Assessing General-Purpose Algorithms to Cope with Fail-stop and Silent Errors

Anne Benoit[1], Aurélien Cavelan[1], Yves Robert[1,2] and Hongyang Sun[1]

1. ENS Lyon
2. University of Tennessee Knoxville

aurelien.cavelan@inria.fr

PMBS14

Definitions

- Instantaneous error detection $\Rightarrow$ fail-stop failures, e.g. resource crash
- Silent errors (data corruption) $\Rightarrow$ detection latency

Silent error detected only when corrupt data is activated and modifies application behavior

- Includes some software faults, some hardware errors (soft errors in L1 cache, ALU), double bit flip
- Cannot always be corrected by ECC memory

Probability distributions for silent errors

?

Theorem: $\mu_p = \frac{\mu_{\text{ind}}}{p}$ for arbitrary distributions

(a.k.a, scale is the enemy)

Probability distributions for silent errors

?

Theorem: $\mu_p = \dfrac{\mu_{\text{ind}}}{p}$ for arbitrary distributions

(a.k.a, scale is the enemy)

Lesson learnt for fail-stop failures

(Not so) Secret data

- Tsubame 2: 962 failures during last 18 months so $\mu = 13$ hrs
- Blue Waters: 2-3 node failures per day
- Titan: a few failures per day
- Tianhe 2: wouldn't say

Lesson learnt for fail-stop failures

(Not so) Secret data

- Tsubame 2: 962 failures during last 18 months so $\mu = 13$ hrs
- Blue Waters: 2-3 node failures per day
- Titan: a few failures per day
- Tianhe 2: wouldn't say

$$T_{\text{opt}} = \sqrt{2\mu C} \quad \Rightarrow \quad \textsc{Waste}_{\text{opt}} \approx \sqrt{\frac{2C}{\mu}}$$

Lesson learnt for fail-stop failures

(Not so) Secret data

- Tsubame 2: 962 failures during last 18 months so $\mu = 13$ hrs
- Blue Waters: 2-3 node failures per day
- Titan: a few failures per day
- Tianhe 2: wouldn't say

$$T_{\text{opt}} = \sqrt{2\mu C} \quad \Rightarrow \quad \text{WASTE}_{\text{opt}} \approx \sqrt{\frac{2C}{\mu}}$$

Petascale:	$C = 20$ min	$\mu = 24$ hrs	$\Rightarrow \text{WASTE}_{\text{opt}} = 17\%$
Scale by 10:	$C = 20$ min	$\mu = 2.4$ hrs	$\Rightarrow \text{WASTE}_{\text{opt}} = 53\%$
Scale by 100:	$C = 20$ min	$\mu = 0.24$ hrs	$\Rightarrow \text{WASTE}_{\text{opt}} = 100\%$

Lesson learnt for fail-stop failures

(Not so) Secret data

- Ts[illegible]e 2: 962 failures during last 18 months so $\mu =$ [illegible] hrs
- Blue Wate[illegible]-3 node failures per day
- Titan: a few failure[illegible] day
- Tianhe 2: wouldn't say

Exascale $\neq$ Petascale $\times 1000$
Need more reliable components
Need to checkpoint faster

Petascale:	$C = 20$ min	$\mu = 24$ hrs	$\Rightarrow$	$\mathrm{WASTE}_{\mathrm{opt}} = 17\%$
Scale by 1[illegible]:	$C = 20$ min	$\mu = 2.4$ hrs	$\Rightarrow$	$\mathrm{WASTE}_{\mathrm{opt}} = 53\%$
Scale [illegible]100:	$C = 20$ min	$\mu = 0.24$ hrs	$\Rightarrow$	$\mathrm{WASTE}_{\mathrm{opt}} = 100\%$

Lesson learnt for fail-stop failures

(Not so) Secret data

- Tsubame 2: 962 failures during last 18 months so $\mu = 13$ hrs
- Blue Waters: 2-3 node failures per day
- Titan: a few failures per day
- Tianhe 2: wouldn't say

Silent errors:
detection latency $\Rightarrow$ additional problems

P ... 7%

Scale by 10: $C = 20$ min $\mu = 2.4$ hrs $\Rightarrow \text{WASTE}_{\text{opt}} = 53\%$

Scale by 100: $C = 20$ min $\mu = 0.24$ hrs $\Rightarrow \text{WASTE}_{\text{opt}} = 100\%$

