Partial Detectors Versus Replication To Cope With Silent Errors




Alix Tremodeux[1], Anne Benoit[1,2], Thomas Herault[3,4], Yves Robert[1]

1 ENS Lyon & Inria; 2 IUF & IDEaS, Georgia Tech;
3 Inria Bordeaux; 4 Univ. Tennessee Knoxville

Scheduling Workshop – July 8, 2025

Outline

Framework

- Iterative algorithm executing on a large-scale platform
- Silent errors may strike
- Minimize expected cost per iteration

Strategies

- Replication: only general-purpose approach
- Detectors: verified checkpoints (application-specific)

⚠ Need perfect detectors: no false negatives (recall $r = 1$)

- Can we use partial detectors with recall $r < 1$?
- If yes, how does it compare to replication?

Strategies

- Replication: only general-purpose approach
- Detectors: verified checkpoints (application-specific)

⚠ Need perfect detectors: no false negatives (recall $r = 1$)

- Can we use partial detectors with recall $r < 1$?
- If yes, how does it compare to replication?

Outline

Approach

- Execution is partitioned into *segments* of M iterations, each followed by a checkpoint
- Execution of a new segment (after a checkpoint C):

Two different errors never lead to the same (incorrect) result

Approach

- Execution is partitioned into *segments* of M iterations, each followed by a checkpoint
- Execution of a new segment (after a checkpoint C):

  Re-execute until getting the same result twice

Two different errors never lead to the same (incorrect) result

Approach

- Execution is partitioned into *segments* of M iterations, each followed by a checkpoint
- Execution of a new segment (after a checkpoint C):
  - Execute segment for the first time and checkpoint result res_1
  - While results after $t \geq 1$ attempts are all different, execute new attempt $t+1$:
    - recover from checkpoint C
    - redo the M iterations
    - checkpoint result res_{t+1}
  - Keep the outcome of the two identical checkpoints
  - Proceed to next segment

Hypothesis

Two different errors never lead to the same (incorrect) result

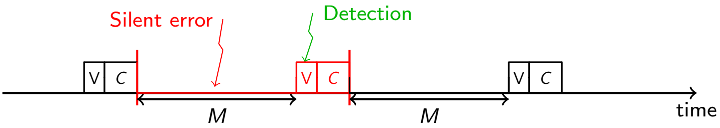
Minimizing expected time per iteration

- A silent error may strike each iteration independently and with fixed probability f (Geometric law for IATs)
- Given a segment:
  - cost of first attempt: $M + C$
  - cost of following attempts: $R + M + C$
  - number of attempts until one is successful:
  geometric law of parameter $p_S = (1 - f)^M$
  - expected cost $cost(M) = (M + C) + \left(\frac{2}{p_S} - 1\right)(R + M + C)$
  - expected slowdown $\mathcal{S} = \frac{cost(M)}{M} = \frac{2(R+C)}{Mp_S} + \frac{2}{p_S} - \frac{R}{M}$
  - differentiate and solve, find optimal M numerically
- No closed-form solution unless $R = 0$ (then Lambert ☺)

Outline

1. Framework
2. Replication
3. Partial detectors
4. Experiments

Perfect detectors



- Perfect detector of cost V
- Segment: M iterations + detector V + checkpoint C
- Recall $r = 1 \Rightarrow$ verified checkpoint
- Optimal value of M well-known
- First-order approximation à la Young-Daly

Perfect detectors

Silent error

Detection

Do you believe it?

- Detectors are not perfect ☹
- High recall is expensive if at all achievable ☹

Can we use partial detectors?

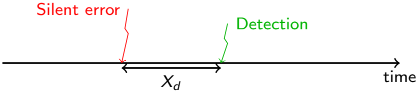
- First-order approximation à la Young-Daly

Can we use partial detectors?

Can we use partial detectors?

. . . unless we make some reasonable assumption . . .

