Identifying the Right Replication Level to Detect and Correct Silent Errors at Scale

Anne Benoit[1] Aurélien Cavelan[1] Franck Cappello[2]
Padma Raghavan[4] Yves Robert[1,3] Hongyang Sun[4]

[1]ENS Lyon & INRIA, France

[2]Argonne National Laboratory, USA

[3]University of Tennessee Knoxville, USA

[4]Vanderbilt University, USA

hongyang.sun@vanderbilt.edu

Fault Tolerance for HPC at eXtreme Scale (FTXS) Workshop
June 26, 2017@Washington, D.C., USA

An Inconvenient Truth

Top ranked supercomputers in the US (June 2017)

Rank	Name	Laboratory	Technology	Cores	PFlops/s	MTBF
4	Titan	ORNL	Cray XK7	560,640	17.59	$\approx$ 1 day
5	Sequoia	LLNL	BG/Q	1,572,864	17.17	$\approx$ 1 day
6	Cori	LBNL	Cray XC40	622,336	14.01	$\approx$ 1 day
9	Mira	ANL	BG/Q	786,432	8.59	$\approx$ 1 day

Fail-stop errors: Node failure, resource crashes

Silent errors or silent data corruptions (SDCs): Double bit flips, soft faults

An Inconvenient Truth

Top ranked supercomputers in the US (June 2017)

Rank	Name	Laboratory	Technology	Cores	PFlops/s	MTBF
4	Titan	ORNL	Cray XK7	560,640	17.59	$\approx$ 1 day
5	Sequoia	LLNL	BG/Q	1,572,864	17.17	$\approx$ 1 day
6	Cori	LBNL	Cray XC40	622,336	14.01	$\approx$ 1 day
9	Mira	ANL	BG/Q	786,432	8.59	$\approx$ 1 day

Fail-stop errors: Node failure, resource crashes

Silent errors or silent data corruptions (SDCs): Double bit flips, soft faults

Exascale computing (1000 PFlops/s):

- Larger core count: millions or even billions of cores
- Shorter Mean Time Between Failures (MTBF) μ

An Inconvenient Truth

Top ranked supercomputers in the US (June 2017)

Rank	Name	Laboratory	Technology	Cores	PFlops/s	MTBF
4	Titan	ORNL	Cray XK7	560,640	17.59	$\approx$ 1 day
5	Sequoia	LLNL	BG/Q	1,572,864	17.17	$\approx$ 1 day
6	Cori	LBNL	Cray XC40	622,336	14.01	$\approx$ 1 day
9	Mira	ANL	BG/Q	786,432	8.59	$\approx$ 1 day

Fail-stop errors: Node failure, resource crashes

Silent errors or silent data corruptions (SDCs): Double bit flips, soft faults

Exascale computing (1000 PFlops/s):

- Larger core count: millions or even billions of cores
- Shorter Mean Time Between Failures (MTBF) μ

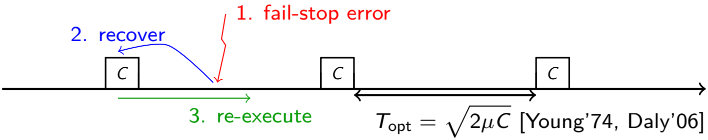
Coping with faults:

- Build more reliable hardware!
- Make applications more fault tolerant!
- Design better resilience techniques/algorithms!

Resilience Techniques for HPC

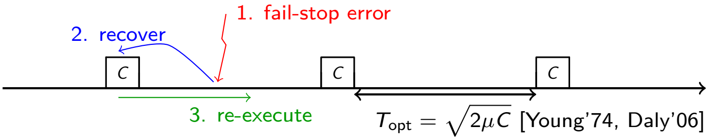
Fail-stop errors (instantaneous error detection)

Standard approach: periodic checkpointing, rollback and recovery

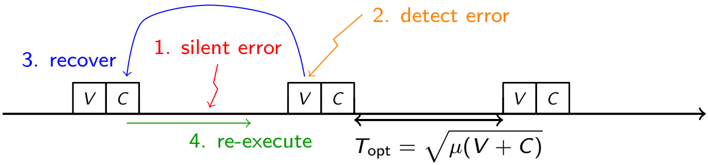


Resilience Techniques for HPC

Fail-stop errors (instantaneous error detection)
Standard approach: periodic checkpointing, rollback and recovery



Silent errors (arbitrary detection latency)
Promising approach: checkpointing + verification (error detection)



[1] A. Benoit, A. Cavelan, Y. Robert and H. Sun. Assessing General-Purpose Algorithms to Cope with Fail-Stop and Silent Errors. ACM Transactions on Parallel Computing, 2016.

