Energy-aware checkpointing of divisible tasks with soft or hard deadlines

Guillaume Aupy\textsuperscript{1}, Anne Benoit\textsuperscript{1,2}, Rami Melhem\textsuperscript{3}, Paul Renaud-Goud\textsuperscript{1} and Yves Robert\textsuperscript{1,2,4}

1. Ecole Normale Supérieure de Lyon, France
2. Institut Universitaire de France
3. University of Pittsburgh, USA
4. University of Tennessee Knoxville, USA

Anne.Benoit@ens-lyon.fr
http://graal.ens-lyon.fr/~abenoit/

International Green Computing Conference 2013
Arlington, USA
Divisible load scheduling and resilience

- **Divisible load scheduling**: divide a computational workload into chunks
  - Arbitrary number of chunks
  - Size of chunks freely chosen by user
- **Goal**: minimize *makespan*, i.e., total execution time

- Current platforms: increasing frequency of failures
- Well-established method to deal with failures: **checkpointing**
  - Take a checkpoint at the end of each chunk and verify result
  - Re-execution in case of transient failure
Divisible load scheduling and resilience

- **Divisible load scheduling**: divide a computational workload into chunks
  - Arbitrary number of chunks
  - Size of chunks freely chosen by user
- **Goal**: minimize makespan, i.e., total execution time

- **Current platforms**: increasing frequency of failures
- **Well-established method to deal with failures**: checkpointing
  - Take a checkpoint at the end of each chunk and verify result
  - **Re-execution** in case of transient failure
Energy: a crucial issue

- IGCC: Green Computing Conference!
- Real need to reduce energy dissipation in current processors
- Processor running at speed $s$: power $s^3$ watts
- Dynamic voltage and frequency scaling techniques (DVFS)

- Our goal: minimize energy consumption
  - including that of checkpointing and re-execution (if failure)
  - while enforcing a bound on execution time
Energy: a crucial issue

• IGCC: Green Computing Conference!
• Real need to reduce energy dissipation in current processors
• Processor running at speed $s$: power $s^3$ watts
• Dynamic voltage and frequency scaling techniques (DVFS)

• Our goal: minimize energy consumption
  • including that of checkpointing and re-execution (if failure)
  • while enforcing a bound on execution time
Outline

1. Framework
2. With a single chunk
3. With several chunks
4. Simulation results
5. Conclusion
Framework

- Execution of a divisible task ($W$ operations)
- Failures may occur
  - Transient faults
  - Resilience through checkpointing
- Objective: minimize expected energy consumption $\mathbb{E}(E)$, given a deadline bound $D$

- Probabilistic nature of failure hits: expectation of energy consumption is natural (average cost over many executions)
- Deadline bound: two relevant scenarios (soft or hard deadline)
Soft vs hard deadline

- Soft deadline: met in expectation, i.e., $\mathbb{E}(T) \leq D$ (average response time)
- Hard deadline: met in the worst case, i.e., $T_{wc} \leq D$
Execution time, one single chunk

One single chunk of size $W$

- Checkpoint overhead: execution time $T_C$
- Instantaneous failure rate: $\lambda$

- **First execution** at speed $s$: $T_{\text{exec}} = \frac{W}{s} + T_C$
- Failure probability: $P_{\text{fail}} = \lambda T_{\text{exec}} = \lambda\left(\frac{W}{s} + T_C\right)$
- In case of failure: **re-execute** at speed $\sigma$: $T_{\text{reexec}} = \frac{W}{\sigma} + T_C$
- And we assume success after re-execution

- $E(T) = T_{\text{exec}} + P_{\text{fail}} T_{\text{reexec}} = \left(\frac{W}{s} + T_C\right) + \lambda\left(\frac{W}{s} + T_C\right)\left(\frac{W}{\sigma} + T_C\right)$
- $T_{wc} = T_{\text{exec}} + T_{\text{reexec}} = \left(\frac{W}{s} + T_C\right) + \left(\frac{W}{\sigma} + T_C\right)$
Energy consumption, one single chunk

One single chunk of size $W$

- Checkpoint overhead: energy consumption $E_C$

- First execution at speed $s$: $\frac{W}{s} \times s^3 + E_C = Ws^2 + E_C$

- Re-execution at speed $\sigma$: $W\sigma^2 + E_C$, with probability $P_{\text{fail}}$

\[ P_{\text{fail}} = \lambda T_{\text{exec}} = \lambda \left( \frac{W}{s} + T_C \right) \]

- $\mathbb{E}(E) = (Ws^2 + E_C) + \lambda \left( \frac{W}{s} + T_C \right) (W\sigma^2 + E_C)$
Multiple chunks