Bounded detection latency



Error and detection latency

Assumption

- If a silent error strikes at iteration I ...
  ...it will be detected at iteration $(I-1)+X$ or after
- X obeys a probability distribution with bounded support $[1, D]$

Rationale

- The impact of the silent error on the application data grows and becomes more and more *detectable*
- For computation errors: numerical amplification as execution progresses

Case study

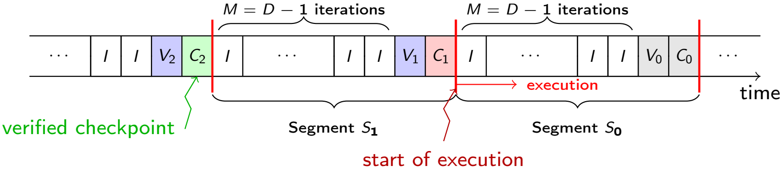
- X truncated geometric R.V. with bounded support $[1, D]$
- $X = \min(Y, D)$, with Y geometric R.V. of parameter θ

Typical values for maximum detection distance D

Recall θ	$\min\{d \mid P(X \geq d) \leq 10^{-6}\}$	$\min\{d \mid P(X \geq d) \leq 10^{-9}\}$
0.2	62	93
0.4	28	41
0.9	6	9

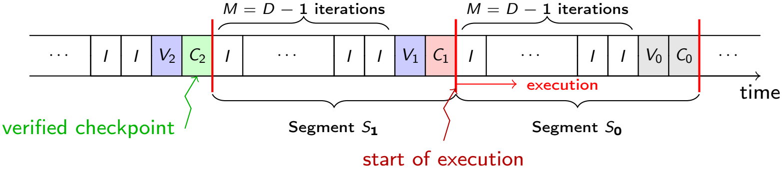
- Efficient partial detector $\theta = 0.9$:
  distance detection never exceeds 10 in practice
- Poor partial detector $\theta = 0.2$ (capturing only 20% of errors):
  distance detection never exceeds 100 in practice

Simple scheme



- Completing the execution of segment S_0
- Checkpoints C_1 and C_2 stored in memory
- C_2 is verified (by induction) but C_1 is not (yet)

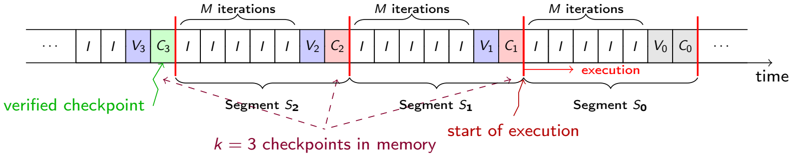
Simple scheme



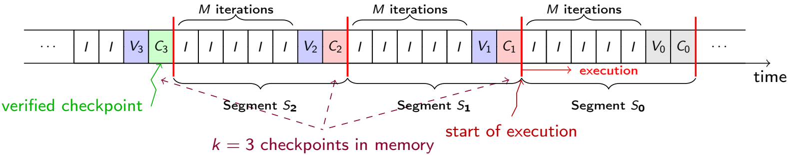
After V_0:

- No error is detected
  $\Rightarrow$ C_1 is verified (Why?)
  $\Rightarrow$ take C_0 and overwrite C_2
- An error is detected
  $\Rightarrow$ Roll back to C_2, re-execute S_1 then S_0

General scheme

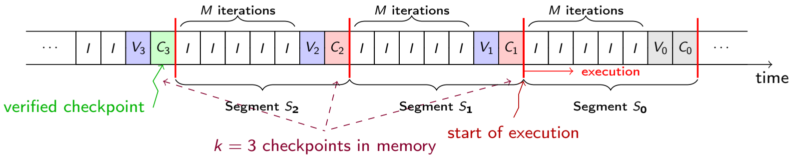


General scheme



- k segments, k checkpoints in memory ($k = 3$)
- M iterations per segment ($M = 5$)
- Need $(k-1)M \geq D - 1$
- Given M, use $k = \left\lceil \frac{D-1}{M} \right\rceil + 1$
- $M = 5$ and $k = 3 \Rightarrow D \leq 11$
- Conversely for $D = 11$, $k = 3 \Rightarrow 5 \leq M \leq 9$

