On the impact of fault prediction on checkpointing strategies.

Guillaume Aupy\textsuperscript{1}, Yves Robert\textsuperscript{1,2}, Frédéric Vivien\textsuperscript{1}, & Dounia Zaidouni\textsuperscript{1}

1 – ENS Lyon & INRIA
2 – University of Tennessee Knoxville

http://graal.ens-lyon.fr/~yrobert/slides/prediction.pdf.gz

ICL Lunch Talk - October 12, 2012
Overview

Context

- Failure-prone platforms
- Very large number of processors ($N = 16K$ to $N = 1024K$)
- **Fault predictor** characterized by its recall and precision
- Resilience: combine coordinated & preventive checkpointing

Objective

- Design efficient checkpointing policies
- Compute expected **waste**
- Assess impact of predictions
Outline

1. Framework
2. Exact date predictions
3. Prediction windows
4. Experiments
5. Conclusion
Outline

1 Framework

2 Exact date predictions

3 Prediction windows

4 Experiments

5 Conclusion
Predictions

Events

- \( True_P \): Faults that were predicted
- \( False_P \): Predictions that did not materialize as actual faults
- \( False_N \): Faults that were not predicted

Recall: Fraction of faults that are predicted

\[
\begin{align*}
r &= \frac{True_P}{True_P + False_N}
\end{align*}
\]

Precision: Fraction of fault predictions that are correct

\[
\begin{align*}
p &= \frac{True_P}{True_P + False_P}
\end{align*}
\]
Event rates

Mean Times

- Platform MTBF: $\mu = \frac{\mu_{\text{ind}}}{N}$
- Mean time between predictions: $\mu_P$
- Mean time between unpredicted faults: $\mu_{NP}$
- Mean time between events: $\mu_e$

Relations

- $\frac{1-r}{\mu} = \frac{1}{\mu_{NP}}$
- $\frac{r}{\mu} = \frac{p}{\mu_P}$
- $\frac{1}{\mu_e} = \frac{1}{\mu_P} + \frac{1}{\mu_{NP}}$
Hypotheses

Regular (coordinated) checkpoints

- Checkpoint cost: $C$
- Downtime: $D$
- Recovery cost after failure: $R$

Two scenarios

1. Exact date predictions
2. Window-based predictions

Lead times

- Predictions must be provided at least $C$ seconds in advance
Hypotheses

Regular (coordinated) checkpoints
- Checkpoint cost: \( C \)
- Downtime: \( D \)
- Recovery cost after failure: \( R \)

Two scenarios
1. Exact date predictions
2. Window-based predictions

Lead times
- Predictions must be provided at least \( C \) seconds in advance
Outline

1. Framework
2. Exact date predictions
3. Prediction windows
4. Experiments
5. Conclusion
Algorithm

- While no fault prediction is available:
  ⇒ Periodic checkpointing with period $T$
Algorithm

- While no fault prediction is available:
  ⇒ Periodic checkpointing with period $T$
- When a fault is predicted:
  ⇒ Decide whether to take prediction into account or not
Algorithm

- While no fault prediction is available:
  - Periodic checkpointing with period $T$
- When a fault is predicted:
  - Decide whether to take prediction into account or not
  - With probability $1 - q$: ignore prediction
  - With probability $q$: trust prediction
Algorithm

- While no fault prediction is available:
  ⇒ Periodic checkpointing with period $T$
- When a fault is predicted:
  ⇒ Decide whether to take prediction into account or not
    - With probability $1 - q$: ignore prediction
    - With probability $q$: trust prediction
      - If enough time before prediction date, checkpoint ALAP:

```
C
T-C
W_{reg}
T-W_{reg}-C
T-C
```

