On the Combination of Silent Error Detection and Checkpointing

Guillaume Aupy, Anne Benoit, Thomas Hérault, Yves Robert, Frédéric Vivien & Dounia Zaidouni

ICL Friday Lunch – September 6, 2013
What about the retreat?
What about the retreat?
What about the retreat?
Silent error detection

Y. Robert

Introduction

Optimal Checkpointing strategy
- Exponential distribution
- Arbitrary distribution

Limited resources

Incorporating detection
- $k$ checkpoints for 1 verification
- $k$ verifications for 1 checkpoint

Conclusion, future work

What about the retreat?
1. Introduction

2. Optimal Checkpointing strategy
   Exponential distribution
   Arbitrary distribution

3. Limited resources

4. Incorporating detection
   $k$ checkpoints for 1 verification
   $k$ verifications for 1 checkpoint

5. Conclusion, future work
Sources of failures

- Analysis of error and failure logs

- In 2005 (Ph. D. of CHARNG-DA LU): “Software halts account for the most number of outages (59-84 percent), and take the shortest time to repair (0.6-1.5 hours). Hardware problems, albeit rarer, need 6.3-100.7 hours to solve.”

- In 2007 (Garth Gibson, ICPP Keynote):

- In 2008 (Oliner and J. Stearley, DSN Conf.):

<table>
<thead>
<tr>
<th>Type</th>
<th>Raw Count</th>
<th>Raw %</th>
<th>Filtered Count</th>
<th>Filtered %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware</td>
<td>174,586,516</td>
<td>98.04</td>
<td>1,999</td>
<td>18.78</td>
</tr>
<tr>
<td>Software</td>
<td>144,899</td>
<td>0.08</td>
<td>6,814</td>
<td>64.01</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>3,350,044</td>
<td>1.88</td>
<td>1,832</td>
<td>17.21</td>
</tr>
</tbody>
</table>

Conclusion: Both Hardware and Software failures have to be considered.
A few definitions

- Many types of faults: software error, hardware malfunction, memory corruption
- Many possible behaviors: silent, transient, unrecoverable
- **Restrict to silent errors**
- This includes some software faults, some hardware errors (soft errors in L1 cache), double bit flip
- **Silent error detected when corrupt data is activated**
A few definitions

- Many types of faults: software error, hardware malfunction, memory corruption
- Many possible behaviors: silent, transient, unrecoverable
- **Restrict to silent errors**
- This includes some software faults, some hardware errors (soft errors in L1 cache), double bit flip
- Silent error detected when corrupt data is activated
- *Silent errors are the black swans of errors* (Marc Snir)
Introduction

Optimal Checkpointing strategy
  Exponential distribution
  Arbitrary distribution

Limited resources

Incorporating detection
  $k$ checkpoints for 1 verification
  $k$ verifications for 1 checkpoint

Conclusion, future work
Silent error detection

Y. Robert

Introduction

Optimal Checkpointing strategy

Exponential distribution

Arbitrary distribution

Limited resources

Incorporating detection

$k$ checkpoints for 1 verification

$k$ verifications for 1 checkpoint

Conclusion, future work

---

Figure: Error and detection latency.

- $X_e$ inter arrival time between errors; mean time $\mu_e$
- $X_d$ error detection time; mean time $\mu_d$
- Assume $X_d$ and $X_e$ independent
Introduction

Optimal Checkpointing strategy

- Exponential distribution
- Arbitrary distribution

Limited resources

Incorporating detection

- $k$ checkpoints for 1 verification
- $k$ verifications for 1 checkpoint

Conclusion, future work

Notations

- $C$ checkpointing time
- $R$ recovery time
- $W$ total work
- $w$ some piece of work
When $X_e$ follows an Exponential law of parameter $\lambda_e = \frac{1}{\mu_e}$, in order to execute a total work of $w + C$, we need:
For one chunk

When $X_e$ follows an Exponential law of parameter $\lambda_e = \frac{1}{\mu_e}$, in order to execute a total work of $w + C$, we need:

- Probability of execution without error

$$
\mathbb{E}(T(w)) = e^{-\lambda_e(w+C)}(w + C) + (1 - e^{-\lambda_e(w+C)}) \left( \mathbb{E}(T_{\text{lost}}) + \mathbb{E}(X_d) + \mathbb{E}(T_{\text{rec}}) + \mathbb{E}(T(w)) \right)
$$
When $X_e$ follows an Exponential law of parameter $\lambda_e = \frac{1}{\mu_e}$, in order to execute a total work of $w + C$, we need:

- Probability of execution without error

$$E(T(w)) = e^{-\lambda_e(w+C)}(w + C)$$

$$+ (1 - e^{-\lambda_e(w+C)}) (E(T_{lost}) + E(X_d) + E(T_{rec}) + E(T(w)))$$

- Probability of error during $w + C$
For one chunk

When $X_e$ follows an Exponential law of parameter $\lambda_e = \frac{1}{\mu_e}$, in order to execute a total work of $w + C$, we need:

- Probability of execution without error

$$\mathbb{E}(T(w)) = e^{-\lambda_e(w+C)}(w + C)$$

- Probability of error during $w + C$

- Execution time with an error
Let us focus on the time lost due to an error:
\[ \mathbb{E}(T_{\text{lost}}) + \mathbb{E}(X_d) + \mathbb{E}(T_{\text{rec}}) \]
Let us focus on the time lost due to an error:
\[ \mathbb{E}(T_{lost}) + \mathbb{E}(X_d) + \mathbb{E}(T_{rec}) \]

This is the time elapsed between the completion of the last checkpoint and the error:

\[
\mathbb{E}(T_{lost}) = \int_0^\infty x \Pr(X = x | X < w + C) \, dx
\]

\[
= \frac{1}{\Pr(X < w + C)} \int_0^{w+C} x \lambda e^{-\lambda x} \, dx
\]

\[
= \frac{1}{\lambda e} - \frac{w + C}{e^{\lambda(w+C)} - 1}
\]
Let us focus on the time lost due to an error:
\[ \mathbb{E}(T_{\text{lost}}) + \mathbb{E}(X_d) + \mathbb{E}(T_{\text{rec}}) \]

This is the time needed for error detection, \( \mathbb{E}(X_d) = \mu_d \)
Let us focus on the time lost due to an error:
\[ \mathbb{E}(T_{lost}) + \mathbb{E}(X_d) + \mathbb{E}(T_{rec}) \]

This is the time to recover from the error (there can be a fault during recovery):

\[
\mathbb{E}(T_{rec}) = e^{-\lambda e^R}R
\]

\[
+ (1 - e^{-\lambda e^R})(\mathbb{E}(R_{lost}) + \mathbb{E}(X_d) + \mathbb{E}(T_{rec}))
\]
Let us focus on the time lost due to an error:
\[ \mathbb{E}(T_{\text{lost}}) + \mathbb{E}(X_d) + \mathbb{E}(T_{\text{rec}}) \]

This is the time to recover from the error (there can be a fault during recovery):

\[
\mathbb{E}(T_{\text{rec}}) = e^{-\lambda e R} R \\
+ (1 - e^{-\lambda e R}) \left( \mathbb{E}(R_{\text{lost}}) + \mathbb{E}(X_d) + \mathbb{E}(T_{\text{rec}}) \right)
\]

Similarly to \( \mathbb{E}(T_{\text{lost}}) \), we have: \( \mathbb{E}(R_{\text{lost}}) = \frac{1}{\lambda e} - \frac{R}{e^{\lambda e R} - 1} \).
Let us focus on the time lost due to an error:

$$\mathbb{E}(T_{\text{lost}}) + \mathbb{E}(X_d) + \mathbb{E}(T_{\text{rec}})$$

This is the time to recover from the error (there can be a fault during recovery):

$$\mathbb{E}(T_{\text{rec}}) = e^{-\lambda e^R} R$$

$$+ (1 - e^{-\lambda e^R}) (\mathbb{E}(R_{\text{lost}}) + \mathbb{E}(X_d) + \mathbb{E}(T_{\text{rec}}))$$

Similarly to $\mathbb{E}(T_{\text{lost}})$, we have: $\mathbb{E}(R_{\text{lost}}) = \frac{1}{\lambda e} - \frac{R}{e^{\lambda e^R} - 1}$.

So finally, $\mathbb{E}(T_{\text{rec}}) = (e^{\lambda e^R} - 1)(\mu e + \mu d)$
At the end of the day,

\[ \mathbb{E}(T(w)) = e^{\lambda e R} (\mu_e + \mu_d) (e^{\lambda e (w+C)} - 1) \]

This is the exact solution!
For multiple chunks

Using \( n \) chunks of size \( w_i \) (with \( \sum_{i=1}^{n} w_i = W \)), we have:

\[
E(T(W)) = K \sum_{i=1}^{n} (e^{\lambda(w_i + C)} - 1)
\]

with \( K \) constant.