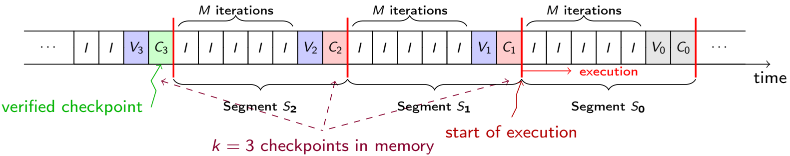
General scheme



After V_0:

- No error is detected
  $\Rightarrow$ C_k is verified
  $\Rightarrow$ take C_0 and overwrite C_k
- An error is detected
  $\Rightarrow$ Roll back to C_k, re-execute $S_{k-1}S_{k-2}\ldots S_0$

General scheme



- Compute E_0: expected time to process segment S_0 with a successful verification V_0 and take new checkpoint C_0 (then delete C_k from memory and move on to next segment)
- Minimize $\mathcal{S} = \frac{E_0}{M}$ (with $k = \lceil \frac{D-1}{M} \rceil + 1$)

Computation of E_0

$$\langle f, R(k) g\rangle_{L^2(\mathbb{T}^{d+1})} = \int_{\mathbb{T}^{d+1}}\int_{\mathbb{T}^{d+1}} \widehat{D}(k,p,q)\,\overline{f(p)}\,g(q)\,dq\,dp$$

$$= (2\pi)^{\frac{1-d}{2}} \sum_{\tau,(0,\xi),(0,\eta)} e^{-ik\cdot\tau} D(\tau,(0,\xi),(0,\eta))\, \bar{\check{f}}(-\xi)\,\check{g}(-\eta)$$

$$= (2\pi)^{\frac{1-d}{2}} \sum_{\tau,(0,\xi),(0,\eta)} e^{-ik\cdot\tau}\left\langle \Theta(\xi), e^{-\frac{1}{2}|\tau_0|H} e^{-\frac{1}{2}i\tau\cdot P}\Theta(\eta)\right\rangle_{\mathcal{H}} \bar{\check{f}}(-\xi)\,\check{g}(-\eta)$$

$$= (2\pi)^{\frac{1-d}{2}}\sum_{\tau} e^{-ik\cdot\tau}\left\langle\Theta(\check{f}), e^{-\frac{1}{2}|\tau_0|H}e^{-\frac{1}{2}i\tau\cdot P}\Theta(\check{g})\right\rangle_{\mathcal{H}} \qquad \text{(VI.3)}$$

$$= (2\pi)^{\frac{1-d}{2}}\int_0^\infty\int_{\mathbb{T}^d}\left(\sum_{\tau_0\in\mathbb{Z}} e^{-ik_0\tau_0}e^{-\frac{1}{2}|\tau_0|\lambda_0}\right)\left(\sum_{\tau\in\mathbb{Z}^d} e^{-i\mathbf{k}\cdot\tau}e^{-\frac{1}{2}i\tau\cdot\lambda}\right) dE_\lambda$$

$$= (2\pi)^{\frac{d+1}{2}}\int_0^\infty\int_{\mathbb{T}^d}\frac{\sinh(\lambda_0/2)}{\cosh(\lambda_0/2)-\cos k_0}\,\delta\left(\frac{\lambda}{2}+\mathbf{k}\right) dE_\lambda$$

$$= (2\pi)^{\frac{d+1}{2}}\int_0^\infty \frac{\sinh(\lambda_0/2)}{\cosh(\lambda_0/2)-\cos k_0}\, dE_{(\lambda_0,-2\mathbf{k})}.$$