Approaches for Detecting Silent Errors

Application-specific approaches

- Algorithm-based fault tolerance (ABFT): checksums in dense matrices, limited to one error detection and/or correction in practice [*Huang and Abraham 1984*]
- Partial differential equations (PDE): use lower-order scheme as verification mechanism [*Benson, Schmit and Schreiber 2014*]
- Generalized minimal residual method (GMRES): inner-outer iterations [*Hoemmen and Heroux 2011*]
- Preconditioned conjugate gradients (PCG): orthogonalization check iteratively, re-orthogonalization if error detected [*Sao and Vuduc 2013, Chen 2013*]

Data-analytics/machine learning approaches

- Dynamic monitoring of datasets based on physical laws (e.g., temperature/speed limit) and space or temporal proximity [*Bautista-Gomez and Cappello 2014*]
- Time-series prediction, spatial multivariate interpolation [*Di et al. 2014*]
- Offline training, online detection based on SDC signature for convergent iterative applications [*Liu and Agrawal 2016*]
- Spatial regression based on support vector machines [*Subasi et al. 2016*]

General-purpose approaches

- Process replication [*Fiala et al. 2012*]
- Group replication [*Casanova et al. 2014*]
- Triple modular redundancy (TMR) and voting [*Lyons and Vanderkulk 1962*]

This Talk

Focus:

Analytical model for applying replication/redundancy (general purpose approaches) in combination with checkpointing to detect and correct silent errors for HPC!

Question:

How to *optimally* execute a parallel job obeying Amdahl's law on an error-prone platform?

What is the optimal *error-aware speedup*?

When Amdahl Meets Young/Daly

Error-free speedup with P processors and α sequential fraction:

Amdahl's Law: $S(P) = \frac{1}{\alpha + \frac{1-\alpha}{P}}$

- Bounded above by $1/\alpha$
- Strictly increasing function of P

When Amdahl Meets Young/Daly

Error-free speedup with P processors and α sequential fraction:

Amdahl's Law: $S(P) = \frac{1}{\alpha + \frac{1-\alpha}{P}}$

- Bounded above by $1/\alpha$
- Strictly increasing function of P

Allocating more processors on an error-prone platform?

- Higher error-free speedup ☺
- More errors/faults ☹
  - More frequent checkpointing ☹
    - More resilience overhead ☹

When Amdahl Meets Young/Daly

Error-free speedup with P processors and α sequential fraction:

$$\textbf{Amdahl's Law}: S(P) = \frac{1}{\alpha + \frac{1-\alpha}{P}}$$

- Bounded above by $1/\alpha$
- Strictly increasing function of P

Allocating more processors on an error-prone platform?

- Higher error-free speedup ☺
- More errors/faults ☹
  - More frequent checkpointing ☹
    - More resilience overhead ☹

Optimal processor allocation and checkpointing interval?

[2] A. Cavelan, J. Li, Y. Robert and H. Sun, When Amdahl Meets Young/Daly. IEEE CLUSTER, 2016.

How Is Replication Used?

On a Q-processor platform, application is replicated n times:

- **Duplication**: each replica has $P = Q/2$ processors
- **Triplication**: each replica has $P = Q/3$ processors
- **General case**: each replica has $P = Q/n$ processors

How Is Replication Used?

On a Q-processor platform, application is replicated n times:

- **Duplication**: each replica has $P = Q/2$ processors
- **Triplication**: each replica has $P = Q/3$ processors
- **General case**: each replica has $P = Q/n$ processors

Having more replicas on an error-prone platform?

- Lower error-free speedup ☹
- More resilient ☺
  - Smaller checkpointing frequency ☺
    - Less resilience overhead ☺

How Is Replication Used?

