Checkpointing strategies for parallel jobs

Marin Bougeret, Henri Casanova, Mikaël Rabie, Yves Robert, and Frédéric Vivien

ENS Lyon & INRIA, France
University of Hawai‘i at Mānoa, USA
University of Montpellier, France
Motivation

Framework

- **Very very** large number of processing elements (e.g., $2^{20}$)
- Failure-prone platform (like any realistic platform)
- Large application to be executed on the whole platform

$\Rightarrow$ Failure(s) will certainly occur before completion!

- Resilience provided through coordinated checkpointing

Question

- When should we checkpoint the application?
One knows that applications should be checkpointed periodically.

Several proposed values for period:

- Young: \( \sqrt{2 \times C \times \text{MTBF}} \) (1st order approximation)

- Daly (1): \( \sqrt{2 \times C \times (R + \text{MTBF})} \) (1st order approximation)

- Daly (2): \( \eta \times \text{MTBF} - C \), where \( \eta = \xi + 1 + L(z) \) (higher order approximation)

How good are these approximations? Could we find the optimal value? At least for Exponential failures? And for Weibull failures?
One knows that applications should be checkpointed periodically
Is this optimal?

Several proposed values for period:

Young:
\[ \sqrt{2 \times C \times MTBF} \] (1st order approximation)

Daly (1):
\[ \sqrt{2 \times C \times (R + MTBF)} \] (1st order approximation)

Daly (2):
\[ \eta \times MTBF - C \], where
\[ \eta = \sqrt{C^2 \times MTBF} \]

\[ \xi = \sqrt{C^2 \times MTBF} \]

\[ L(z) \times e^{L(z)} = z \] (higher order approximation)

How good are these approximations?
Could we find the optimal value? At least for Exponential failures?
And for Weibull failures?
State of the art

One knows that applications should be checkpointed periodically. Is this optimal?

Several proposed values for period

- Young: \( \sqrt{2 \times C \times MTBF} \) (1st order approximation)
- Daly (1): \( \sqrt{2 \times C \times (R + MTBF)} \) (1st order approximation)
- Daly (2): \( \eta \times MTBF - C \), where \( \eta = \xi^2 + 1 + \mathbb{I}(\text{e}^{-(2\xi^2+1)}) \), \( \xi = \sqrt{\frac{C}{2 \times MTBF}} \), and \( \mathbb{I}(z) \text{e}^{\mathbb{I}(z)} = z \) (higher order approximation)
State of the art

One knows that applications should be checkpointed periodically. Is this optimal?

Several proposed values for period

- Young: $\sqrt{2 \times C \times \text{MTBF}}$ (1st order approximation)
- Daly (1): $\sqrt{2 \times C \times (R + \text{MTBF})}$ (1st order approximation)
- Daly (2): $\eta \times \text{MTBF} - C$, where $\eta = \xi^2 + 1 + \mathbb{I}(-e^{-(2\xi^2+1)})$, $\xi = \sqrt{\frac{C}{2\times\text{MTBF}}}$, and $\mathbb{I}(z)e^{\mathbb{I}(z)} = z$ (higher order approximation)

How good are these approximations? Could we find the optimal value? At least for Exponential failures? And for Weibull failures?
Outline

1 Single-processor jobs
   - Solving **MAKESPAN**
   - Solving **NEXTFAILURE**

2 Parallel jobs
   - Solving **MAKESPAN**
   - Solving **NEXTFAILURE**

3 Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4 Conclusion
Outline

1. Single-processor jobs
   - Solving **Makespan**
   - Solving **NextFailure**

2. Parallel jobs
   - Solving **Makespan**
   - Solving **NextFailure**

3. Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4. Conclusion
Hypotheses

- Overall size of work: $\mathcal{W}$
- Checkpoint cost: $C$ (e.g., write on disk the contents of each processor memory)
- Downtime: $D$ (hardware replacement by spare, or software rejuvenation via rebooting)
- Recovery cost after failure: $R$
- Homogeneous platform (same computation speeds, iid failure distributions)
- History of failures has no impact, only the time elapsed since last failure does
- A failure can happen during a checkpoint, a recovery, but not a downtime (otherwise replace $D$ by 0 and $R$ by $R + D$).
Outline

1. Single-processor jobs
   - Solving Makespan
   - Solving NextFailure

2. Parallel jobs
   - Solving Makespan
   - Solving NextFailure

3. Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4. Conclusion
Problem statement

**Makespan**

- Minimize the job’s expected makespan, that is:
  - the expectation $\mathbb{E}$
  - of the time $T$ needed to process
  - a work of size $\mathcal{W}$
  - knowing that the (single) processor failed $\tau$ units of time ago.