- Execution times: **sum** of execution times for each chunk (worst-case or expected)
- Expected energy consumption: **sum** of expected energy for each chunk

Coherent failure model: consider two chunks $W_1 + W_2 = W$

- Probability of failure for first chunk: $P_{\text{fail}}^1 = \lambda \left( \frac{W_1}{s} + T_C \right)$
- For second chunk: $P_{\text{fail}}^2 = \lambda \left( \frac{W_2}{s} + T_C \right)$
- With a single chunk of size $W$: $P_{\text{fail}} = \lambda \left( \frac{W}{s} + T_C \right)$, differs from $P_{\text{fail}}^1 + P_{\text{fail}}^2$ only because of extra checkpoint

Trade-off: many small chunks (more $T_C$ to pay, but small re-execution cost) vs few larger chunks (fewer $T_C$, but increased re-execution cost)
Optimization problem

- Decisions that should be taken before execution:
  - Chunks: how many ($n$)? which sizes ($W_i$ for chunk $i$)?
  - Speeds of each chunk: first run ($s_i$)? re-execution ($\sigma_i$)?

- Input: $W$, $T_C$ (checkpointing time), $E_C$ (energy spent for checkpointing), $\lambda$ (instantaneous failure rate), $D$ (deadline)
Optimization problem

- Decisions that should be taken before execution:
  - Chunks: how many \( n \)? which sizes \( W_i \) for chunk \( i \)?
  - Speeds of each chunk: first run \( s_i \)? re-execution \( \sigma_i \)?

- Input: \( W \), \( T_C \) (checkpointing time), \( E_C \) (energy spent for checkpointing), \( \lambda \) (instantaneous failure rate), \( D \) (deadline)
Optimization problem

- Decisions that should be taken before execution:
  - Chunks: how many ($n$)? which sizes ($W_i$ for chunk $i$)?
  - Speeds of each chunk: first run ($s_i$)? re-execution ($\sigma_i$)?

- Input: $W$, $T_C$ (checkpointing time), $E_C$ (energy spent for checkpointing), $\lambda$ (instantaneous failure rate), $D$ (deadline)
Models

- **Chunks**
  - Single chunk of size $W$
  - VS
  - Multiple chunks ($n$ and $W_i$'s)

- **Speed per chunk**
  - Single speed ($s$)
  - VS
  - Multiple speeds ($s$ and $\sigma$)

- **Deadline bound**
  - Hard ($T_{wc} \leq D$)
  - VS
  - Soft ($\mathbb{E}(T) \leq D$)
Outline

1. Framework
2. With a single chunk
3. With several chunks
4. Simulation results
5. Conclusion
Consider first that \( s = \sigma \) (single speed): need to find optimal speed

- \( \mathbb{E}(E) \) is a function of \( s \):
  \[
  \mathbb{E}(E)(s) = (Ws^2 + E_C)(1 + \lambda(\frac{W}{s} + T_C))
  \]

- Lemma: this function is convex and has a unique minimum \( s^* \) (function of \( \lambda, W, E_c, T_c \))
  \[
  s^* = \frac{\lambda W}{6(1+\lambda T_c)} \left( \frac{-(3\sqrt{3}\sqrt{27a^2-4a-27a+2})^{1/3}}{2^{1/3}} - \frac{2^{1/3}}{(3\sqrt{3}\sqrt{27a^2-4a-27a+2})^{1/3}} - 1 \right),
  \]
  where \( a = \lambda E_C \left( \frac{2(1+\lambda T_C)}{\lambda W} \right)^2 \)

- \( \mathbb{E}(T) \) and \( T_{wc} \): decreasing functions of \( s \)

- Minimum speed \( s_{exp} \) and \( s_{wc} \) required to match deadline \( D \) (function of \( D, W, T_c, \) and \( \lambda \) for \( s_{exp} \))

- \( \rightarrow \) Optimal speed: maximum between \( s^* \) and \( s_{exp} \) or \( s_{wc} \)
Consider now that \( s \neq \sigma \) (multiple speeds): two unknowns

- \( \mathbb{E}(E) \) is a function of \( s \) and \( \sigma \):
  \[
  \mathbb{E}(E)(s, \sigma) = (Ws^2 + E_C) + \lambda \left( \frac{W}{s} + T_C \right)(W\sigma^2 + E_C)
  \]

- Lemma: energy minimized when deadline tight (both for wc and exp)