Independent of \( \mu_d \)!

Minimum when all the \( w_i \)'s are equal to \( w = W/n \).
Using $n$ chunks of size $w_i$ (with $\sum_{i=1}^{n} w_i = W$), we have:

$$\mathbb{E}(T(W)) = K \sum_{i=1}^{n} (e^{\lambda w_i + C} - 1)$$

with $K$ constant.

Minimum when all the $w_i$’s are equal to $w = W/n$.

Optimal $n$ can be found by differentiation

A good approximation is $w = \sqrt{2\mu eC}$ (Young’s formula)
Extend results when $X_e$ follows an arbitrary distribution of mean $\mu_e$. 
Waste: fraction of time not spent for useful computations
- $\text{TIME}_{\text{base}}$: application base time
- $\text{TIME}_{\text{FF}}$: with periodic checkpoints but failure-free
- $\text{TIME}_{\text{Final}}$: expectation of time with failures

\[
(1 - \text{WASTE}_{\text{FF}}) \text{TIME}_{\text{FF}} = \text{TIME}_{\text{base}}
\]

\[
(1 - \text{WASTE}_{\text{Fail}}) \text{TIME}_{\text{Final}} = \text{TIME}_{\text{FF}}
\]

\[
\text{WASTE} = \frac{\text{TIME}_{\text{Final}} - \text{TIME}_{\text{base}}}{\text{TIME}_{\text{Final}}}
\]

\[
\text{WASTE} = 1 - (1 - \text{WASTE}_{\text{FF}})(1 - \text{WASTE}_{\text{Fail}})
\]
We can show that

\[ \text{WASTE}_{FF} = \frac{C}{T} \]

\[ \text{WASTE}_{Fail} = \frac{T}{2} + R + \mu_d \frac{\mu_e}{\mu_e} \]

Back to our model
We can show that

\[ W_{\text{AFF}} = \frac{C}{T} \]

\[ W_{\text{Fail}} = \frac{T/2 + R + \mu_d}{\mu_e} \]

Only valid if \( \frac{T}{2} + R + \mu_d \ll \mu_e \).
We can show that

$$WASTE_{FF} = \frac{C}{T}$$

$$WASTE_{Fail} = \frac{T}{2} + R + \mu_d \frac{\mu_e}{\mu_e}$$

Only valid if $\frac{T}{2} + R + \mu_d \ll \mu_e$.

Then the waste is minimized for

$$T_{opt} = \sqrt{2(\mu_e - (R + \mu_d))C} \approx \sqrt{2\mu_e C}$$
Theorem

- Best period is $T_{opt} \approx \sqrt{2\mu e C}$
- Independent of $X_d$
Limitation of this model

Analytical optimal solutions, valid for arbitrary distributions, without any knowledge on \( X_d \) except its mean

However, if \( X_d \) can be arbitrary large:

- Do not know how far to roll back in time
- Need to store all checkpoints taken during execution
Introduction

Optimal Checkpointing strategy
Exponential distribution
Arbitrary distribution

Limited resources

Incorporating detection
$k$ checkpoints for 1 verification
$k$ verifications for 1 checkpoint

Conclusion, future work
The case with limited resources

Assume that we can only save the last $k$ checkpoints.

Definition (Critical failure)

Error detected when all checkpoints contain corrupted data. Happens with probability $P_{\text{risk}}$ during whole execution.
The case with limited resources

\( \mathbb{P}_{\text{risk}} \) decreases when \( T \) increases (when \( X_d \) is fixed).
Hence, \( \mathbb{P}_{\text{risk}} \leq \varepsilon \) leads to a lower bound \( T_{\text{min}} \) on \( T \)

We have derived an analytical form for \( \mathbb{P}_{\text{risk}} \) when \( X_d \) follows an Exponential law. We use it as a good (?) approximation for arbitrary laws.
Figure: $k = 3$, $\lambda_e = \frac{10^5}{100y}$, $\lambda_d = 30\lambda_e$, $w = 10d$, $C = R = 600s$

$T_{\text{opt}} \approx 100\text{min}$, $P_{\text{risk}}(T_{\text{opt}}) \approx 38 \cdot 10^{-5}$, for a waste of 23.45%

To reduce $P_{\text{risk}}$ to $10^{-4}$, a $T_{\text{min}}$ of 8000 seconds is sufficient, increasing the waste by only 0.6%. In this case, the benefit of fixing the period to $\max(T_{\text{opt}}, T_{\text{min}})$ is obvious.
More optimistic technologic scenario (smaller $C$ and $R$):

$T_{\text{opt}}$ is largely reduced (down to less than 35 minutes), but $\mathbb{P}_{\text{risk}}(T_{\text{opt}})$ climbs to $1/2$, an unacceptable value. To reduce $\mathbb{P}_{\text{risk}}$ to $10^{-4}$, it becomes necessary to consider a $T_{\text{min}}$ of 6650 seconds. The waste increases to 15%, significantly higher than the optimal one, which is below 10%.

Figure: $k = 3$, $\lambda_e = \frac{10^5}{100y}$, $\lambda_d = 30\lambda_e$, $w = 10d$, $C = R = 60s$. 
Figure: $k = 3, \lambda_e = \frac{10^5}{100 y}, \lambda_d = 30 \lambda_e, w = 10 d, C = R = 600 s$
Limitation of the model

It is not clear how can one detect when the error occurred (hence to identify the last valid checkpoint)

Need a verification mechanism to check the correctness of the checkpoints. This has a cost!
Possible solution: add verifications; use a periodic mechanism to verify that there were no silent errors in previous computations.
Introduction

Optimal Checkpointing strategy
- Exponential distribution
- Arbitrary distribution

Limited resources

Incorporating detection
- $k$ checkpoints for 1 verification
- $k$ verifications for 1 checkpoint

Conclusion, future work
Assume there are no errors during checkpoints (less error sources when doing I/O)

Simple approach: Perform a verification before each checkpoint to eliminate risk of corrupted data.
Motivational Examples

\[ R = 0: \]
\[ W_{\text{ASTE}_{\text{FF}}} = \frac{V + C}{w + V + C}, \quad W_{\text{ASTE}_{\text{Fail}}} = \frac{w}{\mu e} \]

When \( V \) is large compared to \( w \), \( W_{\text{ASTE}_{\text{FF}}} \) is large, can we improve that?
Motivational Examples

\[ R = 0: \]

\[
W_{\text{ASTE}_{\text{FF}}} = \frac{V + C}{w + V + C}, \quad W_{\text{ASTE}_{\text{Fail}}} = \frac{w}{\mu_e}
\]

When \( V \) is large compared to \( w \), \( W_{\text{ASTE}_{\text{FF}}} \) is large, can we improve that?

Is this better?
Motivational Examples

\[ R = 0: \]
\[ W_{\text{ASTE}} = \frac{V + C}{w + V + C}, \quad W_{\text{ASTEFail}} = \frac{w}{\mu e} \]

When \( V \) is small in front of \( w \), \( W_{\text{ASTEFail}} \) is large, can we improve that?
Motivational Examples

\[ R = 0: \]
\[ \text{WASTE}_{\text{FF}} = \frac{V+C}{w+V+C}, \text{WASTE}_{\text{Fail}} = \frac{w}{\mu} \]

When \( V \) is small in front of \( w \), \( \text{WASTE}_{\text{Fail}} \) is large, can we improve that?

Is this better?
With multiple checkpoints, the problem is to find when the error occurred.
With multiple checkpoints, the problem is to find when the error occurred.
With multiple checkpoints, the problem is to find when the error occurred.
With multiple checkpoints, the problem is to find when the error occurred.
$k$ checkpoints for 1 verification

\[
\text{WASTE}_{\text{FF}} = \frac{kC + V}{k(w + C) + V}
\]

\[
\text{WASTE}_{\text{Fail}} = \frac{1}{k} \sum_{i=1}^{k} \frac{T_{\text{lost}}(i)}{\mu_e}
\]

where $T_{\text{lost}}(i)$ is the time lost if error occurred in $i^{th}$ segment.
Silent error detection

Y. Robert

Introduction

Optimal Checkpointing strategy

Exponential distribution

Arbitrary distribution

Limited resources

Incorporating detection

$k$ checkpoints for 1 verification

$k$ verifications for 1 checkpoint

Conclusion, future work

$k$ checkpoints for 1 verification

$T_{lost}(k) = R + V + w + V$
$T_{lost}(k) = R + V + w + V$
$k$ checkpoints for 1 verification

$T_{lost}(k) = R + V + w + V$

The diagram illustrates the time $T_{lost}(k)$ for $k$ checkpoints and $k$ verifications with $C$ for checkpointing, $w$ for write, $V$ for verification, and $R$ for recovery. The error is indicated by the red arrow.
Optimal Checkpointing strategy
Exponential distribution
Arbitrary distribution
Limited resources
Incorporating detection
$k$ checkpoints for 1 verification
$k$ verifications for 1 checkpoint