- Notation:
  - minimize $\mathbb{E}(T(\mathcal{W}|\tau))$
  - $\omega_1(\mathcal{W}|\tau)$: amount of work we *attempt* to do before taking the first checkpoint
Recursive approach

\[ E(T(W|\tau)) = \]
Recursive approach

Probability of success

\[ \mathcal{P}_{\text{suc}}(\omega_1 + C|\tau)(\omega_1 + C + \mathbb{E}(T(W - \omega_1|\tau + \omega_1 + C)) \]

\[ \mathbb{E}(T(W|\tau)) = \]
Recursive approach

Time needed to compute the 1st chunk

\[ P_{\text{succ}}(\omega_1 + C|\tau)(\omega_1 + C + \mathbb{E}(T(W - \omega_1|\tau + \omega_1 + C)) \]

\[ \mathbb{E}(T(W|\tau)) = \]
Recursive approach

$$\mathbb{E}(T(W|\tau)) = \mathcal{P}_{\text{succ}}(\omega_1 + C|\tau)(\omega_1 + C + \mathbb{E}(T(W - \omega_1|\tau + \omega_1 + C))$$
Recursive approach

\[ P_{\text{succ}}(\omega_1 + C | \tau)(\omega_1 + C + \mathbb{E}(T(W - \omega_1|\tau + \omega_1 + C)) \]
\[ \mathbb{E}(T(W|\tau)) = \]
\[ + (1 - P_{\text{succ}}(\omega_1 + C | \tau))(\mathbb{E}(T_{\text{lost}}(\omega_1 + C|\tau)) + \mathbb{E}(T_{\text{rec}}) + \mathbb{E}(T(W|R))) \]
Recursive approach

\[ \mathbb{E}(T(W|\tau)) = P_{\text{succ}}(\omega_1 + C|\tau)(\omega_1 + C + \mathbb{E}(T(W - \omega_1|\tau + \omega_1 + C)) \]

\[ + (1 - P_{\text{succ}}(\omega_1 + C|\tau)) (\mathbb{E}(T_{\text{lost}}(\omega_1 + C|\tau)) + \mathbb{E}(T_{\text{rec}}) + \mathbb{E}(T(W|R))) \]

Probability of failure
Recursive approach

\[ E(T(W|\tau)) = P_{\text{succ}}(\omega_1 + C|\tau)(\omega_1 + C + E(T(W - \omega_1|\tau + \omega_1 + C)) \]

\[ + (1 - P_{\text{succ}}(\omega_1 + C|\tau))(E(T_{\text{lost}}(\omega_1 + C|\tau)) + E(T_{\text{rec}}) + E(T(W|R)))) \]

Time elapsed before the failure occurred
Recursive approach

\[
E(T(W|\tau)) = P_{\text{succ}}(\omega_1 + C|\tau)(\omega_1 + C + E(T(W - \omega_1|\tau + \omega_1 + C)) + \\
(1 - P_{\text{succ}}(\omega_1 + C|\tau))(E(T_{\text{lost}}(\omega_1 + C|\tau)) + E(T_{\text{rec}}) + E(T(W|R)))
\]

Time needed to perform downtime and recovery
Recursive approach

\[ \mathbb{P}_{\text{suc}}(\omega_1 + C|\tau) (\omega_1 + C + \mathbb{E}(T(W - \omega_1|\tau + \omega_1 + C)) \]

\[ \mathbb{E}(T(W|\tau)) = + \]

\[ (1 - \mathbb{P}_{\text{suc}}(\omega_1 + C|\tau)) (\mathbb{E}(T_{\text{lost}}(\omega_1 + C|\tau)) + \mathbb{E}(T_{\text{rec}}) + \mathbb{E}(T(W|R))) \]

Time needed
to compute \( W \)
from scratch
Recursive approach

\[ P_{\text{succ}}(\omega_1 + C|\tau) (\omega_1 + C + \mathbb{E}(T(W - \omega_1|\tau + \omega_1 + C)) \]

\[ \mathbb{E}(T(W|\tau)) = + \]

\[ (1 - P_{\text{succ}}(\omega_1 + C|\tau)) (\mathbb{E}(T_{\text{lost}}(\omega_1 + C|\tau)) + \mathbb{E}(T_{\text{rec}}) + \mathbb{E}(T(W|R))) \]

Problem: finding \( \omega_1(W, \tau) \) minimizing \( \mathbb{E}(T(W|\tau)) \)
Failures following an exponential distribution

Theorem

Optimal strategy splits $\mathcal{W}$ into $K^*$ same-size chunks where

$$K^* = \max(1, \lfloor K_0 \rfloor) \text{ or } K^* = \lceil K_0 \rceil$$

(whichever leads to the smaller value)

where

$$K_0 = \frac{\lambda \mathcal{W}}{1 + \mathbb{L}(-e^{-\lambda C - 1})} \text{ and } \mathbb{L}(z)e^{\mathbb{L}(z)} = z$$

Optimal expectation of makespan is

$$K^* \left( e^{\lambda R} \left( \frac{1}{\lambda} + D \right) \right) \left( e^{\lambda \left( \frac{\mathcal{W}}{K^*} + C \right)} - 1 \right)$$
Arbitrary failure distributions

\[ E(T(W|\tau)) = \min_{0 < \omega_1 \leq W} \left( P_{suc}(\omega_1 + C|\tau)(\omega_1 + C + E(T(W - \omega_1|\tau + \omega_1 + C)) \right. \\
\left. + (1 - P_{suc}(\omega_1 + C|\tau)) \times \right. \\
\left. \left( E(T_{lost}(\omega_1 + C|\tau)) + E(T_{rec}) + E(T(W|R)) \right) \right) \]

Solve via dynamic programming

- Time quantum \( u \): all chunk sizes \( \omega_i \) are integer multiples of \( u \)
- Trade-off: accuracy versus higher computing time
Algorithm 1: DPMakespan \((x, b, y, \tau_0)\)

if \(x = 0\) then
  return 0

if solution\([x][b][y]\) = unknown then
  best \(\leftarrow\) \(\infty\);
  \(\tau \leftarrow b\tau_0 + yu\)
  for \(i = 1\) to \(x\) do
    exp_suc \(\leftarrow\) first(DPMakespan\((x - i, b, y + i + \frac{C}{u}, \tau_0)\))
    exp_fai \(\leftarrow\) first(DPMakespan\((x, 0, \frac{R}{u}, \tau_0)\))
    cur \(\leftarrow\) \(P_{suc}(iu + C|\tau)(iu + C + \exp_{suc})\)
    \(+ (1 - P_{suc}(iu + C|\tau))(E(T_{lost}(iu + C, \tau)) \)
    \(\quad + E(T_{rec}) + \exp_{fai})\)
    if cur < best then
      best \(\leftarrow\) cur;
      chunksize \(\leftarrow\) i
      solution\([x][b][y]\) \(\leftarrow\) (best, chunksize)
  return solution\([x][b][y]\)
1. Single-processor jobs
   - Solving MAKESPAN
   - Solving NEXTFAILURE

2. Parallel jobs
   - Solving MAKESPAN
   - Solving NEXTFAILURE

3. Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4. Conclusion
Problem statement

**NextFailure**

- Maximize expected amount of work completed before next failure
- Optimization on a “failure-by-failure” basis
- Hopefully a good approximation, at least for large job sizes $\mathcal{W}$
\[
\mathbb{E}(W(\omega|\tau)) = P_{\text{suc}}(\omega_1 + C|\tau)(\omega_1 + \mathbb{E}(W(\omega - \omega_1|\tau + \omega_1 + C)))
\]

Proposition

\[
\mathbb{E}(W(W|0)) = \sum_{i=1}^{K} \omega_i \times \prod_{j=1}^{i} P_{\text{suc}}(\omega_j + C|t_j)
\]

where \(t_j = \sum_{\ell=1}^{j-1} \omega_\ell + C\) is the total time elapsed (without failure) before execution of chunk \(\omega_1\), and \(K\) is the (unknown) target number of chunks.
Solving through dynamic programming

Algorithm 2: \textsc{DPNextFailure} \((x, n, \tau_0)\)

\begin{verbatim}
if \(x = 0\) then
  return 0
if \(\text{solution}[x][n] = \text{unknown}\) then
  \(\text{best} \leftarrow \infty\)
  \(\tau \leftarrow \tau_0 + (\mathcal{W} - xu) + nC\)
  \textbf{for} \(i = 1\) to \(x\) \textbf{do}
    \(\text{work} = \text{first}(\text{DPNextFailure}(x - i, n + 1, \tau_0))\)
    \(\text{cur} \leftarrow P_{\text{suc}}(iu + C|\tau) \times (iu + \text{work})\)
    \textbf{if} \(\text{cur} < \text{best}\) \textbf{then}
      \(\text{best} \leftarrow \text{cur}; \quad \text{chunksize} \leftarrow i\)
    \(\text{solution}[x][n] \leftarrow (\text{best}, \text{chunksize})\)
\end{verbatim}

return \(\text{solution}[x][n]\)
Outline

1. Single-processor jobs
   - Solving \textit{Makespan}
   - Solving \textit{NextFailure}

2. Parallel jobs
   - Solving \textit{Makespan}
   - Solving \textit{NextFailure}

3. Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4. Conclusion
Outline

1 Single-processor jobs
   - Solving MAKESPAN
   - Solving NEXTFAILURE

2 Parallel jobs
   - Solving MAKESPAN
   - Solving NEXTFAILURE

3 Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4 Conclusion
Failures following an exponential distribution

**Theorem**

*Optimal strategy splits $\mathcal{W}(p)$ in $K^*(p)$ same-size chunks where*

$$K^*(p) = \max(1, \lfloor K_0(p) \rfloor) \text{ or } K^*(p) = \lceil K_0(p) \rceil$$

*(whichever leads to the smaller value)*

*where $K_0(p) = \frac{\lambda \mathcal{W}(p)}{1 + \mathbb{I}(-e^{-p\lambda C} - 1)}$ and $\mathbb{I}(z)e^{\mathbb{I}(z)} = z$*

**Optimal expectation of makespan is**

$$K^*(p) \left( \frac{1}{p\lambda} + \mathbb{E}(T_{rec}(p)) \right) \left( e^{\lambda \left( \frac{\mathcal{W}}{K^*(p)} + pC \right)} - 1 \right)$$
Arbitrary failure distributions

- Cannot solve analytically the recursion
- Cannot extend the dynamic programming algorithm \textsc{DPMakespan} designed for the single-processor case:
  - Would need to memorize all possible failure scenarios for each processor
  - Number of states exponential in $p$
Outline

1. Single-processor jobs
   - Solving **MAKESPAN**
   - Solving **NEXTFAILURE**

2. Parallel jobs
   - Solving **MAKESPAN**
   - Solving **NEXTFAILURE**

3. Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4. Conclusion
Dynamic programming

All $\tau$ variables evolve identically: recursive calls only correspond to cases in which no failure has occurred.

$$\mathbb{E}(W(W|\tau_1, \ldots, \tau_p)) = P_{suc}(\omega_1 + C|\tau_1, \ldots, \tau_p)(\omega_1 + \mathbb{E}(W(W-\omega_1|\tau_1+\omega_1+C, \ldots, \tau_p+\omega_1+C)))$$

$\Rightarrow$ Same dynamic programming approach than previously

- Linear dependency in $p$ (computation of $P_{suc}$)
- Reduce complexity by recording only $x$ most recent $\tau$ values and approximate the other values using $y$ rounding values defined by $x$ regularly-spaced quantiles
Outline