- \( \sim \) \( \sigma \) expressed as a function of \( s \):
  \[
  \sigma_{\text{exp}} = \frac{\lambda W}{\frac{W}{s} + T_C - (1+\lambda T_C)}, \quad \sigma_{\text{wc}} = \frac{W}{(D-2T_C)s - Ws}
  \]

→ Minimization of single-variable function, can be solved numerically (no expression of optimal \( s \))
Outline

1. Framework
2. With a single chunk
3. With several chunks
4. Simulation results
5. Conclusion
General problem with multiple chunks

- **Divisible task of size** $W$
- **Split into** $n$ **chunks of size** $W_i$: $\sum_{i=1}^{n} W_i = W$
- **Chunk** $i$ is executed once at speed $s_i$, and re-executed (if necessary) at speed $\sigma_i$
- **Unknowns:** $n$, $W_i$, $s_i$, $\sigma_i$

$$E(E) = \sum_{i=1}^{n} (W_i s_i^2 + E_C) + \lambda \sum_{i=1}^{n} \left( \frac{W_i}{s_i} + T_C \right) (W_i \sigma_i^2 + E_C)$$
Multiple chunks and single speed

With a single speed, $\sigma_i = s_i$ for each chunk

- Theorem: in optimal solution, $n$ equal-sized chunks $(W_i = \frac{W}{n})$, executed at same speed $s_i = s$
  - Proof by contradiction: consider two chunks $W_1$ and $W_2$
    executed at speed $s_1$ and $s_2$, with either $s_1 \neq s_2$, or $s_1 = s_2$ and $W_1 \neq W_2$
  - $\Rightarrow$ Strictly better solution with two chunks of size $w = (W_1 + W_2)/2$ and same speed $s$

- Only two unknowns, $s$ and $n$
- Minimum speed with $n$ chunks: $s^*_\text{exp}(n) = \frac{W}{2} \left(1 + 2\lambda T_C + \sqrt{\frac{4\lambda D}{n}} + 1\right)
  \frac{2(D - nT_C(1 + \lambda T_C))}{2(D - nT_C(1 + \lambda T_C))}$

\[ \rightarrow \text{Minimization of double-variable function, can be solved numerically both for expected and hard deadline} \]
Multiple chunks and multiple speeds

Need to find $n$, $W_i$, $s_i$, $\sigma_i$

- With expected deadline:
  - All re-execution speeds are equal ($\sigma_i = \sigma$) and tight deadline
  - All chunks have same size and are executed at same speed

- With hard deadline:
  - If $s_i = s$ and $\sigma_i = \sigma$, then all $W_i$'s are equal
  - **Conjecture:** equal-sized chunks, same first-execution / re-execution speeds

- $\sigma$ as a function of $s$, bound on $s$ given $n$

→ Minimization of double-variable function, can be solved numerically
Outline

1. Framework
2. With a single chunk
3. With several chunks
4. Simulation results
5. Conclusion
Simulation settings

- Large set of simulations: illustrate differences between models
- **Maple** software to solve problems
- We plot relative energy consumption as a function of $\lambda$
  - **The lower the better**
  - Given a deadline constraint (hard or expected), normalize with the result of **single-chunk single-speed**
  - **Impact of the constraint:** normalize expected deadline with hard deadline
- Parameters varying within large ranges
Comparison with single-chunk single-speed

- Results identical for any value of $W/D$

- For expected deadline, with small $\lambda (< 10^{-2})$, using multiple chunks or multiple speeds do not improve energy ratio: re-execution term negligible; increasing $\lambda$: improvement with multiple chunks

- For hard deadline, better to run at high speed during second execution: use multiple speeds; use multiple chunks if frequent failures
Important differences for single speed models, confirming previous conclusions: with hard deadline, use multiple speeds

Multiple speeds: no difference for small $\lambda$: re-execution at maximum speed has little impact on expected energy consumption; increasing $\lambda$: more impact of re-execution, and expected deadline may use slower re-execution speed, hence reducing energy consumption
Outline

1. Framework
2. With a single chunk
3. With several chunks
4. Simulation results
5. Conclusion
Conclusion

- **Energy consumption** of a divisible load workload on volatile platforms
- **Soft** or **hard** deadline constraint

**Theoretical side:**
- Formal models for the problem
- Expression of solutions as functions to minimize
- With multiple chunks, use **same size chunks, same speed, and same re-execution speed** (conjecture for multiple-speed hard-deadline)

**Simulations:**
- Single-chunk single-speed is very good for expected deadline
- Hard deadline and small $\lambda$: use **multiple speeds**
- Large values of $\lambda$: use **multiple speeds and multiple chunks**
What we had:

Energy-aware checkpointing
+ frequency scaling

What we aim at: