") in Fig. 1.

Note that the underlying probabilistic failure model does not explicitly appear in the algorithm. It is implicitly used only during the initialization of the algorithm, when the $T_{i,j}^{0}$'s are computed (for all i's and j's such that $1\leqslant i\leqslant j\leqslant n$). Therefore, the same algorithm can be applied to any underlying failure model such that the $T_{i,j}^{0}$'s can be computed.

If we exclude the computation of the $T_{i,j}^0$'s, the time complexity of this algorithm is $O(n^3)$. In fact, the inner loop is executed $O(n^2)$ times, and the most time-consuming operation in this inner loop is the $\mathrm{MIN}_{1<i\leqslant j}$ operation. Since $j = O(n)$, this operation requires $O(n)$-time.

If we assume that the setup costs and the rollback costs are related as described in the next section, then we can reduce the complexity of the algorithm to $O(n^2)$.

We finally note that computing all the $T_{i,j}^0$'s takes $O(n^2)$-time in the two failure models that we consider in § 5. Therefore, with these models, the overall complexity of the algorithms described in Fig. 1 and in the next section is $O(n^3)$ and $O(n^2)$, respectively.

4. An improved algorithm for a restricted model. We assume the following relation between the setup costs and the rollback costs:

For any two checkpoint locations i and j, if $s_i > s_j$ then $r_i \geqslant r_j$.

Note that if there is a non-decreasing function f relating all the rollback costs to the setup costs, i.e., $r_i = f(s_i)$, then the above relation is satisfied. In particular, it is satisfied if for all i we have $r_i = \alpha s_i + \beta$, for some constant $\alpha \geqslant 0$ and $\beta \geqslant 0$. It is also satisfied if all the setup costs or all the rollback costs are equal.

We further assume that the failure model is such that augmenting any segment cannot decrease the probability of a failure occurring in that segment. Formally, for all i,j and k such that $1 \leqslant i \leqslant j \leqslant k < n$, we have $p_{i,j} \geqslant p_{i,k}$, where $p_{i,j}$ denotes the probability that no failures occur during the execution of $[i,j]$. With these assumptions, we can prove the following two theorems.

THEOREM 1. *Suppose the m-optimal solutions for $[i,j]$ are such that their rightmost checkpoint location is k, $i \leqslant k \leqslant j$. Then, for any $p>j$ the m-optimal solutions for $[i,p]$ are such that their rightmost checkpoint location is some h, $h \geqslant k$.*

THEOREM 2. *Suppose the m-optimal solutions for $[i,j]$ are such that their rightmost checkpoint location is k, $i \leqslant k \leqslant j$. Then the $(m+1)$-optimal solutions for $[i,j]$ are such that their rightmost checkpoint location is some h, $h \geqslant k$.*

The proofs of these two theorems can be found in the Appendix. From these two theorems we can immediately derive the following corollary.

COROLLARY. *Let a, b, and c be the rightmost checkpoint locations of the $(k-1)$-optimal solutions for $[1,j]$, the k-optimal solutions for $[1,j]$, and the k-optimal solutions for $[1,j+1]$, respectively $(1<j,\ k<n)$. We have $a \leqslant b \leqslant c$.*

Using this corollary, we can speed up the basic algorithm described in Fig. 1 by a factor of $O(n)$. The main idea is to restrict the range of indexes scanned by the MIN operation in the algorithm's inner loop. Suppose we want to compute $T_{1,j}^k$ and we already know some $\mathbf{L}_{1,j}^{k-1}$ and $\mathbf{L}_{1,j+1}^k$. Let $l_{1,j}^{k-1}$ and $l_{1,j+1}^k$ be the rightmost checkpoint locations of $\mathbf{L}_{1,j}^{k-1}$ and $\mathbf{L}_{1,j+1}^k$, respectively. From the corollary, we know that the rightmost checkpoint location i of any k-optimal solution of $[1,j]$ is such that $l_{1,j}^{k-1} \leqslant i \leqslant l_{1,j+1}^k$. Therefore, we can compute $T_{1,j}^k$ as follows:

$$T \leftarrow \mathop{\mathrm{MIN}}_{l_{1,j}^{k-1} \leqslant i \leqslant l_{1,j+1}^k} (T_{1,i-1}^{k-1} + T_{i,j}^0),$$

$$T \leftarrow \mathrm{MIN}(T, T_{1,j}^{k-1}).$$

Note that, in the algorithm described in Fig. 1, when $T_{1,j}^k$ is computed, $\mathbf{L}_{1,j}^{k-1}$ and $\mathbf{L}_{1,j+1}^{k}$ have already been derived in some earlier steps of the algorithm. Therefore, $l_{1,j}^{k-1}$ and $l_{1,j+1}^{k}$ are known at that point and they can be used to compute $T_{1,j}^k$ as shown above; $\mathbf{L}_{1,j}^k$ is then derived as in the basic algorithm.

In the basic algorithm of Fig. 1, the range of indexes scanned by the MIN operation to compute $T_{1,j}^k$ was $1<i\leqslant j$. With the algorithm modification that we just described, the range is $l_{1,j}^{k-1}\leqslant i\leqslant l_{1,j+1}^{k}$. We now show that this modification results in an $O(n^2)$ algorithm[1]. Note first that a $\mathrm{MIN}_{a\leqslant i\leqslant b}$ operation in this algorithm takes $b-a+1$ steps, i.e., $O(b-a)$ time. Let q, $0\leqslant q\leqslant n-1$, be a fixed positive constant. Let $T(q)$ be the following set

$$T(q)=\{T_{1,j}^k \mid 2\leqslant j\leqslant n,\ 1\leqslant k\leqslant n-1,\text{ and } j-k=q\}.$$

Consider the time taken by all the MIN operations performed while computing all the elements of $T(q)$.

When $T_{1,q+1}^1$ is computed, the range of indexes is $1\leqslant i\leqslant l_{1,q+2}^1$.
When $T_{1,q+2}^2$ is computed, the range of indexes is $l_{1,q+2}^1\leqslant i\leqslant l_{1,q+3}^2$.
When $T_{1,q+3}^3$ is computed, the range of indexes is $l_{1,q+3}^2\leqslant i\leqslant l_{1,q+4}^3$.
...
When $T_{1,n-1}^{n-q-1}$ is computed, the range of indexes is $l_{1,n-1}^{n-q-2}\leqslant i\leqslant l_{1,n}^{n-q-1}$.
When $T_{1,n}^{n-q}$ is computed, the range of indexes is $l_{1,n}^{n-q-1}\leqslant i\leqslant n$.

By summing up the number of computation steps needed by all these MIN operations, we get a total of $n+(n-q)$ steps. Therefore, for any fixed q $(0\leqslant q\leqslant n-1)$, the MIN operations performed while computing all the elements of $T(q)$ take a total of $O(n)$ time. By considering all the n possible values of q, we conclude that the MIN operations performed while computing all the $T_{1,j}^k$'s such that $j-k\geqslant 0$ take a total of $O(n^2)$ time. Similarly, we can show that the MIN operations performed while computing all the $T_{1,j}^k$'s such that $j-k<0$ take $O(n^2)$ time. Therefore, during the execution of the algorithm the total time taken by MIN is $O(n^2)$, and the algorithm's complexity is also $O(n^2)$.

5. Models of failure.

5.1. Discrete case. Our first failure model is discrete where failures occur independently. In the absence of failures each task i has a known completion time t_i and a given probability p_i of completing without failures (i.e., in time t_i). Failures occurring during the execution of task i are detected only at the end of task i. For example, failure detection could be done by checking if an integrity assertion about the state holds at the conclusion of the task. In this model we also assume that the t_i and the rollback times r_i are integers.

To use the algorithms of § 3 and § 4 with this failure model, we need to compute all the $T_{i,j}^0$'s. Consider a segment $[i,j]$. Let the random variable $Y_{i,j}$ denote the time required to execute $[i,j]$ in the presence of failures and no checkpoints. Note that $T_{i,j}^0=E[Y_{i,j}]$. Let $q_t=\mathrm{Prob}[Y_{i,j}=t]$. The moment generating function $\Phi(z)=\Sigma_{t=0}^{\infty} q_t z^t$ of this distribution can be derived as follows. The execution of $[i,j]$ takes at least $t_{i,j}=\Sigma_{k=i}^{j} t_k$ units of time. Therefore,

[1] We exclude the time taken to compute all the $T_{i,j}^0$'s during the initialization of the algorithm. This time depends on the underlying failure model. We consider two failure models in § 5. and with both models it takes $O(n^2)$ time to compute all the $T_{i,j}^0$'s.

$$q_0 = q_1 = \cdots = q_{t_{i,j}-1} = 0 .$$

The execution of $[i,j]$ takes exactly $t_{i,j}$ units of time only if no failures occur. Then we have

$$q_{t_{i,j}} = p_i\, p_{i+1} \cdots p_j .$$

The execution of $[i,j]$ takes more than $t_{i,j}$ units of time only if there is at least one failure during the execution. Consider the first failure that occurs after the computation is started. By conditioning on all the possible tasks where this failure could occur, we derive the following recurrence relation for q_t:

$$q_t = (1-p_i)q_{t-(t_i+r_i)} + p_i(1-p_{i+1})q_{t-(t_i+t_{i+1}+r_i)} + \cdots$$

$$+ p_i p_{i+1} \cdots p_{j-1}(1-p_j)q_{t-(t_i+t_{i+1}+\cdots+t_j+r_i)}$$

for all t such that $t > t_{i,j}$. Multiplying both sides of this equation by z^t and summing over all t, $t > t_{i,j}$, we get

$$\Phi(z) - p_i p_{i+1} \cdots p_j\, z^{t_i+t_{i+1}+\cdots+t_j}$$

$$= (1-p_i)\, z^{t_i+r_i}\Phi(z) + p_i(1-p_{i+1})\, z^{t_i+t_{i+1}+r_i}\Phi(z) + \cdots$$

$$+ p_i p_{i+1} \cdots p_{j-1}(1-p_j)\, z^{t_i+\cdots+t_j+r_i}\Phi(z).$$

Therefore,

$$\Phi(z) =$$

$$\frac{p_i p_{i+1} \cdots p_j\, z^{t_i+t_{i+1}+\cdots+t_j}}{1-(1-p_i)z^{t_i+r_i} - p_i(1-p_{i+1})z^{t_i+t_{i+1}+r_i} - .. - p_i p_{i+1} .. p_{j-1}(1-p_j)z^{t_i+..+t_j+r_i}}$$

By taking the derivatives of $\Phi(z)$ and setting z to one, we can find all the moments of the random variable $Y_{i,j}$ and, in particular, we can obtain $T^0_{i,j} = E[Y_{i,j}]$. It is then easy to verify that the following recurrence relation holds

$$T^0_{i,i} = \frac{t_i}{p_i} + (\frac{1}{p_i} - 1)\, r_i$$

$$T^0_{i,j} = \frac{1}{p_j}(T^0_{i,j-1} + t_j) + (\frac{1}{p_j} - 1)\, r_i \quad \text{for all } j,\ j > i .$$

Using these recurrence relations, we can compute all the $T^0_{i,j}$'s (for i and j, $1 \leqslant i \leqslant j \leqslant n$) in $O(n^2)$-time.

5.2. Continuous case. Rather than assuming the task executions to constitute independent Bernoulli trials as we have done in the previous section, let us consider the case where failures occur according to a stationary renewal process throughout the computation. In other words, the inter-failure times are independent and identically distributed random variables. We assume the existence of a mechanism to detect failures as soon as they occur. As before, when a failure is detected, the computation is rolled back to the most recent checkpoint before it can be resumed. We assume that the checkpoint setups and rollbacks constitute renewal points.

Let the random variable X denote the time until the next failure after a renewal. $F(x) = \text{Prob}[X \leqslant x]$ denotes the distribution function of X which is assumed known. As before, let the random variable $Y_{i,j}$ denote the time required to execute $[i,j]$ in the presence of failures and no checkpoints. Let $V(t) = \text{Prob}[Y_{i,j} \leqslant t]$. We proceed as follows in order to derive an expression for this distribution.

As before, $t_{i,j} = \Sigma_{k=i}^{j} t_k$ be the time required to execute $[i,j]$ without any failures. Clearly, the execution time in the presence of failures cannot be less than this time; that is, $V(t) = 0$ for $t < t_{i,j}$. Conditioning on the time until the first failure after a renewal, we have

$$V(t) = \int_0^\infty \text{Prob}[Y_{i,j} < t \mid X = x] dF(x), \qquad t \geqslant t_{i,j}.$$

If the length of the first failure-free interval is greater than $t_{i,j}$, the computation completes in exactly $t_{i,j}$ time units. Consequently, the above equation can be written as

$$V(t) = \int_0^{t_{i,j}} \text{Prob}[Y_{i,j} < t \mid X = x] dF(x) + (1 - F(t_{i,j})), \quad t \geqslant t_{i,j}.$$

If, however, $X = x < t_{i,j}$, the computation must be rolled back at least once. Since resuming the computation after a rollback constitutes a renewal point, the probability that the total execution time for $[i,j]$ will be at most t time units after having expended x time units due to the failure and r_i time units for the rollback is simply given by $V(t - x - r_i)$. This observation allows us to write $V(t)$ as a *renewal equation* [9]

$$V(t) = \int_0^{t_{i,j}} V(t - x - r_i) dF(x) + (1 - F(t_{i,j})).$$

Making the change of variable $y = x + r_i$ and the substitutions

$$z(x) = \begin{cases} 0\,, & x < t_{i,j}\,, \\ 1 - F(t_{i,j})\,, & x \geqslant t_{i,j} \end{cases}$$

and

$$G(x) = \begin{cases} 0\,, & x < r_i\,, \\ F(x - r_i)\,, & r_i \leqslant x \leqslant t_{i,j} + r_i\,, \\ F(t_{i,j})\,, & x > t_{i,j} + r_i\,, \end{cases}$$

we obtain

$$V(t) = \int_0^t V(t - y) dG(y) + z(t), \qquad \text{for all } t.$$

Note that the integral represents the convolution of the functions V and g where $g(x) = dG(x)/dx$. Taking Laplace transforms yields

$$\bar{V}(s) = \bar{V}(s)\bar{g}(s) + \bar{z}(s)$$

where $\bar{f}(s) = \int_0^\infty e^{-sx} f(x) dx$ denotes the Laplace transform of the function $f(x)$. Finally, the transform of the distribution we are interested is given by

$$\tilde{V}(s) = \frac{\tilde{z}(s)}{1-\tilde{g}(s)}$$

where $\tilde{z}(s) = e^{-s\,t_{i,j}}(1-F(t_{i,j}))/s$ and $\tilde{g}(s) = e^{-s\,r_i}\int_0^{t_{i,j}} e^{-s\,x}\,dF(x)$.