1. Single-processor jobs
   - Solving MAKESPAN
   - Solving NEXTFAILURE

2. Parallel jobs
   - Solving MAKESPAN
   - Solving NEXTFAILURE

3. Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4. Conclusion
Outline

1. Single-processor jobs
   - Solving Makespan
   - Solving NextFailure

2. Parallel jobs
   - Solving Makespan
   - Solving NextFailure

3. Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4. Conclusion
Evaluated approaches

Heuristics

- Young [4]
- DalyLow [2]
- DalyHigh [2]
- Bouguerra [1]
- Liu [3]
- OptExp
- DPMakespan
- DPNextFailure

Theoretical bounds

- LowerBound (omniscient algorithm)
- PeriodLB
## Synthetic failure distributions

Simulation parameters

<table>
<thead>
<tr>
<th></th>
<th>$p_{total}$</th>
<th>$D$</th>
<th>$C,R$</th>
<th>$MTBF$</th>
<th>$W$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-proc</td>
<td>1</td>
<td>60 s</td>
<td>600 s</td>
<td>1 h, 1 d, 1 w</td>
<td>20 d</td>
</tr>
<tr>
<td>Petascale</td>
<td>45, 208</td>
<td>60 s</td>
<td>600 s</td>
<td>125 y, 500 y</td>
<td>1,000 y</td>
</tr>
<tr>
<td>Exascale</td>
<td>$2^{20}$</td>
<td>60 s</td>
<td>600 s</td>
<td>1250 y</td>
<td>10,000 y</td>
</tr>
</tbody>
</table>
Outline

1. Single-processor jobs
   - Solving \textsc{Makespan}
   - Solving \textsc{NextFailure}

2. Parallel jobs
   - Solving \textsc{Makespan}
   - Solving \textsc{NextFailure}

3. Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4. Conclusion
Sequential jobs under Exponential failures

<table>
<thead>
<tr>
<th>Heuristics</th>
<th>MTBF 1 hour</th>
<th>MTBF 1 day</th>
<th>MTBF 1 week</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOWERBOUND</td>
<td>0.62865</td>
<td>0.90714</td>
<td>0.979151</td>
</tr>
<tr>
<td>PERIODLB</td>
<td>1.00705</td>
<td>1.01588</td>
<td>1.02298</td>
</tr>
<tr>
<td>YOUNG</td>
<td>1.01635</td>
<td>1.01590</td>
<td>1.02332</td>
</tr>
<tr>
<td>DALYLOW</td>
<td>1.02711</td>
<td>1.01611</td>
<td>1.02338</td>
</tr>
<tr>
<td>DALYHIGH</td>
<td>1.00700</td>
<td>1.01592</td>
<td>1.02373</td>
</tr>
<tr>
<td>LIU</td>
<td>1.01607</td>
<td>1.01655</td>
<td>1.02333</td>
</tr>
<tr>
<td>BOUGUERRA</td>
<td>1.02562</td>
<td>1.02329</td>
<td>1.02685</td>
</tr>
<tr>
<td>OPTEXP</td>
<td>1.00705</td>
<td>1.01611</td>
<td>1.02298</td>
</tr>
<tr>
<td>DPNEXTFAILURE</td>
<td>1.00785</td>
<td>1.01699</td>
<td>1.02851</td>
</tr>
<tr>
<td>DPMAKESPAN</td>
<td>1.00737</td>
<td>1.01655</td>
<td>1.03467</td>
</tr>
</tbody>
</table>

Degradation from best, single processor, Exponential failures
## Sequential jobs under Exponential failures