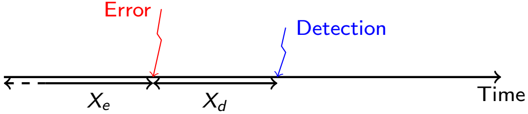
Outline

1. General-purpose approach
2. Checkpointing and Verification
   - Divisible load
   - Linear chains of tasks
3. Simulations
   - SINGLESPEED scenario for makespan
   - SINGLESPEED scenario for energy
   - REEXECSPEED and MULTISPEED scenarios

Outline

1. General-purpose approach
2. Checkpointing and Verification
   - Divisible load
   - Linear chains of tasks
3. Simulations
   - SINGLESPEED scenario for makespan
   - SINGLESPEED scenario for energy
   - REEXECSPEED and MULTISPEED scenarios

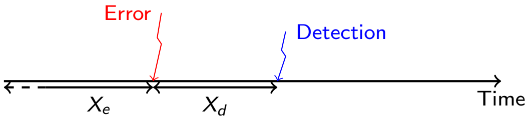
General-purpose approach



Error and detection latency

- Last checkpoint may have saved an already corrupted state
- Saving k checkpoints (Lu, Zheng and Chien):
  1. Critical failure when all live checkpoints are invalid
  2. Which checkpoint to roll back to?

General-purpose approach



Error and detection latency

- Last checkpoint may have saved an already corrupted state
- Saving k checkpoints (Lu, Zheng and Chien):
  1. Critical failure when all live checkpoints are invalid
     Assume unlimited storage resources
  2. Which checkpoint to roll back to?
     Need a verification mechanism

Outline

1. General-purpose approach

2. Checkpointing and Verification
   - Divisible load
   - Linear chains of tasks

3. Simulations
   - SingleSpeed scenario for makespan
   - SingleSpeed scenario for energy
   - ReExecSpeed and MultiSpeed scenarios

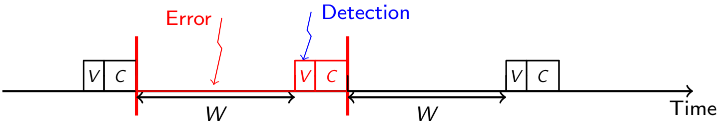
Coupling checkpointing and verification

- Verification mechanism of cost V
- Silent errors detected only when verification is executed
- Approach agnostic of the nature of verification mechanism (checksum, error correcting code, coherence tests, triple modular redundancy, etc)
- Fully general-purpose
  (application-specific information, if available, can always be used to decrease V)

Outline

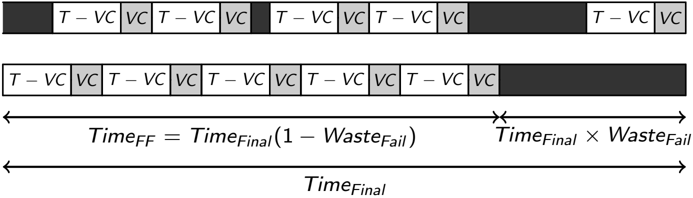
1. General-purpose approach
2. Checkpointing and Verification
   - Divisible load
   - Linear chains of tasks
3. Simulations
   - SINGLESPEED scenario for makespan
   - SINGLESPEED scenario for energy
   - REEXECSPEED and MULTISPEED scenarios

Base pattern (and revisiting Young/Daly)



	Fail-stop (classical)	Silent errors
Pattern	$T = W + C$	$S = W + V + C$
WASTE_{FF}	$\frac{C}{T}$	$\frac{V+C}{S}$
$\text{WASTE}_{\text{fail}}$	$\frac{1}{\mu}(D + R + \frac{W}{2})$	$\frac{1}{\mu}(R + W + V)$
Optimal	$T_{\text{opt}} = \sqrt{2C\mu}$	$S_{\text{opt}} = \sqrt{(C + V)\mu}$
$\text{WASTE}_{\text{opt}}$	$\sqrt{\frac{2C}{\mu}}$	$2\sqrt{\frac{C+V}{\mu}}$

Young/Daly



$$\text{Waste} = \text{Waste}_{ef} + \text{Waste}_{fail}$$

$$\text{Waste} = \frac{V + C}{T} + \lambda^F(s)(R + \frac{T}{2}) + \lambda^S(s)(R + T)$$

$$T_{\text{opt}} = \sqrt{\frac{2(V + C)}{\lambda^F(s) + 2\lambda^S(s)}}$$

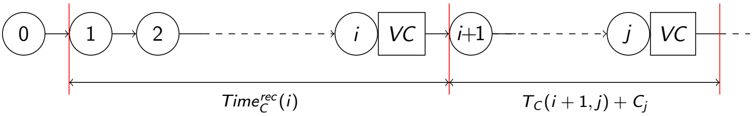
Outline

1. General-purpose approach
2. Checkpointing and Verification
   - Divisible load
   - Linear chains of tasks
3. Simulations
   - SingleSpeed scenario for makespan
   - SingleSpeed scenario for energy
   - ReExecSpeed and MultiSpeed scenarios

Linear chain

- $\{T_1, T_2, \ldots, T_n\}$: linear chain of n tasks
- Each task T_i fully parametrized:
  - w_i computational weight
  - C_i, R_i, V_i : checkpoint, recovery, verification
- Error rates:
  - λ^F rate of fail-stop errors
  - λ^S rate of silent errors

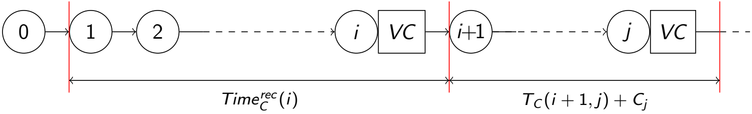
VC-ONLY



$$Time_C^{rec}(n)$$

$$Time_C^{rec}(j) = \min_{0 \leq i < j} \{ Time_C^{rec}(i) + T_C^{SF}(i+1, j) \}$$

VC-ONLY

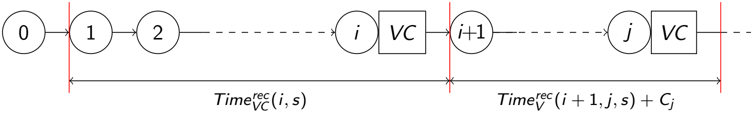


$$Time_C^{rec}(n)$$

$$Time_C^{rec}(j) = \min_{0 \leq i < j} \{ Time_C^{rec}(i) + T_C^{SF}(i+1, j) \}$$

$$T_C^{SF}(i,j) = p_{i,j}^F \left(T_{lost_{i,j}} + R_{i-1} + T_C^{SF}(i,j) \right)$$
$$+ \left(1 - p_{i,j}^F \right) \left(\sum_{\ell=i}^{j} w_\ell + V_j + p_{i,j}^S \left(R_{i-1} + T_C^{SF}(i,j) \right) + \left(1 - p_{i,j}^S \right) C_j \right)$$

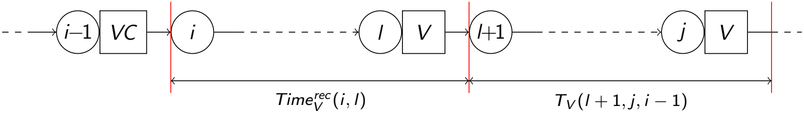
VC+V



$$Time^{rec}_{VC}(n)$$

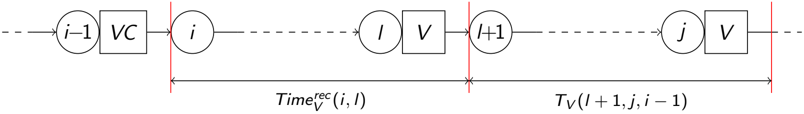
$$Time^{rec}_{VC}(j) = \min_{0 \le i < j} \{ Time^{rec}_{VC}(i) + Time^{rec}_{V}(i+1,j) + C_j \}$$

VC+V: $Time_V^{rec}$



$$Time_V^{rec}(i,j) = \min_{i-1 \leq l < j} \{ Time_V^{rec}(i,l) + T_V(l+1, j, i-1) \}$$

VC+V: $Time_V^{rec}$



$$Time_V^{rec}(i,j) = \min_{i-1 \leq l < j} \{ Time_V^{rec}(i,l) + T_V(l+1,j,i-1) \}$$

$$T_V^{SF}(i,j,l_c) = p_{i,j}^F \left(T_{lost_{i,j}} + R_{l_c} + Time_V^{rec}(l_c+1,i-1) + T_V^{SF}(i,j,l_c) \right)$$
$$+ (1-p_{i,j}^F) \left(\sum_{\ell=i}^{j} w_\ell + V_j + p_{i,j}^S \left(R_{l_c} + Time_V^{rec}(l_c+1,i-1) + T_V^{SF}(i,j,l_c) \right) \right)$$

Extensions

- VC-ONLY and VC+V
- Different speeds with DVFS, different error rates
- Different execution modes
- Optimize for time or for energy consumption

☺ ☺ ☺

- Use verification to correct some errors (ABFT)
- Same analysis (smaller error rate but higher verification cost)

Extensions

- VC-ONLY and VC+V
- Different speeds with DVFS, different error rates
- Different execution modes
- Optimize for time or for energy consumption

☺ ☺ ☺

- Use verification to correct some errors (ABFT)
- Same analysis (smaller error rate but higher verification cost)

Outline

1. General-purpose approach
2. Checkpointing and Verification
   - Divisible load
   - Linear chains of tasks
3. Simulations
   - SingleSpeed scenario for makespan
   - SingleSpeed scenario for energy
   - ReExecSpeed and MultiSpeed scenarios

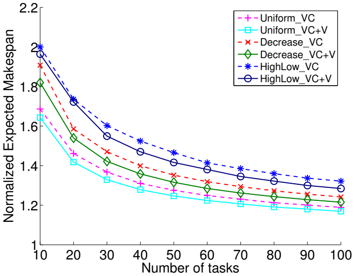
Settings

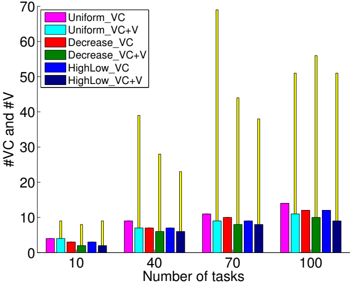
- Linear chains with n tasks
  - Total work $W \approx 14$ hours
  - Patterns: (1) Uniform; (2) Decrease; (3) HighLow
- Set of speeds from Intel Xscale processor
  - Normalized speeds $\{0.15, 0.4, 0.6, 0.8, 1\}$
  - Fitted power function $P(s) = 1550s^3 + 60$
  - $\lambda^F(s) = \lambda^F_{\mathrm{ref}} \cdot 10^{\frac{d \cdot |s_{\mathrm{ref}} - s|}{s_{\max} - s_{\min}}}$
  - Reference speed $s_{\mathrm{ref}} = 0.6$ and $\lambda^F_{\mathrm{ref}} = 10^{-5}$ for fail-stop errors
  - Sensitivity parameter $d = 3$
  - Corresponds to $0.83 \sim 129$ errors over entire chain
  - Silent errors: $\lambda^S(s) = \eta \cdot \lambda^F(s)$
- Checkpoint and verification costs for a task
  - cr ratio of checkpointing cost over computational cost
  - vr ratio of verification cost over computational cost
  - Default: checkpoint cost $\gg$ verification cost

Outline

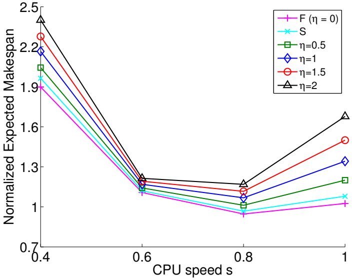
1. General-purpose approach
2. Checkpointing and Verification
   - Divisible load
   - Linear chains of tasks
3. Simulations
   - SingleSpeed scenario for makespan
   - SingleSpeed scenario for energy
   - ReExecSpeed and MultiSpeed scenarios

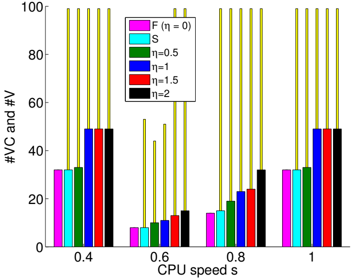
Impact of n and cost distribution



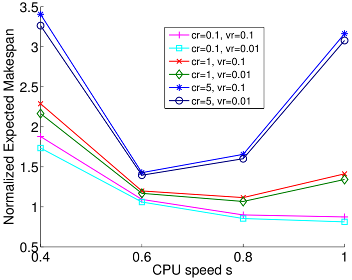


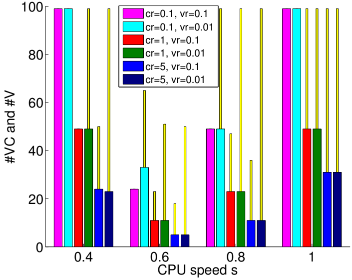
Impact of η ($\textsc{Time}$-VC+V, $n = 100$, Uniform)