(a) Predicted failure

given $yves.robert@ens-lyon.fr$
Algorithm

- While no fault prediction is available:
  ⇒ Periodic checkpointing with period $T$
- When a fault is predicted:
  ⇒ Decide whether to take prediction into account or not
    - With probability $1 - q$: ignore prediction
    - With probability $q$: trust prediction
      - If enough time before prediction date, checkpoint ALAP:

\[
\begin{align*}
\text{Predicted failure} \\
\begin{array}{c}
C \quad C \quad C \\
T-C \quad W_{reg} \quad T-W_{reg} - C \quad T-C
\end{array}
\end{align*}
\]

- Otherwise, execute some extra work during $\epsilon$ seconds:

\[
\begin{align*}
\text{Predicted failure} \\
\begin{array}{c}
C \quad C \quad \boxed{\epsilon} \\
T-C \quad T-C \quad T-C
\end{array}
\end{align*}
\]
Expected waste

Waste: fraction of time when processors do not perform useful computations (checkpoints, failures)
**Expected waste**

**Waste:** fraction of time when processors do not perform useful computations (checkpoints, failures)

- **Failure without prediction:**
  - Time: $T-C$
  - $T_{\text{lost}}$
  - Failed

- **Prediction without failure:**
  - Time: $T-C$
  - $W_{\text{reg}}$
  - $T-W_{\text{reg}}$
  - $T-C$

- **Prediction with failure:**
  - Time: $T-C$
  - $W_{\text{reg}}$
  - $T-W_{\text{reg}}$
  - $T-C$

**Actions taken when predictor provides exact dates**
Computing the waste

- Checkpoints: $\frac{C}{T}$
Computing the waste

- Checkpoints: \( \frac{C}{T} \)
- Unpredicted faults: \( \frac{1}{\mu_{NP}} \left[ \frac{T}{2} + D + R \right] \)
Computing the waste

- Checkpoints: \( \frac{C}{T} \)
- Unpredicted faults: \( \frac{1}{\mu_{NP}} \left[ \frac{T}{2} + D + R \right] \)
- Predictions taken into account:
  \[
  \frac{1}{\mu_P} q \left[ p(C + D + R) + (1 - p)C \right]
  \]
Computing the waste

- Checkpoints: \( \frac{C}{T} \)
- Unpredicted faults: \( \frac{1}{\mu_{NP}} \left[ \frac{T}{2} + D + R \right] \)
- Predictions taken into account:

\[
\frac{1}{\mu_p} q \left[ p(C + D + R) + (1 - p)C \right]
\]

- Ignored predictions:

\[
\frac{1}{\mu_p} (1 - q) \left[ p\left( \frac{T}{2} + D + R \right) + (1 - p)0 \right]
\]
Computing the waste

- Checkpoints: $\frac{C}{T}$
- Unpredicted faults: $\frac{1}{\mu_{NP}} \left[ \frac{T}{2} + D + R \right]$
- Predictions taken into account:
  \[ \frac{1}{\mu_P} q \left[ p(C + D + R) + (1 - p)C \right] \]
- Ignored predictions:
  \[ \frac{1}{\mu_P} (1 - q) \left[ p\left( \frac{T}{2} + D + R \right) + (1 - p)0 \right] \]

WASTE = $\frac{C}{T} + \frac{1}{\mu} \left[ (1 - rq) \frac{T}{2} + D + R + \frac{qr}{p} C \right]$
Validity of analysis

Previous equation accurate only if we enforce following conditions:

- \( T < \alpha \mu e \) where \( \alpha = 0.27 \): with probability 0.97, two events do not take place within same period.
Validity of analysis

Previous equation accurate only if we enforce following conditions:

- \( T < \alpha \mu_e \) where \( \alpha = 0.27 \): with probability 0.97, two events do not take place within same period
- Poisson process of parameter \( \beta = \frac{T}{\mu_e} \) (or \( \beta = \frac{T}{\mu} \) without prediction)
- Probability of having \( k \geq 0 \) events: \( P(X = k) = \frac{\beta^k}{k!} e^{-\beta} \)
- Probability of having two or more events: \( \pi = P(X \geq 2) = 1 - (P(X = 0) + P(X = 1)) = 1 - (1 + \beta) e^{-\beta} \)
- \( \alpha = 0.27 \Rightarrow \pi \leq 0.03 \)
Validity of analysis

Previous equation accurate only if we enforce following conditions:

- \( T < \alpha \mu_e \) where \( \alpha = 0.27 \): with probability 0.97, two events do not take place within same period
- Poisson process of parameter \( \beta = \frac{T}{\mu_e} \) (or \( \beta = \frac{T}{\mu} \) without prediction)
- Probability of having \( k \geq 0 \) events: \( P(X = k) = \frac{\beta^k}{k!} e^{-\beta} \)
- Probability of having two or more events:
  \[ \pi = P(X \geq 2) = 1 - (P(X = 0) + P(X = 1)) = 1 - (1 + \beta)e^{-\beta} \]
- \( \alpha = 0.27 \Rightarrow \pi \leq 0.03 \)
- \( C \leq T \): by construction
- \( \text{WASTE} \leq 1 \): by definition
Waste minimization

- $\text{WASTE}(q)$ minimized either for $q = 0$ or for $q = 1$
  - $\text{WASTE}_Y = \text{WASTE}_{\{q=0\}}$ minimized for:
    $$T_Y = \min \left( \alpha \mu, \max(\sqrt{2\mu C}, C) \right)$$
  - $\text{WASTE}_{\{q=1\}}$ minimized for:
    $$T_1 = \min \left( \alpha \mu_e, \max(\sqrt{\frac{2\mu C}{1-r}}, C) \right)$$

Optimal waste:

$$\text{WASTE}_{\text{opt}} = \min \left( \text{WASTE}_Y(T_Y), \text{WASTE}_{\{1\}}(T_1) \right)$$
Strategies

Hypotheses

- Predictor gives a time window for each prediction
- Predictor generates predictions at least $C$ seconds before beginning of time window

Description of strategies

Two modes for scheduling algorithm:

- **Regular:** Periodic checkpointing with period $T_R$
- **Proactive (Several variants):**
  - **INSTANT:** Ignore time-window ($\Leftrightarrow$ exact date)
  - **NOCKPTI:** No checkpoint during time window
  - **WITHCKPTI:** Several checkpoints during time window
Algorithm 1: WithCkptI.

1 if fault happens then
2     After downtime, execute recovery;
3     Enter regular mode;
4 if in proactive mode for a time greater than or equal to I then
5     Switch to regular mode
6 if Prediction made with interval \([t, t+I]\) and prediction taken into account then
7     Let \(t_C\) be the date of the last checkpoint under regular mode to start no later than \(t - C\);
8     if \(t_C + C < t - C\) then (enough time for an extra checkpoint)
9         Take a checkpoint starting at time \(t - C\)
10    else (no time for the extra checkpoint)
11         Work in the time interval \([t_C + C, t]\]
12         \(W_{reg} \leftarrow \max(0, t - C - (t_C + C))\);
13         Switch to proactive mode at time \(t\);
14 while in regular mode and no predictions are made and no faults happen do
15     Work for a time \(T_R - W_{reg} - C\) and then checkpoint;
16     \(W_{reg} \leftarrow 0\);
17 while in proactive mode and no faults happen do
18     Work for a time \(T_P - C\) and then checkpoint;
Outline of Algorithm 1 (strategy WITHCKPTI)
Waste due to periodic checkpointing:

- Regular mode: \( \left(1 - \frac{l'}{\mu_P}\right) \frac{C}{T_R} \) and Proactive mode: \( \frac{l'}{\mu_P} \frac{C}{T_P} \)
Waste due to periodic checkpointing:
- Regular mode: \( \left(1 - \frac{l'}{\mu_P}\right) \frac{C}{T_R} \) and Proactive mode: \( \frac{l'}{\mu_P} \frac{C}{T_P} \)

Waste incurred when switching to proactive mode: \( \frac{q}{\mu_P} C \)
Waste for strategy **WITHCkptI** (1/2)