<table>
<thead>
<tr>
<th>Heuristics</th>
<th>MTBF 1 hour</th>
<th>MTBF 1 day</th>
<th>MTBF 1 week</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LOWERBOUND</strong></td>
<td>0.62865</td>
<td>0.90714</td>
<td>0.979151</td>
</tr>
<tr>
<td><strong>PERIODLB</strong></td>
<td>1.00705</td>
<td>1.01588</td>
<td>1.02298</td>
</tr>
<tr>
<td><strong>YOUNG</strong></td>
<td>1.01635</td>
<td>1.01590</td>
<td>1.02332</td>
</tr>
<tr>
<td><strong>DALYLOW</strong></td>
<td>1.02711</td>
<td>1.01611</td>
<td>1.02338</td>
</tr>
<tr>
<td><strong>DALYHIGH</strong></td>
<td>1.00700</td>
<td>1.01592</td>
<td>1.02373</td>
</tr>
<tr>
<td><strong>Liu</strong></td>
<td>1.01607</td>
<td>1.01655</td>
<td>1.02333</td>
</tr>
<tr>
<td><strong>BOUGUERRA</strong></td>
<td>1.02562</td>
<td>1.02329</td>
<td>1.02685</td>
</tr>
<tr>
<td><strong>OptExp</strong></td>
<td>1.00705</td>
<td>1.01611</td>
<td>1.02298</td>
</tr>
<tr>
<td><strong>DPNextFailure</strong></td>
<td>1.00785</td>
<td>1.01699</td>
<td>1.02851</td>
</tr>
<tr>
<td><strong>DPMakespan</strong></td>
<td>1.00737</td>
<td>1.01655</td>
<td>1.03467</td>
</tr>
</tbody>
</table>

Degradation from best, single processor, **Exponential** failures
Sequential jobs under Weibull failures

<table>
<thead>
<tr>
<th>Heuristics</th>
<th>1 hour</th>
<th>1 day</th>
<th>1 week</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOWERBOUND</td>
<td>0.66417</td>
<td>0.90714</td>
<td>0.97915</td>
</tr>
<tr>
<td>PERIODLB</td>
<td>1.00960</td>
<td>1.01588</td>
<td>1.02298</td>
</tr>
<tr>
<td>YOUNG</td>
<td>1.00965</td>
<td>1.01590</td>
<td>1.02332</td>
</tr>
<tr>
<td>DALYLOW</td>
<td>1.01155</td>
<td>1.01611</td>
<td>1.02338</td>
</tr>
<tr>
<td>DALYHIGH</td>
<td>1.01785</td>
<td>1.01592</td>
<td>1.02373</td>
</tr>
<tr>
<td>LIU</td>
<td>1.00914</td>
<td>1.01655</td>
<td>1.02333</td>
</tr>
<tr>
<td>BOUGUERRA</td>
<td>1.02936</td>
<td>1.02329</td>
<td>1.02685</td>
</tr>
<tr>
<td>OPTEXP</td>
<td>1.01788</td>
<td>1.01611</td>
<td>1.02298</td>
</tr>
<tr>
<td>DPNEXTFailure</td>
<td>1.01408</td>
<td>1.01699</td>
<td>1.02851</td>
</tr>
<tr>
<td>DPMakespan</td>
<td>1.00731</td>
<td>1.01655</td>
<td>1.03467</td>
</tr>
</tbody>
</table>

Degradation from best, single processor, Weibull failures
Outline

1 Single-processor jobs
   - Solving \textsc{Makespan}
   - Solving \textsc{NextFailure}

2 Parallel jobs
   - Solving \textsc{Makespan}
   - Solving \textsc{NextFailure}

3 Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4 Conclusion
Parallel jobs under Exponential failures (1/2)

Petascale, MTBF = 125 years
Parallel jobs under Exponential failures (1/2)

Petascale, MTBF = 125 years
Parallel jobs under Exponential failures (1/2)

<table>
<thead>
<tr>
<th>number of processors</th>
<th>average makespan degradation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^{10}$</td>
<td>0.97</td>
</tr>
<tr>
<td>$2^{11}$</td>
<td>0.93</td>
</tr>
<tr>
<td>$2^{12}$</td>
<td>0.90</td>
</tr>
<tr>
<td>$2^{13}$</td>
<td>0.87</td>
</tr>
<tr>
<td>$2^{14}$</td>
<td>0.84</td>
</tr>
<tr>
<td>$2^{15}$</td>
<td>0.81</td>
</tr>
</tbody>
</table>

Lower Bound $LB$, Daly High $DLYHIGH$, Daly Low $DLYLOW$, Young $YOUNG$

Petascale, MTBF = 125 years
Parallel jobs under Exponential failures (1/2)

Petascale, MTBF = 125 years
Parallel jobs under Exponential failures (1/2)

Average makespan degradation as a function of the number of processors. The graph shows different lines representing various lower bounds and algorithms:

- **LowerBound**
- **PeriodLB**
- **Young**
- **DalyLow**
- **DalyHigh**

The x-axis represents the number of processors, and the y-axis shows the average makespan degradation. The graph indicates that as the number of processors increases, the average makespan degradation also increases.