After a long and painful derivation . . . ☹

Computation of E_0

$$E_0 = a_k C + b_k(M+V) + c_k R$$
$$a_k = 1 + \left(\frac{1}{\Phi_{k-1}} - 1\right) u_k,\ b_k = \frac{1}{\Phi_{k-1}} + \left(\frac{1}{\Phi_{k-1}} - 1\right) v_k,\ c_k = \left(\frac{1}{\Phi_{k-1}} - 1\right) w_k$$
$$u_k = 1 + \sum_{m=0}^{k-3} \prod_{\ell=0}^{m} \frac{1}{\Phi_{k-2-l}},\ v_k = \sum_{m=0}^{k-2} \prod_{\ell=0}^{m} \frac{1}{\Phi_{k-2-l}},\ w_k = \prod_{\ell=0}^{k-2} \frac{1}{\Phi_\ell}$$

where

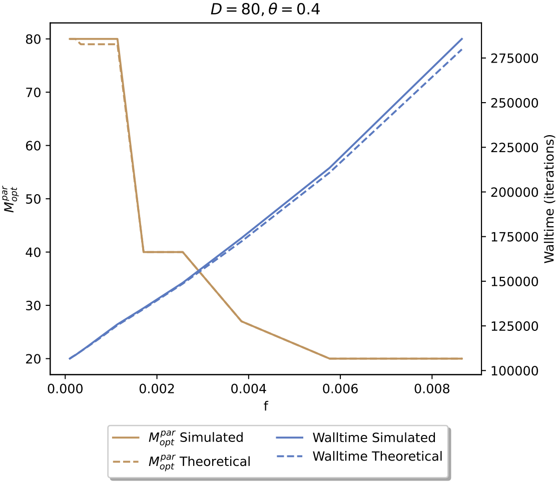
$$P_{i,j} = P\big(X \le jM + (M-i+1)\big) - P\big(X \le (j-1)M + (M-i+1)\big)$$
$$P_{i,>j} = P\big(X > jM + (M-i+1)\big)$$
$$Q_\ell = \prod_{i=1}^{M} \left(1 - \frac{fP_{i,\ell}}{(1-f)+f(P_{i,>\ell}+P_{i,\ell})}\right)$$
$$\Phi_j = \prod_{\ell=0}^{j} Q_\ell$$

Probabilities

$P_{i,O}$: error at iteration i detected at the end of current segment
$P_{i,j}$: error at iteration i detected exactly j segments later
$P_{i,>j}$: error at iteration i detected more than j segments later

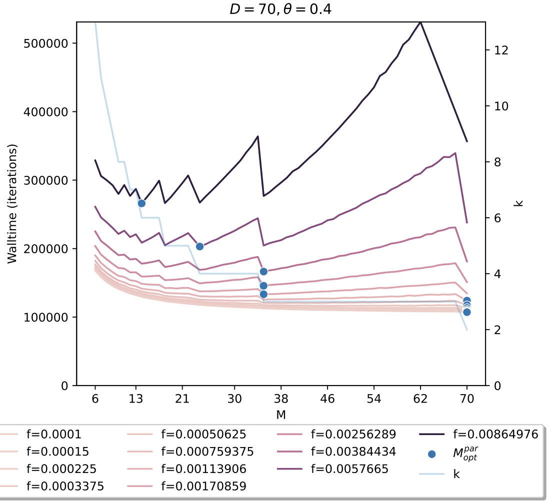
Outline

Validation



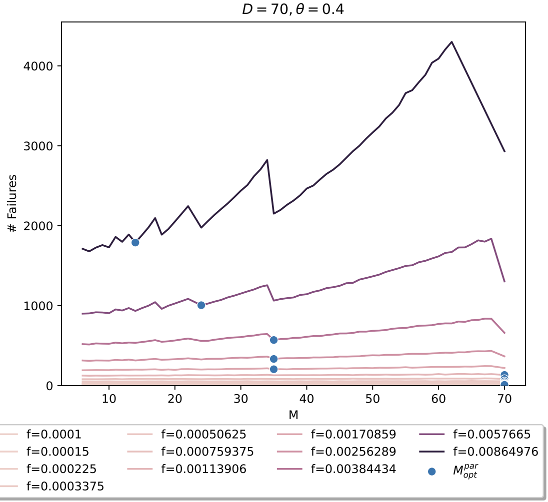
Optimal values of (k, M_{opt}^{par}) and walltime, model vs. simulation

Parameter exploration (1/4)



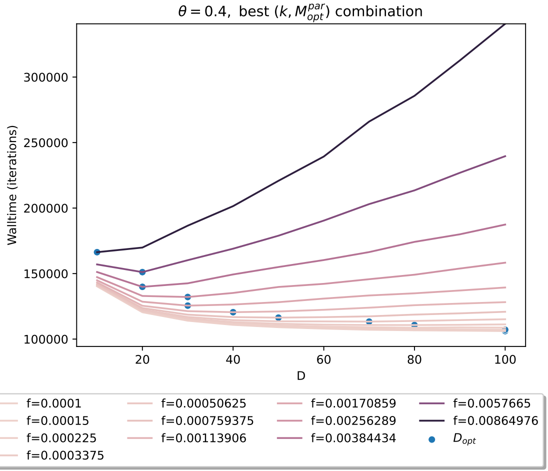
Simulated walltime varying error risk f and segment size M

Parameter exploration (2/4)



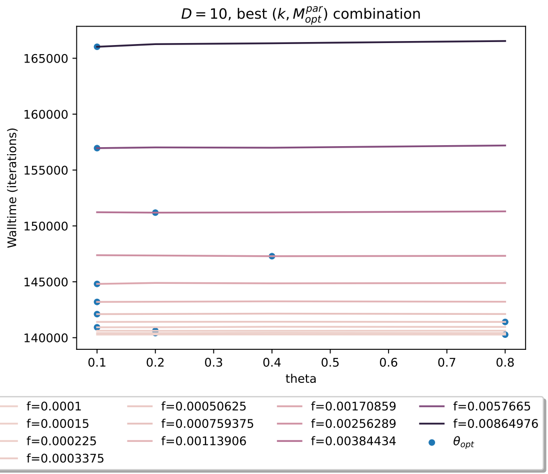
Number of errors varying error risk f and segment size M

Parameter exploration (3/4)



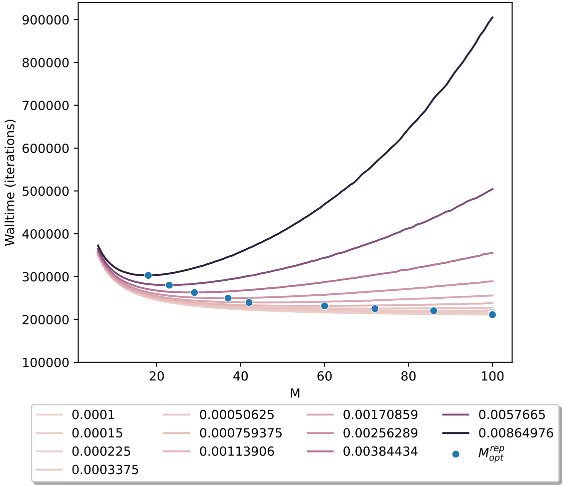
Impact of latency bound D on walltime, varying f and M

Parameter exploration (4/4)