On a Q-processor platform, application is replicated n times:

- **Duplication**: each replica has $P = Q/2$ processors
- **Triplication**: each replica has $P = Q/3$ processors
- **General case**: each replica has $P = Q/n$ processors

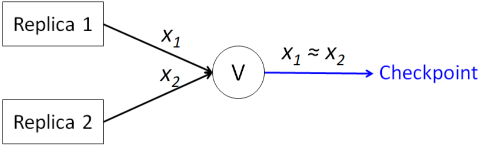
Having more replicas on an error-prone platform?

- Lower error-free speedup ☹
- More resilient ☺
  - Smaller checkpointing frequency ☺
    - Less resilience overhead ☺

Optimal replication level, processor allocation per replica and checkpointing interval?

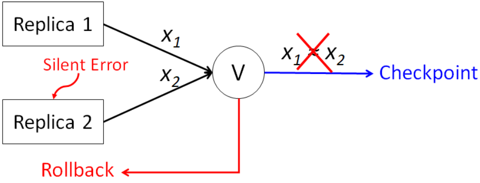
Why Is Replication Useful?

- **Error detection (duplication)**:



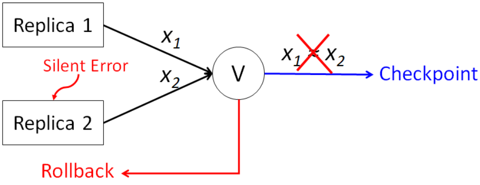
Why Is Replication Useful?

- **Error detection (duplication)**:

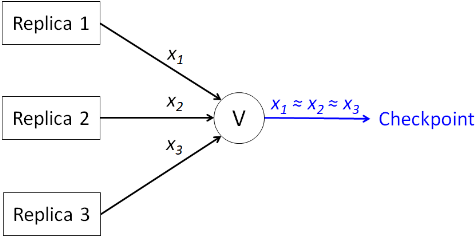


Why Is Replication Useful?

- **Error detection (duplication)**:

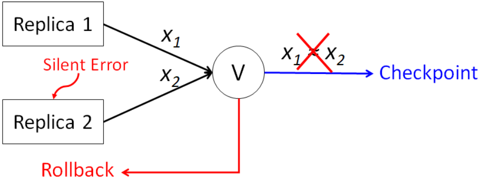


- **Error correction (triplication)**:

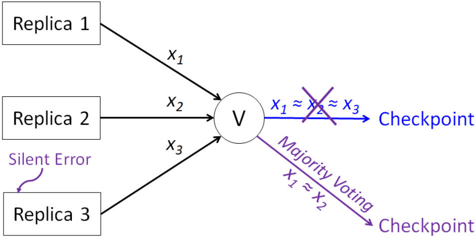


Why Is Replication Useful?

- **Error detection (duplication)**:

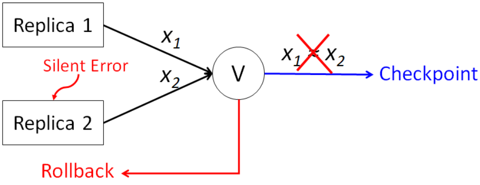


- **Error correction (triplication)**:

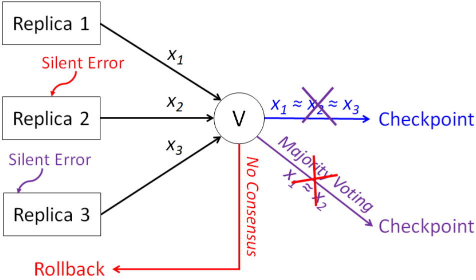


Why Is Replication Useful?

- **Error detection (duplication)**:

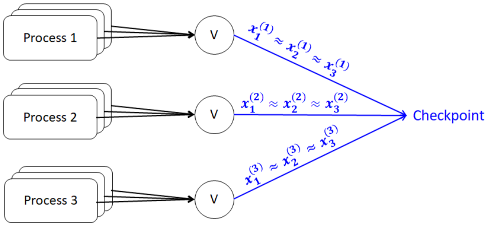


- **Error correction (triplication)**:

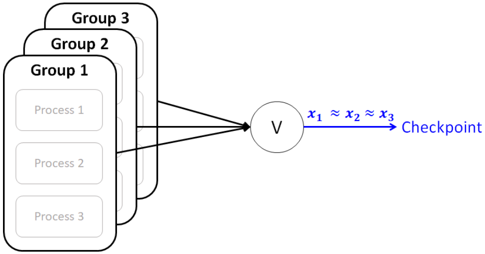


Two Replication Modes

- **Process Replication**:

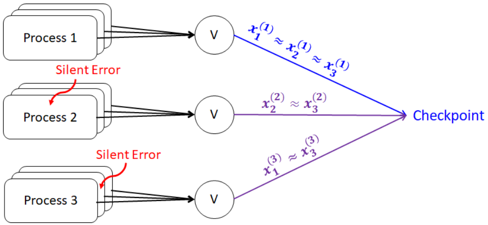


- **Group Replication**:

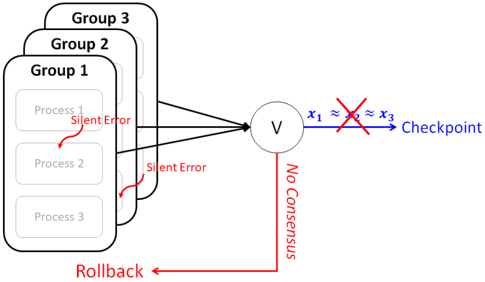


Two Replication Modes

- **Process Replication**:



- **Group Replication**:



Probability of Failure

Independent process error distribution

- Exponential $Exp(\lambda)$, $\lambda = 1/\mu$ (Memoryless)
- *Error probability of one process during T time of computation:*

$$\mathbb{P}(T) = 1 - e^{-\lambda T}$$

Probability of Failure

Independent process error distribution

- Exponential $Exp(\lambda)$, $\lambda = 1/\mu$ (Memoryless)
- *Error probability of one process during T time of computation:*

$$\mathbb{P}(T) = 1 - e^{-\lambda T}$$

Process Triplication:

- *Failure probability of any triplicated process:*

$$\begin{aligned}\mathbb{P}_3^{\text{prc}}(T,1) &= \binom{3}{2}\Big(1-\mathbb{P}(T)\Big)\mathbb{P}(T)^2 + \mathbb{P}(T)^3 \\ &= 3e^{-\lambda T}\left(1-e^{-\lambda T}\right)^2 + \left(1-e^{-\lambda T}\right)^3 \\ &= 1 - 3e^{-2\lambda T} + 2e^{-3\lambda T}\end{aligned}$$

Probability of Failure

Independent process error distribution

- Exponential $Exp(\lambda)$, $\lambda = 1/\mu$ (Memoryless)
- *Error probability of one process during T time of computation:*

$$\mathbb{P}(T) = 1 - e^{-\lambda T}$$

Process Triplication:

- *Failure probability of any triplicated process:*

$$\begin{aligned}\mathbb{P}_3^{\text{prc}}(T,1) &= \binom{3}{2}\Big(1-\mathbb{P}(T)\Big)\mathbb{P}(T)^2 + \mathbb{P}(T)^3 \\ &= 3e^{-\lambda T}\left(1-e^{-\lambda T}\right)^2 + \left(1-e^{-\lambda T}\right)^3 \\ &= 1 - 3e^{-2\lambda T} + 2e^{-3\lambda T}\end{aligned}$$

- *Failure probability of P-process application:*

$$\begin{aligned}\mathbb{P}_3^{\text{prc}}(T,P) &= 1 - \mathbb{P}(\text{"No process fails"}) \\ &= 1 - (1 - \mathbb{P}_3^{\text{prc}}(T,1))^P \\ &= 1 - \left(3e^{-2\lambda T} - 2e^{-3\lambda T}\right)^P\end{aligned}$$

Probability of Failure

Group Triplication:

- *Failure probability of any* <u>*P-process group*</u>*:*

$$\begin{aligned}\mathbb{P}_1^{\text{grp}}(T,P) &= 1 - \mathbb{P}(\text{“No process in group fails”}) \\ &= 1 - \big(1 - \mathbb{P}(T)\big)^P \\ &= 1 - e^{-\lambda PT}\end{aligned}$$

Probability of Failure

Group Triplication:

- *Failure probability of any P-process group:*

$$\begin{aligned}\mathbb{P}_1^{\text{grp}}(T,P) &= 1 - \mathbb{P}(\text{"No process in group fails"}) \\ &= 1 - \big(1 - \mathbb{P}(T)\big)^P \\ &= 1 - e^{-\lambda P T}\end{aligned}$$