Conclusion, future work

$k$ checkpoints for 1 verification

$T_{lost}(k) = R + V + w + V$
$k$ checkpoints for 1 verification

\[ T_{\text{lost}}(k) = R + V + w + V \]
\[ T_{\text{lost}}(i) = (k - i + 1)(R + V + w) + (k - i)C + V \]
$k$ checkpoints for 1 verification

$T_{\text{lost}}(k) = R + V + w + V$

$T_{\text{lost}}(i) = (k - i + 1)(R + V + w) + (k - i)C + V$

$T_{\text{lost}}(1) = k(R + V + w) - V + (k - 1)C + V$
$k$ checkpoints for 1 verification

$$T_{\text{lost}}(k) = R + V + w + V$$

$$T_{\text{lost}}(i) = (k - i + 1)(R + V + w) + (k - i)C + V$$

$$T_{\text{lost}}(1) = k(R + V + w) - V + (k - 1)C + V$$

And this leads us to optimal solution ...
Silent error detection

Y. Robert

Introduction

Optimal Checkpointing strategy

Exponential distribution

Arbitrary distribution

Limited resources

Incorporating detection

$k$ checkpoints for 1 verification

$k$ verifications for 1 checkpoint

Conclusion, future work

**Figure:** $V = 100s$, $C = R = 6s$, and $\mu = \frac{10\text{y}}{10^5}$.

$C = 6s \ll V$.

When $V = 100$ seconds, a verification is done only every $k = 3$ checkpoints optimally $\Rightarrow$ 10% improvement compared to $k = 1$. 
C = 60s is not negligible anymore in front of V (V ≈ 5C). The waste is dominated by the cost of verification, and little improvement can be achieved by taking the optimal value for k. 

**Figure**: V = 300s, C = R = 60s, and \( \mu = \frac{10^5}{10^5} \).
Introduction

Optimal Checkpointing strategy

Exponential distribution

Arbitrary distribution

Limited resources

Incorporating detection

$k$ checkpoints for $1$ verification

$k$ verifications for $1$ checkpoint

Conclusion, future work

Figure: $V = 100s$, $C = R = 6s$, and $\mu = \frac{10^5}{10^5}$.

Figure: $V = 300s$, $C = R = 60s$, and $\mu = \frac{10^5}{10^5}$.
Very similarly, we obtain:

\[
WASTE_{FF} = \frac{kV + C}{k(w + V) + C}
\]

\[
WASTE_{Fail} = \frac{1}{k} \sum_{i=1}^{k} \frac{T_{lost}(i)}{\mu_e}
\]

\[
T_{lost}(i) = R + i(V + w)
\]

where \(T_{lost}(i)\) is the time lost if error occurred in \(i^{th}\) segment.
V = 20s ≪ C.
When C = 600 seconds, 5 verifications are done for every checkpoint optimally ⇒ 14% improvement compared to k = 1.

Figure: \( V = 20s, C = R = 600s, \) and \( \mu = \frac{10y}{10^5} \).
\[ V = 2s \ll C. \]

When \( C = 60 \) seconds, 5 verifications are done every checkpoint optimally \( \Rightarrow 18\% \) improvement compared to \( k = 1 \).

**Figure:** \( V = 2s, C = R = 60s, \) and \( \mu = \frac{10y}{10^5} \).
Silent error detection

Y. Robert

Introduction

Optimal Checkpointing strategy

Exponential distribution

Arbitrary distribution

Limited resources

Incorporating detection

$k$ checkpoints for 1 verification

$k$ verifications for 1 checkpoint

Conclusion, future work

Figure: $V = 20s$, $C = R = 600s$, and $\mu = \frac{10y}{10^5}$.

Figure: $V = 2s$, $C = R = 60s$, and $\mu = \frac{10y}{10^5}$.
1. Introduction

2. Optimal Checkpointing strategy
   Exponential distribution
   Arbitrary distribution

3. Limited resources

4. Incorporating detection
   $k$ checkpoints for 1 verification
   $k$ verifications for 1 checkpoint

5. Conclusion, future work
• Study of optimal checkpointing strategy in presence of silent errors
• *Analytical* solution for the different probability distributions
• Study in presence of verification mechanisms
Future work

- **Without verification:** When we keep $k$ checkpoints in memory, we do not have to keep the $k$ last checkpoints: new strategies (Fibonacci, binary, . . .)?

- **With verification:** We focused on an integer number of checkpoints per verification (or conversely): extensions?