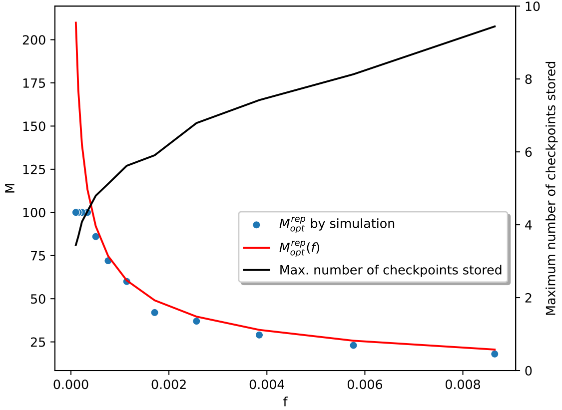
Impact of detection probability θ on walltime, varying f and M

Replication (1/3)



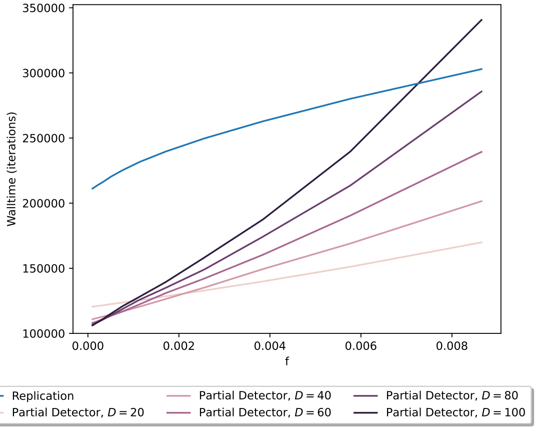
Walltime for replication, varying f and M

Replication (2/3)



Best segment size M for replication, model vs. simulation

Replication (3/3)



Comparing walltime, replication vs. partial detection with $\theta = 0.4$

Conclusion

Synthesis

- First comparison of replication with partial detection
- Optimal solution for both approaches
- Monte-Carlo simulations perfectly match model predictions
- Partial detectors can massively outperform replication
- Number of stored checkpoints:
  fixed for partial detection, unknown for replication

Future work

- Extend analysis to false positives (precision < 1):
  extra rollbacks, recoveries, re-executions due to false alarms
- Experimental validation with PCG

What about AI?

- No Artificial Intelligence in this work . . .
  . . . some Human Magnificence instead ☺
- AI tools give simple solutions to complicated problems
- Scheduling guys give complicated solutions to simple problems

What about AI?

- No Artificial Intelligence in this work . . .
  . . . some Human Magnificence instead ☺
- AI tools give simple solutions to complicated problems
- Scheduling guys give complicated solutions to simple problems

What about AI?

- No Artificial Intelligence in this work . . .
  . . . some Human Magnificence instead ☺
- AI tools give simple solutions to complicated problems
- Scheduling guys give complicated solutions to simple problems

Solution to an Optimization Problem

- A greedy algorithm better than ten other greedy algorithms
- A greedy algorithm whose performance seems close to a lower bound
- A greedy algorithm whose performance is guaranteed to be close to a lower bound
- An optimal solution (usually via Dynamic Programming)

Solution to an Optimization Problem

- A greedy algorithm better than ten other greedy algorithms
- A greedy algorithm whose performance seems close to a lower bound
- A greedy algorithm whose performance is guaranteed to be close to a lower bound
- An optimal solution (usually via Dynamic Programming)

Solution to an Optimization Problem

- A greedy algorithm better than ten other greedy algorithms
- A greedy algorithm whose performance
  seems close to a lower bound
- A greedy algorithm whose performance
  is guaranteed to be close to a lower bound
- An optimal solution (usually via Dynamic Programming)

Solution to an Optimization Problem

- A greedy algorithm better than ten other greedy algorithms
- A greedy algorithm whose performance
  seems close to a lower bound
- A greedy algorithm whose performance
  is guaranteed to be close to a lower bound
- An optimal solution (usually via Dynamic Programming)

What AI could have done for this problem?

- The problem: magically materialized during some work for the NumPeX project
- The model: several similar studies 
- The pattern with segments and global rollback 
- The solution: not sure! even with the assistance of several computer algebra tools / theorem provers