- *Failure probability of three-group application:*

$$\begin{aligned}\mathbb{P}_3^{\text{grp}}(T,P) &= \binom{3}{2}\left(1 - \mathbb{P}_1^{\text{grp}}(T,1)\right)\mathbb{P}_1^{\text{grp}}(T,1)^2 + \mathbb{P}_1^{\text{grp}}(T,1)^3 \\ &= 3e^{-\lambda P T}\left(1 - e^{-\lambda P T}\right)^2 + \left(1 - e^{-\lambda P T}\right)^3 \\ &= 1 - 3e^{-2\lambda P T} + 2e^{-3\lambda P T}\end{aligned}$$

Probability of Failure

Group Triplication:

- *Failure probability of any <u>P-process group</u>:*

$$\begin{aligned}\mathbb{P}_1^{\text{grp}}(T,P) &= 1 - \mathbb{P}(\text{“No process in group fails”}) \\ &= 1 - \big(1 - \mathbb{P}(T)\big)^P \\ &= 1 - e^{-\lambda PT}\end{aligned}$$

- *Failure probability of <u>three-group application</u>:*

$$\begin{aligned}\mathbb{P}_3^{\text{grp}}(T,P) &= \binom{3}{2}\left(1 - \mathbb{P}_1^{\text{grp}}(T,1)\right)\mathbb{P}_1^{\text{grp}}(T,1)^2 + \mathbb{P}_1^{\text{grp}}(T,1)^3 \\ &= 3e^{-\lambda PT}\left(1 - e^{-\lambda PT}\right)^2 + \left(1 - e^{-\lambda PT}\right)^3 \\ &= 1 - 3e^{-2\lambda PT} + 2e^{-3\lambda PT} \\ &> 1 - \left(3e^{-2\lambda T} - 2e^{-3\lambda T}\right)^P = \mathbb{P}_3^{\text{prc}}(T,P)\end{aligned}$$

Probability of Failure

Group Triplication:

- *Failure probability of any P-process group:*

$$\mathbb{P}_1^{\text{grp}}(T,P) = 1 - \mathbb{P}(\text{“No process in group fails”}) = 1 - \left(1 - \mathbb{P}(T)\right)^P = 1 - e^{-\lambda PT}$$

- *Failure probability of three-group application:*

$$\begin{aligned}\mathbb{P}_3^{\text{grp}}(T,P) &= \binom{3}{2}\left(1 - \mathbb{P}_1^{\text{grp}}(T,1)\right)\mathbb{P}_1^{\text{grp}}(T,1)^2 + \mathbb{P}_1^{\text{grp}}(T,1)^3 \\ &= 3e^{-\lambda PT}\left(1 - e^{-\lambda PT}\right)^2 + \left(1 - e^{-\lambda PT}\right)^3 \\ &= 1 - 3e^{-2\lambda PT} + 2e^{-3\lambda PT} \\ &> 1 - \left(3e^{-2\lambda T} - 2e^{-3\lambda T}\right)^P = \mathbb{P}_3^{\text{prc}}(T,P)\end{aligned}$$

What about duplication? (any error kills both cases)

$$\mathbb{P}_2^{\text{prc}}(T,P) = \mathbb{P}_2^{\text{grp}}(T,P) = 1 - e^{-2\lambda TP}$$

Two Observations

Observation 1 (Implementation)

- **Process replication** is more resilient than group replication (assuming same overhead)
- **Group replication** is easier to implement by treating an application as a blackbox

Two Observations

Observation 1 (Implementation)

- **Process replication** is more resilient than group replication (assuming same overhead)
- **Group replication** is easier to implement by treating an application as a blackbox

Observation 2 (Analysis)

Following two scenarios are equivalent w.r.t. failure probability:

- **Group replication** with n replicas, where each replica has P processes and each process has error rate λ
- **Process replication** with one process, which has error rate λP and which is replicated n times

Benefit of analysis: Group$(n, P, \lambda) \rightarrow$ Process$(n, 1, \lambda P)$