- **Waste due to periodic checkpointing:**
  - Regular mode: \( \left( 1 - \frac{l'}{\mu_P} \right) \frac{C}{T_R} \) and Proactive mode: \( \frac{l'}{\mu_P} \frac{C}{T_P} \)
  - **Waste incurred when switching to proactive mode:** \( \frac{q}{\mu_P} C \)
  - **Waste due to predicted faults:**
    - Regular mode: \( \frac{p(1-q)}{\mu_P} \left( \frac{T_R}{2} + D + R \right) \)
    - Proactive mode: \( \frac{qp}{\mu_P} (T_P + D + R) \)
Waste due to periodic checkpointing:
- Regular mode: \( \left(1 - \frac{l'}{\mu_p}\right) \frac{C}{T_R} \) and Proactive mode: \( \frac{l'}{\mu_p} \frac{C}{T_p} \)
- Waste incurred when switching to proactive mode: \( \frac{q}{\mu_p} C \)

Waste due to predicted faults:
- Regular mode: \( \frac{p(1-q)}{\mu_p} \left( \frac{T_R}{2} + D + R \right) \)
- Proactive mode: \( \frac{qp}{\mu_p} \left( T_P + D + R \right) \)

Waste due to unpredicted faults:
- Regular mode: \( \left(1 - \frac{l'}{\mu_p}\right) \frac{1}{\mu_{NP}} \frac{T_R}{2} \)
- Proactive mode: \( \frac{l'}{\mu_p \mu_{NP}} \left( T_P + D + R \right) \)
Waste for strategy $\text{WITHCkptI} \ (2/2)$

\[
\text{WASTE}_{\text{WITHCkptI}} = \left( \left( 1 - \frac{l'}{\mu_P} \right) \frac{1}{T_R} + \frac{l'}{\mu_P} \frac{1}{T_P} + \frac{q}{\mu_P} \right) C
\]
\[
+ \frac{p(1-q)}{\mu_P} \frac{T_R}{2} + \frac{pq}{\mu_P} T_P + \left( 1 - \frac{l'}{\mu_P} \right) \frac{1}{\mu_{NP}} \frac{T_R}{2}
\]
\[
+ \left( \frac{p}{\mu_P} + \left( 1 - \frac{l'}{\mu_P} \right) \frac{1}{\mu_{NP}} \right) (D + R)
\]

**Validity:** $T + l < \alpha \mu_e$
**Waste minimization**

- \( \text{WASTE}(q) \) minimized either for \( q = 0 \) or for \( q = 1 \)
- \( \text{WASTE}_{\text{WITH} \text{CKPT} I}^{\{q=0\}} \) minimized for:
  \[
  T_Y = \min \left( \alpha \mu, \max(\sqrt{2\mu C}, C) \right)
  \]
- \( \text{WASTE}_{\text{WITH} \text{CKPT} I}^{\{q=1\}} \) minimized for:
  \[
  T_{R}^{\text{opt} 1} = \min \left( \alpha \mu_e - I, \max \left( \sqrt{\frac{2\mu C}{1-r}}, C \right) \right)
  \]

**Optimal waste:**

\[
\text{WASTE}_{\text{opt}} = \min \left( \text{WASTE}_{\text{WITH} \text{CKPT} I}^{\{0\}}(T_Y), \text{WASTE}_{\text{WITH} \text{CKPT} I}^{\{1\}}(T_{R}^{\text{opt} 1}) \right)
\]
Heuristics

- **YOUNG** periodic checkpointing strategy
  (period $T_{\text{extr}}^{\{0\}} = \sqrt{2\mu C}$)

- **EXACTPREDICTION** strategy with exact prediction dates
  (uncapped period $T_{\text{extr}}^{\{1\}} = \sqrt{\frac{2\mu C}{1 - r}}$)

- **INSTANT, NOCKPTI, WITHCKPTI** window-based strategies
  (uncapped period $T_{\text{extr}}^{\{1\}} = \sqrt{\frac{2\mu C}{1 - r}}$)
Prediction and failure distributions

- Failure traces (predicted and non predicted failures):
  - Exponential failure distribution
  - Weibull distribution law with shape parameter 0.5 and 0.7
- False predictions:
  - Same distribution as failure trace
  - Uniform distribution

<table>
<thead>
<tr>
<th>Number of processors</th>
<th>$D$</th>
<th>$C,R$</th>
<th>$\mu_{\text{ind}}$</th>
<th>$W$</th>
</tr>
</thead>
<tbody>
<tr>
<td>16,384 to 524,288</td>
<td>60 s</td>
<td>600 s</td>
<td>125 y</td>
<td>10,000 y</td>
</tr>
</tbody>
</table>

Simulation parameters
Comparing job execution times for a Weibull distribution \((k = 0.7)\), and reporting gain when comparing to YOUNG.
## Job execution times for a Weibull distribution ($k = 0.7$)

### $I = 300$

<table>
<thead>
<tr>
<th></th>
<th>Execution time (in days) ($p = 0.82, r = 0.85$)</th>
<th>Execution time (in days) ($p = 0.4, r = 0.7$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$2^{16}$ procs</td>
<td>$2^{19}$ procs</td>
</tr>
<tr>
<td><strong>Young</strong></td>
<td>81.3</td>
<td>30.1</td>
</tr>
<tr>
<td><strong>ExactPrediction</strong></td>
<td>65.9 (19%)</td>
<td>15.9 (47%)</td>
</tr>
<tr>
<td><strong>NoCkptI</strong></td>
<td>66.5 (18%)</td>
<td>16.9 (44%)</td>
</tr>
<tr>
<td><strong>Instant</strong></td>
<td>66.5 (18%)</td>
<td>17.0 (44%)</td>
</tr>
</tbody>
</table>

### $I = 3,000$

<table>
<thead>
<tr>
<th></th>
<th>Execution time (in days) ($p = 0.82, r = 0.85$)</th>
<th>Execution time (in days) ($p = 0.4, r = 0.7$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$2^{16}$ procs</td>
<td>$2^{19}$ procs</td>
</tr>
<tr>
<td><strong>Young</strong></td>
<td>81.2</td>
<td>30.1</td>
</tr>
<tr>
<td><strong>ExactPrediction</strong></td>
<td>66.0 (19%)</td>
<td>15.9 (47%)</td>
</tr>
<tr>
<td><strong>NoCkptI</strong></td>
<td>71.1 (12%)</td>
<td>24.6 (18%)</td>
</tr>
<tr>
<td><strong>WithCkptI</strong></td>
<td>70.0 (14%)</td>
<td>22.6 (25%)</td>
</tr>
<tr>
<td><strong>Instant</strong></td>
<td>71.2 (12%)</td>
<td>24.2 (20%)</td>
</tr>
</tbody>
</table>

Comparing job execution times for a Weibull distribution ($k = 0.7$), and reporting gain when comparing to **Young**.
### Job execution times for a Weibull distribution ($k = 0.5$)

<table>
<thead>
<tr>
<th>$I = 300$</th>
<th>Execution time (in days)</th>
<th>$I = 3,000$</th>
<th>Execution time (in days)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>($p = 0.82, r = 0.85$)</td>
<td>($p = 0.82, r = 0.85$)</td>
<td>($p = 0.4, r = 0.7$)</td>
</tr>
<tr>
<td></td>
<td>$2^{16}$ procs</td>
<td>$2^{19}$ procs</td>
<td>$2^{16}$ procs</td>
</tr>
<tr>
<td>Young</td>
<td>125.4</td>
<td>171.8</td>
<td>125.4</td>
</tr>
<tr>
<td>ExactPrediction</td>
<td>75.8 (40%)</td>
<td>39.4 (77%)</td>
<td>82.9 (34%)</td>
</tr>
<tr>
<td>NoCkptI</td>
<td>77.3 (38%)</td>
<td>44.8 (74%)</td>
<td>84.6 (33%)</td>
</tr>
<tr>
<td>Instant</td>
<td>77.4 (38%)</td>
<td>45.1 (74%)</td>
<td>84.7 (33%)</td>
</tr>
</tbody>
</table>

### Comparing job execution times for a Weibull distribution ($k = 0.5$), and reporting gain when comparing to Young.
## Job execution times for a Weibull distribution \((k = 0.5)\)

<table>
<thead>
<tr>
<th>(I = 300)</th>
<th>Execution time (in days) ((p = 0.82, r = 0.85))</th>
<th>Execution time (in days) ((p = 0.4, r = 0.7))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2(^{16}) procs</td>
<td>2(^{19}) procs</td>
</tr>
<tr>
<td><strong>Young</strong></td>
<td>125.4</td>
<td>171.8</td>
</tr>
<tr>
<td><strong>ExactPrediction</strong></td>
<td>75.8 (40%)</td>
<td>39.4 (77%)</td>
</tr>
<tr>
<td><strong>NoCkptI</strong></td>
<td>77.3 (38%)</td>
<td>44.8 (74%)</td>
</tr>
<tr>
<td><strong>Instant</strong></td>
<td>77.4 (38%)</td>
<td>45.1 (74%)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(I = 3,000)</th>
<th>Execution time (in days) ((p = 0.82, r = 0.85))</th>
<th>Execution time (in days) ((p = 0.4, r = 0.7))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2(^{16}) procs</td>
<td>2(^{19}) procs</td>
</tr>
<tr>
<td><strong>Young</strong></td>
<td>125.4</td>
<td>171.9</td>
</tr>
<tr>
<td><strong>ExactPrediction</strong></td>
<td>76.1 (39%)</td>
<td>39.4 (77%)</td>
</tr>
<tr>
<td><strong>NoCkptI</strong></td>
<td>90.0 (28%)</td>
<td>71.8 (58%)</td>
</tr>
<tr>
<td><strong>WithCkptI</strong></td>
<td>87.8 (30%)</td>
<td>66.6 (61%)</td>
</tr>
<tr>
<td><strong>Instant</strong></td>
<td>89.8 (28%)</td>
<td>70.9 (59%)</td>
</tr>
</tbody>
</table>

Comparing job execution times for a Weibull distribution \((k = 0.5)\), and reporting gain when comparing to **Young**.
Waste with $p = 0.82$, $r = 0.85$ and $I = 300s$

(a) Capped period
(b) Uncapped period
(c) Exponential
(d) Weibull $k = 0.7$
(e) Weibull $k = 0.5$
Waste with $p = 0.82$, $r = 0.85$ and $I = 3000s$
Waste with $p = 0.4$, $r = 0.7$ and $l = 300s$

(a) Capped period
(b) Uncapped period
(c) Exponential
(d) Weibull $k = 0.7$
(e) Weibull $k = 0.5$
Waste with $p = 0.4$, $r = 0.7$ and $I = 3000s$

(a) Capped period

(b) Uncapped period

(c) Exponential

(d) Weibull $k = 0.7$

(e) Weibull $k = 0.5$
Impact of precision for a fixed recall

(a) $r = 0.4$, $N = 2^{16}$
(b) $r = 0.4$, $N = 2^{19}$
(c) $r = 0.8$, $N = 2^{16}$
(d) $r = 0.8$, $N = 2^{19}$
Impact of recall for a fixed precision

(a) $p = 0.4, N = 2^{16}$

(b) $p = 0.4, N = 2^{19}$

(c) $p = 0.8, N = 2^{16}$

(d) $p = 0.8, N = 2^{19}$

yves.robert@ens-lyon.fr
Conclusion and perspectives

- Model is quite accurate

- Unified formula for optimal checkpointing period: \( \sqrt{\frac{2\mu C}{1 - rq}} \)

- Simulations fully validate the model:
  - Significant gain even for mid-range recall and precision
  - Best period always very close to one given by unified formula

- Recall has more impact on waste than precision

- Future work
  Use trace-based failure and prediction logs from current large-scale supercomputers