**Petascale, MTBF = 125 years**
Parallel jobs under Exponential failures (1/2)

\[
\begin{array}{c|c|c|c|c|c|c}
\text{number of processors} & 2^10 & 2^11 & 2^12 & 2^13 & 2^14 & 2^15 \\
\hline
\text{average makespan degradation} & 0.9 & 1 & 1.1 & 1.2 & 1.3 & 1.4 \\
\end{array}
\]

LowerBoundPeriodLBYoungDalyLowDalyHigh BouguerraLiu OptExp

Petascale, MTBF = 125 years
Parallel jobs under Exponential failures (1/2)

Lower Bound
Period LB
Young
Daly Low
Daly High
Liu Bouguerra Opt Exp
DPMakespan DP Next Failure

Petascale, MTBF = 125 years
Parallel jobs under Exponential failures (2/2)

Petascale
MTBF = 125 years

Petascale
MTBF = 500 years
Parallel jobs under Exponential failures (2/2)

- **Petascale**
  - MTBF = 125 years

- **Exascale**
  - MTBF = 1250 years
Parallel jobs under Weibull failures (1/2)

Petascale, MTBF = 125 years, k = 0.70
Parallel jobs under Weibull failures (1/2)

Petascale, MTBF = 125 years, k = 0.70
Parallel jobs under Weibull failures (1/2)

Petascale, MTBF = 125 years, k = 0.70
Parallel jobs under Weibull failures (1/2)

\[
\text{Petascale, MTBF} = 125 \text{ years, } k = 0.70
\]
Parallel jobs under Weibull failures (1/2)

Petascale, MTBF = 125 years, k = 0.70
Parallel jobs under Weibull failures (2/2)

**Petascale**

\[ MTBF = 125 \text{ years} \]
\[ k = 0.70 \]

**Petascale**

\[ MTBF = 500 \text{ years} \]
\[ k = 0.70 \]
Parallel jobs under Weibull failures (2/2)

Petascale
MTBF = 125 years
\( k = 0.70 \)

Petascale
MTBF = 125 years
45,208 processors

MTBF = 125 years
\( k = 0.70 \)
### Petascale
- **MTBF = 125 years**
- **$k = 0.70$**

### Exascale
- **MTBF = 1250 years**
- **$k = 0.70$**
Outline

1 Single-processor jobs
   - Solving **MAKESPAN**
   - Solving **NEXTFAILURE**

2 Parallel jobs
   - Solving **MAKESPAN**
   - Solving **NEXTFAILURE**

3 **Experiments**
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4 Conclusion
LANL trace set

Petascale / LANL Cluster 18

average makespan degradation

number of processors

Petascale / LANL Cluster 18
LANL trace set

Petascale / LANL Cluster 18
LANL trace set

Petascale / LANL Cluster 18
LANL trace set

Petascale / LANL Cluster 18
LANL trace set

![Graph for Petascale / LANL Cluster 18](image1)

![Graph for Petascale / LANL Cluster 19](image2)
Outline

1 Single-processor jobs
   - Solving \text{MAKESPAN}
   - Solving \text{NEXTFAILURE}

2 Parallel jobs
   - Solving \text{MAKESPAN}
   - Solving \text{NEXTFAILURE}

3 Experiments
   - Simulation framework
   - Sequential jobs under synthetic failures
   - Parallel jobs under synthetic failures
   - Parallel jobs under trace-based failures

4 Conclusion
Conclusion and perspectives

- Complete analytical solution for \textsc{Makespan}/Exponential
- Dynamic programming algorithms for \textsc{NextFailure}/Arbitrary distribution
- Makespan decreased by \textsc{DPNextFailure} (for the hardest cases)

- Future work
  Target non-coordinated checkpointing (e.g., hierarchical checkpointing with message logging)


J. W. Young. A first order approximation to the optimum checkpoint interval.