Analysis Steps

Maximize error-aware speedup

$$\mathbb{S}_n(T, P) = \frac{S(P)}{\mathbb{E}_n(T, P)/T}$$

1. Derive failure probability $\mathbb{P}_n^{\text{prc}}(T, P)$ or $\mathbb{P}_n^{\text{grp}}(T, P)$ — exact
2. Compute expected execution time $\mathbb{E}_n(T, P)$ — exact
3. Compute first-order approx. of error-aware speedup $\mathbb{S}_n(T, P)$
4. Derive optimal T_{opt}, P_{opt} and get $\mathbb{S}_n(T_{\text{opt}}, P_{\text{opt}})$
5. Choose right replication level n

Analytical Results

Duplication:
On a platform with Q processors and checkpointing cost C, the optimal resilience parameters for *process/group duplication* are:

$$P_{\text{opt}} = \min\left\{\frac{Q}{2}, \left(\frac{1}{2}\left(\frac{1-\alpha}{\alpha}\right)^2 \frac{1}{C\lambda}\right)^{\frac{1}{3}}\right\}$$

$$T_{\text{opt}} = \left(\frac{C}{2\lambda P_{\text{opt}}}\right)^{\frac{1}{2}}$$

$$\mathbb{S}_{\text{opt}} = \frac{S(P_{\text{opt}})}{1 + 2\left(2\lambda C P_{\text{opt}}\right)^{\frac{1}{2}}}$$

Analytical Results

Duplication:
On a platform with Q processors and checkpointing cost C, the optimal resilience parameters for *process/group duplication* are:

$$P_{\text{opt}} = \min\left\{\frac{Q}{2}, \left(\frac{1}{2}\left(\frac{1-\alpha}{\alpha}\right)^2 \frac{1}{C\lambda}\right)^{\frac{1}{3}}\right\}$$

$$T_{\text{opt}} = \left(\frac{C}{2\lambda P_{\text{opt}}}\right)^{\frac{1}{2}}$$

$$\mathbb{S}_{\text{opt}} = \frac{S(P_{\text{opt}})}{1 + 2\left(2\lambda C P_{\text{opt}}\right)^{\frac{1}{2}}}$$

Triplication & (n, k)-replication (k-out-of-n replica consensus):
similar results but different for process and group, less practical for $n > 3$

Analytical Results

Duplication:
On a platform with Q processors and checkpointing cost C, the optimal resilience parameters for *process/group duplication* are:

$$P_{\text{opt}} = \min\left\{\frac{Q}{2}, \left(\frac{1}{2}\left(\frac{1-\alpha}{\alpha}\right)^2 \frac{1}{C\lambda}\right)^{\frac{1}{3}}\right\}$$

$$T_{\text{opt}} = \left(\frac{C}{2\lambda P_{\text{opt}}}\right)^{\frac{1}{2}}$$

$$\mathbb{S}_{\text{opt}} = \frac{S(P_{\text{opt}})}{1 + 2\left(2\lambda C P_{\text{opt}}\right)^{\frac{1}{2}}}$$

Triplication & (n, k)-replication (k-out-of-n replica consensus):
similar results but different for process and group, less practical for $n > 3$

- For $\alpha > 0$, not necessarily use up all available Q processors
- Checkpointing interval T_{opt} nicely extends Young/Daly's result
- Error-aware speedup $\mathbb{S}_{\text{opt}}$ minimally affected for small λ

Results Comparison

For fully parallel jobs, i.e., $\alpha = 0$ (similar for $\alpha > 0$)

- Duplication v.s. Process triplication

Duplication	Process triplication	
$P_{\text{opt}} = \frac{Q}{2}$	$P_{\text{opt}} = \frac{Q}{3}$	(Processors ↓)
$T_{\text{opt}} = \sqrt{\frac{C}{\lambda Q}}$	$T_{\text{opt}} = \sqrt[3]{\frac{C}{2\lambda^2 Q}}$	(Chkpt interval ↑)
$\mathbb{S}_{\text{opt}} = \frac{Q/2}{1 + 2\sqrt{\lambda C Q}}$	$\mathbb{S}_{\text{opt}} = \frac{Q/3}{1 + 3\sqrt[3]{\left(\frac{\lambda C}{2}\right)^2 Q}}$	(Exp. speedup??)

Results Comparison

For fully parallel jobs, i.e., $\alpha = 0$ (similar for $\alpha > 0$)

- Duplication v.s. Process triplication

$$P_{\text{opt}} = \frac{Q}{2} \qquad P_{\text{opt}} = \frac{Q}{3} \qquad \text{(Processors } \downarrow\text{)}$$

$$T_{\text{opt}} = \sqrt{\frac{C}{\lambda Q}} \qquad T_{\text{opt}} = \sqrt[3]{\frac{C}{2\lambda^2 Q}} \qquad \text{(Chkpt interval } \uparrow\text{)}$$

$$\mathbb{S}_{\text{opt}} = \frac{Q/2}{1 + 2\sqrt{\lambda C Q}} \qquad \mathbb{S}_{\text{opt}} = \frac{Q/3}{1 + 3\sqrt[3]{\left(\frac{\lambda C}{2}\right)^2 Q}} \qquad \text{(Exp. speedup??)}$$

- Process triplication v.s. Group triplication

$$P_{\text{opt}} = \frac{Q}{3} \qquad P_{\text{opt}} = \frac{Q}{3} \qquad \text{(Processors } =\text{)}$$

$$T_{\text{opt}} = \sqrt[3]{\frac{C}{2\lambda^2 Q}} \qquad T_{\text{opt}} = \sqrt[3]{\frac{3C}{2(\lambda Q)^2}} \qquad \text{(Chkpt interval } \downarrow\text{)}$$

$$\mathbb{S}_{\text{opt}} = \frac{Q/3}{1 + 3\sqrt[3]{\left(\frac{\lambda C}{2}\right)^2 Q}} \qquad \mathbb{S}_{\text{opt}} = \frac{Q/3}{1 + 3\sqrt[3]{\frac{1}{3}\left(\frac{\lambda C Q}{2}\right)^2}} \qquad \text{(Exp. speedup } \downarrow\text{)}$$

Results Comparison

For fully parallel jobs, i.e., $\alpha = 0$ (similar for $\alpha > 0$)

- Duplication v.s. Process triplication

$P_{\text{opt}} = \frac{Q}{2}$ $P_{\text{opt}} = \frac{Q}{3}$ (Processors ↓)

val ↑)

up??)

- P

=)

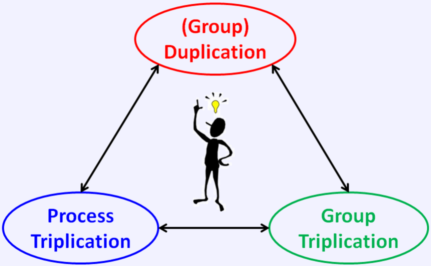
rval ↓)

up ↓)

$1 + 3\sqrt[3]{\left(\frac{\lambda C}{2}\right)^2} Q$ $1 + 3\sqrt[3]{\frac{1}{3}\left(\frac{\lambda CQ}{2}\right)^2}$

Choosing Right Mode & Level of Replication

Based on analytical model and whether process replication is supported



Limitation of First-Order Approximation

Observation 3 (First-Order)

Suppose $P = \Theta(\lambda^{-x})$ and $T = \Theta(\lambda^{-y})$. Then, for first-order approximation to accurately estimate error probabilities (e.g., $1 - e^{-\lambda PT} \approx \lambda PT$), we need:

$$x + y < 1$$
$$\text{or} \quad P \cdot T = o(\mu)$$

e.g., $\mu = 10$ years $\Rightarrow P \cdot T < 3 \cdot 10^8$ processor-seconds
Generally accurate for platform MTBF $\mu_P = \Theta(\text{days})$ or $\mu_P = \Theta(\text{hours})$ depending on checkpointing cost C

Limitation of First-Order Approximation

Observation 3 (First-Order)

Suppose $P = \Theta(\lambda^{-x})$ and $T = \Theta(\lambda^{-y})$. Then, for first-order approximation to accurately estimate error probabilities (e.g., $1 - e^{-\lambda PT} \approx \lambda PT$), we need:

$$x + y < 1$$

$$\text{or} \quad P \cdot T = o(\mu)$$

e.g., $\mu = 10$ years $\Rightarrow P \cdot T < 3 \cdot 10^8$ processor-seconds

Generally accurate for platform MTBF $\mu_P = \Theta(\text{days})$ or $\mu_P = \Theta(\text{hours})$ depending on checkpointing cost C

What about larger systems?