Impact of *cr* and *vr* (Time-VC+V, $n = 100$, Uniform)

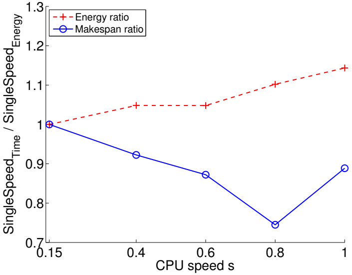


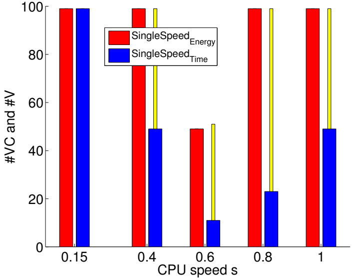


Outline

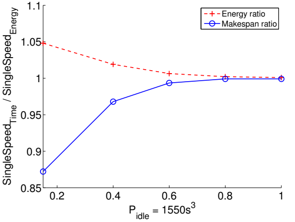
1. General-purpose approach
2. Checkpointing and Verification
   - Divisible load
   - Linear chains of tasks
3. Simulations
   - SingleSpeed scenario for makespan
   - SingleSpeed scenario for energy
   - ReExecSpeed and MultiSpeed scenarios

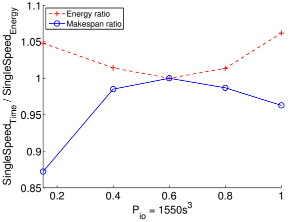
Impact of CPU speed s

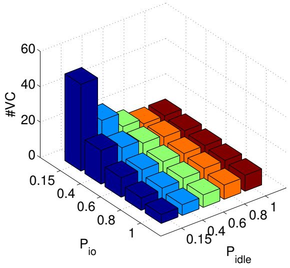




Impact of P_{idle} and P_{io}



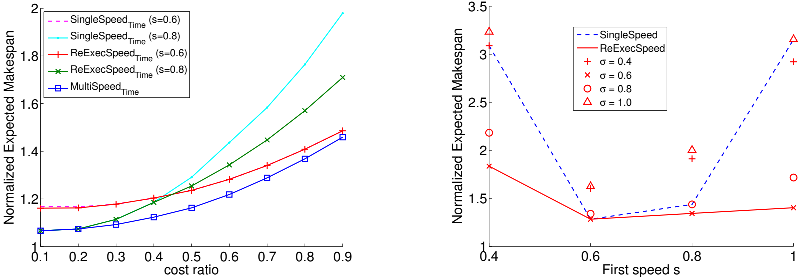




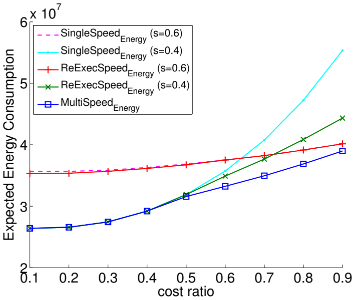
Outline

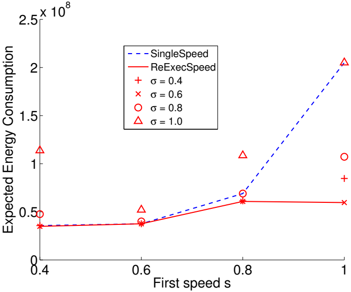
1. General-purpose approach
2. Checkpointing and Verification
   - Divisible load
   - Linear chains of tasks
3. Simulations
   - SingleSpeed scenario for makespan
   - SingleSpeed scenario for energy
   - ReExecSpeed and MultiSpeed scenarios

Time-VC+V (HighLow)



Energy-VC+V (HighLow)





Conclusion

- Soft errors difficult to cope with, even for divisible workloads or linear chains
- Investigate general task graphs
- Combine checkpointing, replication and application-specific techniques
- Multi-criteria optimization problem
  execution time/energy/reliability
  best resource usage (performance trade-offs)

Several challenging algorithmic/scheduling problems ☺