Whether the inverse transform of $\tilde{V}(s)$ has a closed form solution depends on the nature of $F(x)$. Note that the probability *density* function of $Y_{i,j}$, $v(t)=dV(t)/dt$, has the transform

$$\tilde{v}(s) = s\tilde{V}(s) = \frac{e^{-s\,t_{i,j}}(1-F(t_{i,j}))}{1-\tilde{g}(s)}.$$

Since the algorithms given in the previous sections need only the values for $T^0_{i,j}$ (by definition, this is the first moment of $Y_{i,j}$) for all $1 \leqslant i \leqslant n$ and $i \leqslant j \leqslant n$, we do not have to obtain the inverse transform of $\tilde{v}(s)$. All of the moments of $Y_{i,j}$ can be obtained by differentiating the transform of its density. In particular, the first moment is given by

$$T^0_{i,j} = E[Y_{i,j}] = -\frac{d}{ds}\tilde{v}(s)\bigg|_{s=0} = t_{i,j} - \frac{\frac{d}{ds}\tilde{g}(s)|_{s=0}}{1-\tilde{g}(0)} \tag{5.2.1}$$

$$= t_{i,j} + \frac{r_i F(t_{i,j}) + \int_0^{t_{i,j}} t\,dF(t)}{1-F(t_{i,j})}.$$

As an example, we will derive an expression for $E[Y_{i,j}]$ in the presence of Poisson failures (i.e., $F(x)=1-e^{-\lambda x}$ and $dF(x)=\lambda e^{-\lambda x}dx$ where λ is the failure rate). It is important to note that the above derivation of $E[Y_{i,j}]$ does *not* rely on the inter-failure times being exponentially distributed. We have selected the example simply to define one possible form of $F(x)$. Substituting the expressions for $F(x)$ and $dF(x)$ into equation (5.2.1) and simplifying, we obtain

$$T^0_{i,j} = E[Y_{i,j}] = \frac{(e^{\lambda t_{i,j}}-1)(\lambda r_i+1)}{\lambda}.$$

6. Extensions. Up to this point, we have assumed that the checkpoint setups and rollbacks are failure-free. We now discuss how each one of these assumptions can be easily relaxed.

Let the checkpoint setups be subject to the same failure model as the normal tasks. Consider $[i,j]$ and our computation of $T^0_{i,j}$ for this segment. Recall that in the algorithm, $T^0_{i,j}$ denotes the expected execution time of $[i,j]$ given that a checkpoint is established at the end of task j. We consider this checkpoint setup to be a new task of length s_{j+1} which augments segment $[i,j]$. (Obviously, if $j=n$, the completion of the computation, rather than a checkpoint, marks the end of the segment. For notational convenience, we define $s_{n+1}=0$.) The execution time of the augmented segment without any failures becomes $t_{i,j} = \Sigma_{k=i}^{j} t_k + s_{j+1}$. Following this change, the expected execution time $T^0_{i,j}$ of the augmented segment can be computed as shown in § 5. Clearly, since the checkpoint setup costs are now included in the $T^0_{i,j}$'s, the minimization step in the algorithm of Fig. 1 becomes

$$T \leftarrow \min_{1<i\leq j} (T_{1,i-1}^{k-1} + T_{i,j}^{0}) .$$

Suppose now that checkpoint setups are failure-free but the rollbacks are subject to failures. We can apply the algorithm of Fig. 1 provided that the $T_{i,j}^0$'s are computed appropriately. For example, in our discrete model of failure, let ρ_i be the probability that a rollback to location i is failure-free. Proceeding in a similar fashion to § 5.1, we can derive the following recurrence relation for the new $T_{i,j}^0$'s:

$$T_{i,i}^0 = \frac{t_i}{p_i} + (\frac{1}{p_i} - 1)\frac{r_i}{\rho_i} ,$$

$$T_{i,j}^0 = \frac{1}{p_j}(T_{i,j-1}^0 + t_j) + (\frac{1}{p_j} - 1)\frac{r_i}{\rho_i} \quad \text{for all } j,\ j>i.$$

This extension can be similarly incorporated into the continuous failure model analysis.

Finally, since the necessary modifications for the above extensions to the problem are orthogonal, both of them could be present in the basic algorithm.

7. Discussion and conclusions. We have presented an algorithm to select a set of checkpoint locations out of the $n-1$ candidate locations such that the resulting expected execution time for the computation is minimized. The algorithm can be applied to any failure model such that the expected execution time can be computed for any segment with no checkpoints. The time complexity of this algorithm was shown to be $O(n^3)$ where n is the number of tasks making up the computation. We also described an $O(n^2)$ algorithm for the case that, for all i and j, $s_i > s_j$ implies $r_i \geqslant r_j$. In most applications, this is a reasonable assumption since rolling back the state involves loading the same state information which was saved at the checkpoint.

The objective function we have selected for the optimization problem is the expected execution time. For certain time-critical applications, we may be interested in knowing the probability with which the computation will complete in less than some given time. Note that, given the optimum checkpoint locations, the computation can be viewed as a sequence of *segments* rather than tasks where each segment is delimited by a checkpoint. The execution times for these segments are independent and we have derived their moment generating functions (in the two failure models that we considered). The moment generating function for the total execution time of the computation is simply the product of the individual segment moment generating functions. In principle, this function can be inverted to obtain the distribution of the total execution time. Given this distribution, confidence bounds for the total execution time can be derived. If inverting the moment generating function is not possible, we can use it to derive the mean, the variance, and any higher moments of the total execution time (note that the mean is already part of the algorithm's output). Given these values, we can derive the probability with which the computation completes within a given interval about the known mean by an application of Chebyshev's Inequality.

If the static structure of a computation is not simple, its dynamic characterization as a sequence of tasks may be difficult to obtain. Given this observation, a generalization of our work would represent the static computation as a Directed Acyclic Graph (DAG) where the vertices are tasks and an edge (i,j) denotes a possible execution where task j follows task i. Here we may want to position the checkpoints on this DAG in a way that minimizes the maximum expected execution time over all the possible execution paths. If we augment the DAG by associating probabilities with each edge such that the two tasks connected by the edge appear in succession with that probability, we can pose the problem of selecting checkpoint locations such that the total expected execution time for the computation is minimized.

Appendix. We introduce some definitions to be used in the following proofs leading to those of Theorems 1 and 2 of § 4. Consider segment $[i,j]$ for some $i \leqslant j$. Let $p_{i,j}$ be the probability that no failures occur during the execution of $[i,j]$, $\alpha_{i,j}$ be $1/p_{i,j}$ and $t_{i,j}$ be the execution time of $[i,j]$ when no failures occur. Given that a failure occurs during the execution of $[i,j]$, $l_{i,j}$ denotes the expected time interval from the beginning of task i to the time the failure is detected.

LEMMA 1. *For all i and j such that $i \leqslant j$ we have*

$$T^0_{i,j} = t_{i,j} + (\alpha_{i,j} - 1)(l_{i,j} + r_i).$$

For all i,j and k such that $i < k \leqslant j$ we have

$$T^0_{i,j} = \alpha_{k,j}\, T^0_{i,k-1} + t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_i)\,.$$

Proof. We present the proofs for the case where the failure times are discrete random variables. The continuous case proofs follow simply by replacing the sums with integrals and discrete probabilities with density functions.

The first equality can be proven as follows. Let the random variable X denote the time until the first failure is detected after the beginning of $[i,j]$. Conditioning on this time, we have

$$T^0_{i,j} = t_{i,j} \cdot \mathrm{Prob}[X > t_{i,j}] + \sum_{x \leqslant t_{i,j}} (x + r_i + T^0_{i,j}) \cdot \mathrm{Prob}[X = x].$$

By definition, $p_{i,j} = \mathrm{Prob}[X > t_{i,j}]$. Consequently,

$$T^0_{i,j} = t_{i,j}\, p_{i,j} + (r_i + T^0_{i,j})(1 - p_{i,j}) + \sum_{x \leqslant t_{i,j}} x\, \mathrm{Prob}[X = x].$$

Noting that, by definition, $l_{i,j} = \Sigma_{x \leqslant t_{i,j}} x\, \mathrm{Prob}[X=x]/(1-p_{i,j})$, and solving for $T^0_{i,j}$ we have $p_{i,j}\, T^0_{i,j} = p_{i,j} t_{i,j} + (1 - p_{i,j})(l_{i,j} + r_i)$. Dividing both sides by $p_{i,j}$ results in the desired expression.

To prove the second equality, we condition on the time until the first failure is detected after the beginning of $[k,j]$. Letting X denote this time, we have

$$T^0_{i,j} = T^0_{i,k-1} + t_{k,j} \cdot \mathrm{Prob}[X > t_{k,j}] + \sum_{x \leqslant t_{k,j}} (x + r_i + T^0_{i,j}) \cdot \mathrm{Prob}[X = x].$$

As before, rewriting this equation in terms of $l_{k,j}$, and solving for $T^0_{i,j}$ we obtain

$$p_{k,j}\, T^0_{i,j} = T^0_{i,k-1} + p_{k,j}\, t_{k,j} + (1 - p_{k,j})(l_{k,j} + r_i)\,.$$

Dividing the equation by $p_{k,j}$ completes the proof. □

LEMMA 2. *For all i,j and k such that $i < k \leqslant j$ and $\alpha_{k,j} \neq 1$ we have*

$$T^0_{i,k-1} + T^0_{k,j} + s_k \leqslant T^0_{i,j} \quad \textit{if and only if} \quad s_k \leqslant (\alpha_{k,j} - 1)(r_i + T^0_{i,k-1} - r_k)\,.$$

Furthermore, the equality holds in the left expression if and only if it holds in the right one.

Proof. From Lemma 1 we have

$$T^0_{i,j} = \alpha_{k,j}\, T^0_{i,k-1} + t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_i)$$

and

$$T^0_{k,j} = t_{k,j} + (\alpha_{k,j} - 1)\,(l_{k,j} + r_k)\,.$$

Therefore $T^0_{i,k-1} + T^0_{k,j} + s_k \leqslant T^0_{i,j}$ if and only if

$$T^0_{i,k-1} + t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_k) + s_k \leqslant \alpha_{k,j}\, T^0_{i,k-1} + t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_i)$$

This inequality holds if and only if

$$(\alpha_{k,j} - 1)(l_{k,j} + r_k) + s_k \leqslant (\alpha_{k,j} - 1)(l_{k,j} + r_i + T^0_{i,k-1})\,.$$

Since $\alpha_{k,j} > 1$, this is satisfied if and only if

$$l_{k,j} + r_k + s_k\,(\alpha_{k,j} - 1)^{-1} \leqslant l_{k,j} + r_i + T^0_{i,k-1}\,.$$

This last inequality holds if and only if $s_k \leqslant (\alpha_{k,j} - 1)(r_i + T^0_{i,k-1} - r_k)$. It is also easy to check that $T^0_{i,k-1} + T^0_{k,j} + s_k = T^0_{i,j}$ if and only if $s_k = (\alpha_{k,j} - 1)(r_i + T^0_{i,k-1} - r_k)$. □

LEMMA 3. *Suppose the m-optimal solutions for* $[i,j]$ *are such that their rightmost checkpoint location is* k, $i < k \leqslant j$. *Then we have* $\alpha_{k,j} > 1$ *and* $s_k < (\alpha_{k,j} - 1)(r_i + T^0_{i,k-1} - r_k)$.

Proof. The proof is by induction on m (note that by hypothesis $m \geqslant 1$).

Suppose $m = 1$. By hypothesis we have

$$T^1_{i,j} = T^0_{i,k-1} + T^0_{k,j} + s_k < T^0_{i,j}\,. \tag{L3.1}$$

If we show that $\alpha_{k,j} \neq 1$ then Lemma 3 follows directly from Lemma 2. Suppose $\alpha_{k,j} = 1$. From Lemma 1 we have $T^0_{i,j} = T^0_{i,k-1} + t_{k,j}$ and $T^0_{k,j} = t_{k,j}$. Therefore, $T^0_{i,j} = T^0_{i,k-1} + T^0_{k,j}$ and this contradicts (L3.1). So $\alpha_{k,j} > 1$ and Lemma 3 follows.

Now assume Lemma 3 holds for all m, $1 \leqslant m < r$, for some $r > 1$. We show that it must also hold for $m = r$. Suppose the r-optimal solutions for $[i,j]$ are such that their rightmost checkpoint location is k, $i < k \leqslant j$. Let $\mathbf{L}^{r-1}_{i,k-1}$ be a $(r-1)$-optimal solution for $[i,k-1]$. We consider $\mathbf{L}^r_{i,j} = \mathbf{L}^{r-1}_{i,k-1} \,\|\, \langle k \rangle$. Note that this must be a r-optimal solution for $[i,j]$. Let h be the rightmost checkpoint location of $\mathbf{L}^{r-1}_{i,k-1}$. We have $i \leqslant h < k \leqslant j$.

Assume first that $i < h$. By induction hypothesis we have $s_h < (\alpha_{h,k-1} - 1)(r_i + T^0_{i,h-1} - r_h)$, and $\alpha_{h,k-1} > 1$. Note that $s_h \geqslant 0$, and therefore

$$r_h < r_i + T^0_{i,h-1}\,. \tag{L3.2}$$

Since $\mathbf{L}^1_{h,j} = \langle k \rangle$ is a 1-optimal solution for $[h,j]$ then, by induction hypothesis, we have $\alpha_{k,j} > 1$ and $s_k < (\alpha_{k,j} - 1)(r_h + T^0_{h,k-1} - r_k)$. Suppose, for contradiction, that $s_k \geqslant (\alpha_{k,j} - 1)(r_i + T^0_{i,k-1} - r_k)$. Combining the last two inequalities we have $r_i + T^0_{i,k-1} < r_h + T^0_{h,k-1}$. From Lemma 1 we have

$$r_i + [\,\alpha_{h,k-1}\, T^0_{i,h-1} + t_{h,k-1} + (\alpha_{h,k-1} - 1)(l_{h,k-1} + r_i)]$$
$$< r_h + [\,t_{h,k-1} + (\alpha_{h,k-1} - 1)(l_{h,k-1} + r_h)]\,.$$

So $(r_i + T^0_{i,h-1})\alpha_{h,k-1} < r_h\, \alpha_{h,k-1}$, and $r_i + T^0_{i,h-1} < r_h$ contradicting (L3.2). Therefore we must have $s_k < (\alpha_{k,j} - 1)(r_i + T^0_{i,k-1} - r_k)$.

Suppose now that $i = h$. In this case it is clear that $\mathrm{L}^{r-1}_{i,k-1} = <>$ and $\mathrm{L}^r_{i,j} = <k>$, therefore $\mathrm{L}^1_{i,j} = <k>$, i.e., $<k>$ is the 1-optimal checkpoint selection for $[i,j]$. Then, by induction hypothesis, we have $\alpha_{k,j} > 1$ and $s_k < (\alpha_{k,j} - 1)(r_i + T^0_{i,k-1} - r_k)$. $\square$

LEMMA 4. *Suppose the m-optimal solutions for $[i,j]$ are such that the rightmost checkpoint location is k, $i < k \leqslant j$. Then for all h, $i < h < k$, we have*

$$s_k - s_h \leqslant (\alpha_{k,j} - 1)(r_h + T^0_{h,k-1} - r_k).$$

Proof. We prove the lemma by induction on m (note that by hypothesis $m \geqslant 1$).

Assume first that $m = 1$. Suppose, for contradiction, that for some h, $i < h < k$, we have

$$s_k - s_h > (\alpha_{k,j} - 1)(r_h + T^0_{h,k-1} - r_k). \tag{L4.1}$$

From Lemma 3 we have $s_k < (\alpha_{k,j} - 1)(r_i + T^0_{i,k-1} - r_k)$, and $\alpha_{k,j} > 1$. Therefore, $r_h + T^0_{h,k-1} < r_i + T^0_{i,k-1}$, and from Lemma 1 we have

$$r_h + [\, t_{h,k-1} + (\alpha_{h,k-1} - 1)(l_{h,k-1} + r_h)]$$

$$< r_i + [\, \alpha_{h,k-1}\, T^0_{i,h-1} + t_{h,k-1} + (\alpha_{h,k-1} - 1)(l_{h,k-1} + r_i)]$$

so, $\alpha_{h,k-1}\, r_h < \alpha_{h,k-1}(r_i + T^0_{i,h-1})$, and

$$r_h < r_i + T^0_{i,h-1}. \tag{L4.2}$$

Since $<k>$ is the 1-optimal solution for $[i,j]$ we have

$$T^1_{i,j} = T^0_{i,k-1} + T^0_{k,j} + s_k \leqslant T^0_{i,h-1} + T^0_{h,j} + s_h$$

and therefore $T^0_{i,k-1} - T^0_{i,h-1} + s_k \leqslant T^0_{h,j} - T^0_{k,j} + s_h$. We define $\Delta_1 = T^0_{i,k-1} - T^0_{i,h-1}$ and $\Delta_2 = T^0_{h,j} - T^0_{k,j}$, and we write the last inequality as

$$\Delta_1 + s_k \leqslant \Delta_2 + s_h. \tag{L4.3}$$

Applying Lemma 1 we have

$$\Delta_1 = [\, \alpha_{h,k-1}\, T^0_{i,h-1} + t_{h,k-1} + (\alpha_{h,k-1} - 1)(l_{h,k-1} + r_i)] - T^0_{i,h-1}$$

so

$$\Delta_1 = t_{h,k-1} + (\alpha_{h,k-1} - 1)(l_{h,k-1} + r_i + T^0_{i,h-1}).$$

We also have

$$\Delta_2 = [\, \alpha_{k,j}\, T^0_{h,k-1} + t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_h)] - [\, t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_k)]$$

and therefore

$$\Delta_2 = \alpha_{k,j}\, T^0_{h,k-1} + (\alpha_{k,j} - 1)(r_h - r_k).$$

From (L4.1) we have $r_h - r_k < -T^0_{h,k-1} + (s_k - s_h)(\alpha_{k,j} - 1)^{-1}$. Therefore, $\Delta_2 < \alpha_{k,j}\, T^0_{h,k-1} - (\alpha_{k,j} - 1)\, T^0_{h,k-1} + s_k - s_h$, that is

$$\Delta_2 < T^0_{h,k-1} + s_k - s_h \, . \tag{L4.4}$$

From (L4.3) and (L4.4) we obtain

$$\Delta_1 < T^0_{h,k-1} \, , \tag{L4.5}$$

that is, $t_{h,k-1} + (\alpha_{h,k-1} - 1)(l_{h,k-1} + r_i + T^0_{i,h-1}) < T^0_{h,k-1}$. By Lemma 1 we have

$$t_{h,k-1} + (\alpha_{h,k-1} - 1)(l_{h,k-1} + r_i + T^0_{i,h-1}) < t_{h,k-1} + (\alpha_{h,k-1} - 1)(l_{h,k-1} + r_h)$$

so $(\alpha_{h,k-1} - 1)(r_i + T^0_{i,h-1}) < (\alpha_{h,k-1} - 1)\, r_h$. Suppose $\alpha_{h,k-1} = 1$. Then $\Delta_1 = t_{h,k-1}$ and $T^0_{h,k-1} = t_{h,k-1}$, a contradiction to (L4.5). So we must have $\alpha_{h,k-1} > 1$ and $r_i + T^0_{i,h-1} < r_h$. But this contradicts (L4.2) and therefore $s_k - s_h \leqslant (\alpha_{k,j} - 1)(r_h + T^0_{h,k-1} - r_k)$ for all h, $i < h < k$.

Now assume the lemma holds for all m, $1 \leqslant m < r$, for some $r > 1$. We show that it must also hold for $m = r$. Suppose the r-optimal solutions for $[i,j]$ are such that their rightmost checkpoint location is k, $i < k \leqslant j$. Let $\mathbf{L}^{r-1}_{i,k-1}$ be a $(r-1)$-optimal solution for $[i,k-1]$. We consider $\mathbf{L}^r_{i,j} = \mathbf{L}^{r-1}_{i,k-1} \,\|\, \langle k \rangle$. Note that this must be a r-optimal solution for $[i,j]$. Let p be the location of the rightmost checkpoint of $\mathbf{L}^{r-1}_{i,k-1}$. Note that $i \leqslant p < k \leqslant j$.

Assume first that $i < p$. Consider some h such that $i < h < k$. There are two possible cases, either $p < h$ or $h \leqslant p$. We show that in both cases $s_k - s_h \leqslant (\alpha_{k,j} - 1)(r_h + T^0_{h,k-1} - r_k)$.

(*i*) Suppose $i < p < h < k \leqslant j$. Since k must be the (rightmost) checkpoint location of the 1-optimal solution for $[p,j]$ then, by induction hypothesis, we have $s_k - s_h \leqslant (\alpha_{k,j} - 1)(r_h + T^0_{h,k-1} - r_k)$.

(*ii*) We now consider h such that $i < h \leqslant p < k \leqslant j$. Suppose, for contradiction, that $s_k - s_h > (\alpha_{k,j} - 1)(r_h + T^0_{h,k-1} - r_k)$. Since $\mathbf{L}^1_{p,j} = \langle k \rangle$ is a 1-optimal solution for $[p,j]$ then, by Lemma 3, we have $s_k < (\alpha_{k,j} - 1)(r_p + T^0_{p,k-1} - r_k)$, and $\alpha_{k,j} > 1$. Combining the last two inequalities we have $r_h + T^0_{h,k-1} < r_p + T^0_{p,k-1}$. Therefore $h \neq p$ (i.e., $h < p$), and from Lemma 1 we have

$$r_h + [\,\alpha_{p,k-1}\, T^0_{h,p-1} + t_{p,k-1} + (\alpha_{p,k-1} - 1)(l_{p,k-1} + r_h)]$$
$$< r_p + [\,t_{p,k-1} + (\alpha_{p,k-1} - 1)(l_{p,k-1} + r_p)] \, .$$

By simplifying we obtain

$$r_h + T^0_{h,p-1} < r_p \, . \tag{L4.6}$$

Since the rightmost checkpoint location of the $(r-1)$-optimal solutions for $[i,k-1]$ is p, and $i < h < p \leqslant k-1$, then, by induction hypothesis, we have

$$s_p - s_h \leqslant (\alpha_{p,k-1} - 1)(r_h + T^0_{h,p-1} - r_p) \, . \tag{L4.7}$$

From Lemma 3 we also have $\alpha_{p,k-1} > 1$. From (L4.6) and (L4.7) we obtain $s_p - s_h < 0$. From our assumption about checkpointing costs either $s_p = s_h$, a contradiction, or $r_p \leqslant r_h$ which contradicts (L4.6). So we must have $s_k - s_h \leqslant (\alpha_{k,j} - 1)(r_h + T^0_{h,k-1} - r_k)$.

Suppose now that $i = p$. In this case it is clear that $L_{i,k-1}^{r-1} = <>$ and $L_{i,j}^{r} = <k>$, and therefore $L_{i,j}^{1} = <k>$, i.e., $<k>$ is the 1-optimal checkpoint selection for $[i,j]$. Then, by induction hypothesis, we have $s_k - s_h \leqslant (\alpha_{k,j} - 1)(r_h + T_{h,k-1}^0 - r_k)$, for all h, $i \leqslant h < k$. □

LEMMA 5. *Suppose the m-optimal solutions for $[i,j]$ are such that their rightmost checkpoint location is k, $i \leqslant k \leqslant j$. Consider h and p such that $i \leqslant h \leqslant k \leqslant j \leqslant p$. If $i = h$ or $s_h \leqslant s_k$ then we have $T_{k,p}^0 - T_{k,j}^0 \leqslant T_{h,p}^0 - T_{h,j}^0$.*

Proof. The result is obvious for $i = k$, or $h = k$, or $j = p$. Otherwise, suppose that for contradiction

$$T_{k,p}^0 - T_{k,j}^0 > T_{h,p}^0 - T_{h,j}^0 \tag{L5.1}$$

for some $i \leqslant h < k \leqslant j < p$ such that $i = h$ or $s_h \leqslant s_k$. From Lemma 1 we have

$$T_{k,p}^0 = \alpha_{j+1,p}\, T_{k,j}^0 + t_{j+1,p} + (\alpha_{j+1,p} - 1)(l_{j+1,p} + r_k)$$

and therefore

$$T_{k,p}^0 - T_{k,j}^0 = t_{j+1,p} + (\alpha_{j+1,p} - 1)(T_{k,j}^0 + l_{j+1,p} + r_k). \tag{L5.2}$$

Similarly we have

$$T_{h,p}^0 - T_{h,j}^0 = t_{j+1,p} + (\alpha_{j+1,p} - 1)(T_{h,j}^0 + l_{j+1,p} + r_h). \tag{L5.3}$$

From (L5.1), (L5.2) and (L5.3) we have

$$(\alpha_{j+1,p} - 1)(T_{k,j}^0 + l_{j+1,p} + r_k) > (\alpha_{j+1,p} - 1)(T_{h,j}^0 + l_{j+1,p} + r_h).$$

Suppose $\alpha_{j+1,p} = 1$. In this case, from (L5.2) and (L5.3) we have $T_{k,p}^0 - T_{k,j}^0 = T_{h,p}^0 - T_{h,j}^0 = t_{j+1,p}$ which contradicts (L5.1). Therefore $\alpha_{j+1,p} > 1$, and we have

$$r_k + T_{k,j}^0 > r_h + T_{h,j}^0. \tag{L5.4}$$

Note that $h < k \leqslant j$ and by applying Lemma 1 we derive

$$r_k + [t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_k)] > r_h + [\alpha_{k,j}\, T_{h,k-1}^0 + t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_h)].$$

Then we have $\alpha_{k,j}\, r_k > \alpha_{k,j}\,(r_h + T_{h,k-1}^0)$ and

$$r_k > r_h + T_{h,k-1}^0. \tag{L5.5}$$

From Lemma 3, we have $s_k < (\alpha_{k,j} - 1)(r_i + T_{i,k-1}^0 - r_k)$, and $\alpha_{k,j} > 1$, therefore

$$r_k < r_i + T_{i,k-1}^0. \tag{L5.6}$$

If $i = h$ then (L5.6) contradicts (L5.5). Suppose $i < h$. From Lemma 4, we have $s_k - s_h \leqslant (\alpha_{k,j} - 1)(r_h + T_{h,k-1}^0 - r_k)$. Therefore, if $s_h \leqslant s_k$, then $r_k \leqslant r_h + T_{h,k-1}^0$ which also contradicts (L5.5). Then neither $i = h$ nor $s_h \leqslant s_k$ and the proof is complete. □

LEMMA 6. *If $i \leqslant h \leqslant k \leqslant j$ and $r_k \leqslant r_h$ then $T_{k,j}^0 \leqslant T_{h,j}^0$.*

Proof. The lemma is obvious for $h = k$. We now assume $i \leqslant h < k \leqslant j$. From Lemma 1 we have

$$T^0_{h,j} = \alpha_{k,j}\, T^0_{h,k-1} + t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_h)$$

and $T^0_{k,j} = t_{k,j} + (\alpha_{k,j} - 1)(l_{k,j} + r_k)$. Therefore we have

$$T^0_{h,j} - T^0_{k,j} = \alpha_{k,j}\, T^0_{h,k-1} + (\alpha_{k,j} - 1)(r_h - r_k)$$

and since $r_k \leqslant r_h$ then $T^0_{h,j} - T^0_{k,j} \geqslant 0$. □

LEMMA 7. *If $i \leqslant h \leqslant k \leqslant j \leqslant p$ and $r_k \leqslant r_h$ then*

$$T^0_{h,j} - T^0_{k,j} \leqslant T^0_{h,p} - T^0_{k,p}.$$

Proof. The lemma is obvious if $h = k$ or $j = p$. We now assume that $i \leqslant h < k \leqslant j < p$. In the proof of Lemma 6 we showed that

$$T^0_{h,j} - T^0_{k,j} = \alpha_{k,j}\, T^0_{h,k-1} + (\alpha_{k,j} - 1)(r_h - r_k)$$

and similarly we have

$$T^0_{h,p} - T^0_{k,p} = \alpha_{k,p}\, T^0_{h,k-1} + (\alpha_{k,p} - 1)(r_h - r_k).$$

Since $p > j$ it is clear that $\alpha_{k,p} \geqslant \alpha_{k,j}$. By hypothesis we also have $r_h - r_k \geqslant 0$ and therefore $T^0_{h,j} - T^0_{k,j} \leqslant T^0_{h,p} - T^0_{k,p}$. □

THEOREM 1. *Suppose the m-optimal solutions for $[i,j]$ are such that their rightmost checkpoint location is k, $i \leqslant k \leqslant j$. Then for any $p > j$ the m-optimal solutions for $[i,p]$ are such that their rightmost checkpoint location is some h, $h \geqslant k$.*

Proof. If $i = k$, the theorem is obvious. Assume $i < k$ and therefore $m \geqslant 1$. Suppose, for contradiction, that the rightmost checkpoint location of the m-optimal solutions for $[i,p]$ is h, $i \leqslant h < k$.

Assume first that $i < h$. By hypothesis we must have

$$T^{m-1}_{i,k-1} + T^0_{k,j} + s_k \leqslant T^{m-1}_{i,h-1} + T^0_{h,j} + s_h. \tag{T1.1}$$

From our definition of h we also have

$$T^{m-1}_{i,k-1} + T^0_{k,p} + s_k \geqslant T^{m-1}_{i,h-1} + T^0_{h,p} + s_h. \tag{T1.2}$$

Suppose equality holds in (T1.1). Then the $(m-1)$-optimal solutions for $[i,h-1]$ cannot include fewer checkpoints than the $(m-1)$-optimal solutions for $[i,k-1]$. Therefore equality cannot also hold in (T1.2), otherwise k would be the rightmost checkpoint location of the m-optimal solutions for $[i,p]$ contradicting our definition of h. So equality cannot hold simultaneously in (T1.1) and in (T1.2). Then, subtracting (T1.1) from (T1.2) we obtain

$$T^0_{k,p} - T^0_{k,j} > T^0_{h,p} - T^0_{h,j}. \tag{T1.3}$$

If $s_h \leqslant s_k$ then (T1.3) contradicts Lemma 5; therefore $s_k < s_h$. From our assumption about checkpointing costs we have $r_k \leqslant r_h$. Then, from Lemma 6, we also have $T^0_{k,j} \leqslant T^0_{h,j}$, so

$$r_k + T^0_{k,j} \leqslant r_h + T^0_{h,j}. \tag{T1.4}$$

However, in the proof of Lemma 5 we showed that (T1.3) implies $r_k + T^0_{k,j} > r_h + T^0_{h,j}$ which contradicts (T1.4).

Suppose now that $i = h$, i.e., the m-optimal solutions for $[i,p]$ contain no checkpoints ($\mathbf{L}^m_{i,p} = \langle\, \rangle$). In this case we must have

$$T^{m-1}_{i,k-1} + T^0_{k,j} + s_k < T^0_{i,j} . \tag{T1.5}$$

Since $\mathbf{L}^m_{i,p} = < >$ then

$$T^{m-1}_{i,k-1} + T^0_{k,p} + s_k \geqslant T^0_{i,p} . \tag{T1.6}$$

Subtracting (T1.5) from (T1.6) we obtain (T1.3). From (T1.3) we can also derive a contradiction exactly as before, and the proof is complete. □

THEOREM 2. *Suppose the m-optimal solutions for $[i,j]$ are such that their rightmost checkpoint location is k, $i \leqslant k \leqslant j$. Then the $(m+1)$-optimal solutions for $[i,j]$ are such that their rightmost checkpoint location is some h, $h \geqslant k$.*

Proof. If $i = k$, the theorem is obvious. Assume $i < k$ and therefore $m \geqslant 1$. If $T^{m+1}_{i,j} = T^m_{i,j}$ then m-optimal solutions for $[i,j]$ are also $(m+1)$-optimal solutions for $[i,j]$ and the theorem clearly holds. We now assume $T^{m+1}_{i,j} < T^m_{i,j}$. Note that in this case $(m+1)$-optimal solutions for $[i,j]$ have exactly $m+1$ checkpoints. Let g be the number of checkpoints in the m-optimal solutions for $[i,j]$. Note that $1 \leqslant g \leqslant m$. We form a particular m-optimal solution $\mathbf{L}^m_{i,j} = <u_g, u_{g-1}, \ldots, u_1>$ for $[i,j]$ as follows. Let u_1 be the rightmost checkpoint location of the m-optimal solutions for $[i,j]$, let u_2 be the rightmost checkpoint location of the $(m-1)$-optimal solutions for $[i,u_1 - 1]$, ..., and let u_g be the rightmost checkpoint location of the $[(m+1)-g]$-optimal solutions for $[i,u_{g-1} - 1]$. Note that with this construction $<u_g, u_{g-1}, \ldots, u_f>$ is a $[(m+1)-f]$-optimal solution for $[i,u_{f-1} - 1]$, for all f, $2 \leqslant f \leqslant g$. By hypothesis we have $u_1 = k$. We also form in a similar way a particular $(m+1)$-solution $\mathbf{L}^{m+1}_{i,j} = <v_{m+1}, v_m, \ldots, v_1>$ for $[i,j]$. We define $v_1 = h$. Suppose, for contradiction, that $h < k$ (i.e., $v_1 < u_1$). There are two possible cases. We show that each one leads to a contradiction.

I. Suppose first that $v_m \leqslant u_g$. Note that $v_{m+1} < v_m$ and therefore $v_{m+1} < u_g$. Since $\mathbf{L}^m_{i,j} = <u_g, u_{g-1}, \ldots, u_1>$ is a m-optimal solution for $[i,j]$ we have

$$T^m_{i,j} = T^0_{i,u_g - 1} + \sum_{l=g-1}^{1} T^0_{u_{l+1},u_l - 1} + T^0_{u_1,j} + \sum_{l=g}^{1} s_{u_l}$$

and

$$T^m_{i,j} \leqslant T^0_{i,v_m - 1} + \sum_{l=m-1}^{1} T^0_{v_{l+1},v_l - 1} + T^0_{v_1,j} + \sum_{l=m}^{1} s_{v_l}$$

where the indexes of the summations scan the segments from left to right, and the summation $\sum_{l=a}^{b} \ldots$ is defined to be zero if $a < b$.

Therefore,

$$T^0_{i,u_g - 1} - T^0_{i,v_m - 1} + F \leqslant 0 \tag{T2.1}$$

where

$$F = \sum_{l=g-1}^{1} T^0_{u_{l+1},u_l - 1} - \sum_{l=m-1}^{1} T^0_{v_{l+1},v_l - 1} + T^0_{u_1,j} - T^0_{v_1,j} + \sum_{l=g}^{1} s_{u_l} - \sum_{l=m}^{1} s_{v_l} .$$

Since $\mathbf{L}^{m+1}_{i,j} = <v_{m+1}, v_m, \ldots, v_1>$ is a $(m+1)$-optimal solution for $[i,j]$ we have

$$T^{m+1}_{i,j} = T^0_{i,v_{m+1} - 1} + T^0_{v_{m+1},v_m - 1} + \sum_{l=m-1}^{1} T^0_{v_{l+1},v_l - 1} + T^0_{v_1,j} + s_{v_{m+1}} + \sum_{l=m}^{1} s_{v_l} . \tag{T2.2}$$

Note that if the $<v_{m+1}, u_g, u_{g-1}, \ldots, u_1>$ selection of checkpoints achieved the time $T_{i,j}^{m+1}$ (or less) this would contradict our assumption that the rightmost checkpoint location of the $(m+1)$-optimal solutions for $[i,j]$ is $v_1 < u_1$. Therefore we must have

$$T_{i,j}^{m+1} < T_{i,v_{m+1}-1}^0 + T_{v_{m+1},u_g-1}^0 + \sum_{l=g-1}^{1} T_{u_{l+1},u_l-1}^0 + T_{u_1,j}^0 + s_{v_{m+1}} + \sum_{l=g}^{1} s_{u_l} . \tag{T2.3}$$

Subtracting (T2.2) from (T2.3) we get

$$0 < T_{v_{m+1},u_g-1}^0 - T_{v_{m+1},v_m-1}^0 + F . \tag{T2.4}$$

From (T2.1) and (T2.4) we have

$$T_{v_{m+1},u_g-1}^0 - T_{v_{m+1},v_m-1}^0 > T_{i,u_g-1}^0 - T_{i,v_m-1}^0 . \tag{T2.5}$$

Note that $i < v_{m+1} \leqslant v_m - 1 \leqslant u_g - 1$ and $<v_{m+1}>$ is the 1-optimal solution for $[i,v_m - 1]$. By Lemma 5 we have

$$T_{v_{m+1},u_g-1}^0 - T_{v_{m+1},v_m-1}^0 \leqslant T_{i,u_g-1}^0 - T_{i,v_m-1}^0$$

which contradicts (T2.5).

II. Suppose now that $v_m > u_g$. Let k be the minimum r such that $v_1 \leqslant u_1$, $v_2 \leqslant u_2$, ..., $v_{r-1} \leqslant u_{r-1}$, and $v_r > u_r$. Note that such k exists because $v_m > u_g$ (and therefore $v_g > u_g$). There are two possible cases.

(*i*) Suppose $v_{k+1} \leqslant u_k$. In this case we have $v_{k+1} \leqslant u_k < v_k < v_{k-1} \leqslant u_{k-1}$. Since $\mathbf{L}_{i,j}^m = <u_g, \ldots, u_1>$ is a m-optimal solution for $[i,j]$ we have

$$T_{i,j}^m = T_{i,u_g-1}^0 + \sum_{l=g-1}^{k} T_{u_{l+1},u_l-1}^0 + T_{u_k,u_{k-1}-1}^0 + \sum_{l=k-2}^{1} T_{u_{l+1},u_l-1}^0 + T_{u_1,j}^0 + \sum_{l=g}^{1} s_{u_l} . \tag{T2.6}$$

The checkpoint selection $<u_g, \ldots, u_k, v_{k-1}, \ldots, v_1>$ cannot achieve a smaller time than $T_{i,j}^m$ for $[i,j]$. Therefore we have

$$T_{i,j}^m \leqslant T(u,v) \tag{T2.7}$$

where $T(u,v)$ is defined as

$$T(u,v) = T_{i,u_g-1}^0 + \sum_{l=g-1}^{k} T_{u_{l+1},u_l-1}^0 + T_{u_k,v_{k-1}-1}^0 +$$

$$+ \sum_{l=k-2}^{1} T_{v_{l+1},v_l-1}^0 + T_{v_1,j}^0 + \sum_{l=g}^{k} s_{u_l} + \sum_{l=k-1}^{1} s_{v_l} .$$

From (T2.6) and (T2.7) we have

$$T_{u_k,u_{k-1}-1}^0 - T_{u_k,v_{k-1}-1}^0 + G \leqslant 0 \tag{T2.8}$$

where

$$G = \sum_{l=k-2}^{1} (T_{u_{l+1},u_l-1}^0 - T_{v_{l+1},v_l-1}^0) + T_{u_1,j}^0 - T_{v_1,j}^0 + \sum_{l=k-1}^{1} (s_{u_l} - s_{v_l}) .$$

Since $\mathbf{L}_{i,j}^{m+1} = \langle v_{m+1}, \ldots, v_k, v_{k-1}, \ldots, v_1 \rangle$ is a $(m+1)$-optimal solution for $[i,j]$ we have

$$T_{i,j}^{m+1} = T_{i,v_{m+1}-1}^0 + \sum_{l=m}^{k} T_{v_{l+1},v_l-1}^0 + T_{v_k,v_{k-1}-1}^0 + \sum_{l=k-2}^{1} T_{v_{l+1},v_l-1}^0 + T_{v_1,j}^0 + \sum_{l=m+1}^{1} s_{v_l} . \tag{T2.9}$$

Since with $u_1 > v_1$ then the $\langle v_{m+1}, \ldots, v_k, u_{k-1}, \ldots, u_1 \rangle$ checkpoint selection cannot achieve the time $T_{i,j}^{m+1}$ (or less) for $[i,j]$ without contradicting the optimality of $\mathbf{L}_{i,j}^{m+1}$. Therefore we have

$$T_{i,j}^{m+1} < T(v,u) \tag{T2.10}$$

where $T(v,u)$ is defined as

$$T(v,u) = T_{i,v_{m+1}-1}^0 + \sum_{l=m}^{k} T_{v_{l+1},v_l-1}^0 + T_{v_k,u_{k-1}-1}^0$$

$$+ \sum_{l=k-2}^{1} T_{u_{l+1},u_l-1}^0 + T_{u_1,j}^0 + \sum_{l=m+1}^{k} s_{v_l} + \sum_{l=k-1}^{1} s_{u_l} .$$

Subtracting (T2.9) from (T2.10) we have

$$0 < T_{v_k,u_{k-1}-1}^0 - T_{v_k,v_{k-1}-1}^0 + G . \tag{T2.11}$$

From (T2.8) and (T2.11) we have

$$T_{v_k,u_{k-1}-1}^0 - T_{v_k,v_{k-1}-1}^0 > T_{u_k,u_{k-1}-1}^0 - T_{u_k,v_{k-1}-1}^0 . \tag{T2.12}$$

Note that we have $v_{k+1} \leqslant u_k < v_k \leqslant v_{k-1} - 1 \leqslant u_{k-1} - 1$ and $\langle v_k \rangle$ is the 1-optimal solution for $[v_{k+1}, v_{k-1} - 1]$. By Lemma 5 we know that if $s_{v_k} \geqslant s_{u_k}$ then

$$T_{v_k,u_{k-1}-1}^0 - T_{v_k,v_{k-1}-1}^0 \leqslant T_{u_k,u_{k-1}-1}^0 - T_{u_k,v_{k-1}-1}^0 \tag{T2.13}$$

which contradicts (T2.12); therefore $s_{v_k} < s_{u_k}$. From our assumption about checkpointing costs we have $r_{v_k} \leqslant r_{u_k}$.

From (T2.7) and (T2.10) we have $T(u,v) - T_{i,j}^{m+1} > T_{i,j}^{m} - T(v,u)$, that is

$$T_{u_k,v_{k-1}-1}^0 - T_{v_k,v_{k-1}-1}^0 > T_{u_k,u_{k-1}-1}^0 - T_{v_k,u_{k-1}-1}^0 . \tag{T2.14}$$

But since $v_{k+1} \leqslant u_k < v_k \leqslant v_{k-1} - 1 \leqslant u_{k-1} - 1$, and $r_{v_k} \leqslant r_{u_k}$ then we can apply Lemma 7 and obtain $T_{u_k,v_{k-1}-1}^0 - T_{v_k,v_{k-1}-1}^0 \leqslant T_{u_k,u_{k-1}-1}^0 - T_{v_k,u_{k-1}-1}^0$. This contradicts (T2.14) and therefore we cannot have $v_{k+1} \leqslant u_k$.

(*ii*) Suppose $v_{k+1} > u_k$. In this case we have $u_k < v_{k+1} < v_k < v_{k-1} \leqslant u_{k-1}$. We show that this is not possible. Observe that $\langle u_g, \ldots, u_k \rangle$ is a $[(m+1)-k]$-optimal solution for $[i, u_{k-1} - 1]$; its rightmost checkpoint location is u_k, $u_k < v_{k+1}$. We also note that $\langle v_{m+1}, \ldots, v_{k+1} \rangle$ is a $[(m+1)-k]$-optimal solution for $[i, v_k - 1]$. Since $u_{k-1} - 1 > v_k - 1$ then, by Theorem 1, the rightmost checkpoint location of the $[(m+1)-k]$-optimal solutions for $[i, u_{k-1} - 1]$ is some u such that $u \geqslant v_{k+1}$. Since $u_k < v_{k+1}$, this contradicts the optimality of $\langle u_g, \ldots, u_k \rangle$ for $[i, u_{k-1} - 1]$, and the proof is complete. □

REFERENCES

[1] K. M. CHANDY, *A survey of analytic models of rollback and recovery strategies*, Computer, 5 (1975), pp. 40-47.

[2] K. M. CHANDY AND C.V. RAMAMOORTHY, *Rollback and recovery strategies for computer programs*, IEEE Trans. on Comput., 6 (1972), pp. 546-556.

[3] K. M. CHANDY, J. C. BROWNE, C. W. DISSLY AND W. R. UHRIG, *Analytic models for rollback and recovery strategies in data base systems*, IEEE Trans. on Soft. Engr., 1 (1975), pp. 100-110.

[4] E. GELENBE, *On the optimum checkpoint interval*, J. Assoc. Comput. Mach., 2 (1979), pp. 259-270.

[5] E. GELENBE AND D. DEROCHETTE, *Performance of rollback recovery systems under intermittent failures*, Comm. Assoc. Comput. Mach., 6 (1978), pp. 493-499.

[6] J. GRAY, P. MCJONES, M. BLASGEN, B. LINSAY, R. LORIE, T. PRICE, F. PUTZOLU AND I. TRAIGER, *The recovery manager of the System R database manager*, Comput. Surveys, 2 (1981), pp. 223-242.

[7] I. KOREN, Z. KOREN AND S. SU, *Analysis of a recovery procedure*, Tech. Report, Technion-Israel Institute of Technology, (1983). To appear in IEEE Trans. on Comput.

[8] B. RANDELL, P. LEE AND P. TRELEAVEN, *Reliability issues in computing system design*, Comput. Surveys, 2 (1978), pp. 123-166.

[9] S. M. ROSS, *Applied probability models with optimization applications*, Holden-Day, San Francisco, CA, 1970.

[10] R. D. SCHLICHTING AND F.B. SCHNEIDER, *Fail-stop processors: an approach to designing fault-tolerant computing systems*, ACM Trans. on Computer Systems, 3 (1983), pp. 222-238.

[11] J. W. YOUNG, *A first order approximation to the optimum checkpoint interval*, Comm. Assoc. Comput. Mach., 9 (1974), pp. 530-531.