One solution: multi-level checkpointing $\Rightarrow$ error separation

[3] A. Benoit, A. Cavelan, V. Le Fèvre, Y. Robert and H. Sun. Towards Optimal Multi-Level Checkpointing. IEEE Transactions on Computers, 2017.

Simulations

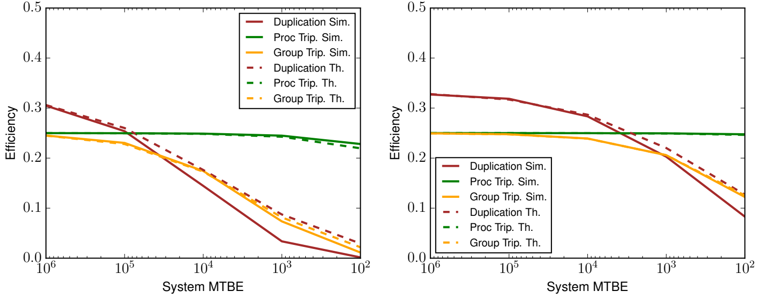
Consider an platform with $Q = 10^6$, and study

$$\textit{Efficiency} = \frac{\mathbb{S}_{\mathsf{opt}}}{Q}$$

- Impact of MTBE and checkpointing cost C
- Impact of sequential fraction α
- Impact of number of processes P

Impact of MTBE and Checkpointing Cost

$$\alpha = 10^{-6}$$

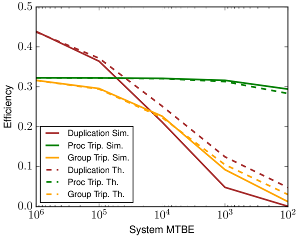


(a) $C = 1800s$ (b) $C = 60s$

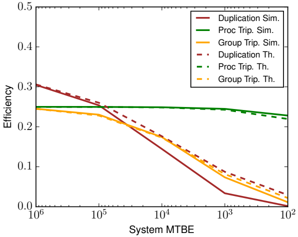
- First-order accurate except for duplication (where P is larger) and with small MTBE
- Duplication can be sufficient for large MTBE, especially for small checkpointing cost

Impact of Sequential Fraction

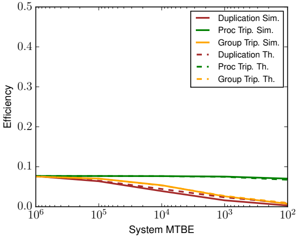
$C = 1800s$



(c) $\alpha = 10^{-7}$



(d) $\alpha = 10^{-6}$

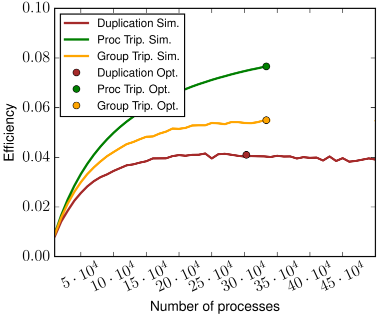


(e) $\alpha = 10^{-5}$

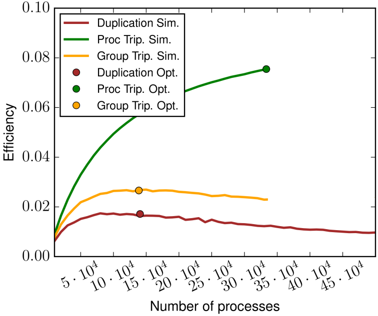
- Increased α reduces efficiency
- Increased α increases minimum MTBE for which duplication is sufficient

Impact of Number of Processes

$$\alpha = 10^{-5}, C = 1800s$$



(f) $MTBE = 10^4$



(g) $MTBE = 10^3$

- Efficiency/error-aware speedup no longer strictly increasing function of P
- First-order P_{opt} close to actual optimum

Conclusion

What to Remember

- "Replication + checkpointing" as a general-purpose fault-tolerance protocol for coping with silent errors in HPC
- Process replication is more resilient than group replication, but group replication is easier to implement
- Analytical solution for P_{opt}, T_{opt}, *and* $\mathbb{S}_{\text{opt}}$ and for choosing right replication mode and level

Future Work

- Analyzing partial replication paradigm: different replication modes and levels for tasks with different criticality
- Dealing with co-existence of fail-stop errors and silent errors
- Experimenting with real applications/platforms