Static Strategies for Worksharing with Unrecoverable Interruptions

Anne Benoit, Yves Robert,
Arnold Rosenberg and Frédéric Vivien

École Normale Supérieure de Lyon
Yves.Robert@ens-lyon.fr
http://graal.ens-lyon.fr/~yrobert

Asheville, September 2008

My historical perspective

- I made it to the 9 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I talked about a nice little scheduling problem in 2006
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 9 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I talked about a nice little scheduling problem in 2006
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 9 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I talked about a nice little scheduling problem in 2006
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it [illegible] 9 CCGSC workshops!
- I talked about a [illegible] scheduling [illegible] problem in 1992
- I talked about a nice little [illegible] problem in 1994
- I talked about a nice [illegible] g problem in 1996
- I talked about a [illegible] scheduling [illegible] in 1998
- I talked about [illegible] little scheduling [illegible] in 2000
- I talked ab[illegible] a nice little scheduling proble[illegible] in 2002
- I talked [illegible] 004
- I ta[illegible] 06
- [illegible] onde[illegible]

Maybe [illegible] n! ☺

Or rather,
a fundamental problem
in cloud computing?!

Outline

1. Problem description
2. Technical framework
3. Single remote computer
4. Two remote computers
5. p remote computers

Outline

1. Problem description
2. Technical framework
3. Single remote computer
4. Two remote computers
5. p remote computers

Problem

- Large divisible computational workload
- Assemblage of p identical computers
- Unrecoverable interruptions
- A-priori knowledge of risk (failure probability)

Goal: maximize expected amount of work done

Related work

- Landmark paper by Bhatt, Chung, Leighton & Rosenberg on cycle stealing
- Hardware failures

☺ **Fault tolerant computing (hence scheduling) unavoidable for top500 machines, grids and clouds**

☹ **Well, same story told since first CCGSC?**

Related work

- Landmark paper by Bhatt, Chung, Leighton & Rosenberg on cycle stealing
- Hardware failures

☺ **Fault tolerant computing (hence scheduling) unavoidable for top500 machines, grids and clouds**

☹ **Well, same story told since first CCGSC?**

Related work

- Landmark paper by Bhatt, Chung, Leighton & Rosenberg on cycle stealing
- Hardware failures

☺ **Fault tolerant computing (hence scheduling) unavoidable for top500 machines, grids and clouds**

☹ **Well, same story told since first CCGSC?**

Chunking

- Sending each remote computer **large** amounts of work:
  ☺ decrease message packaging overhead
  ☹ maximize vulnerability to interruption-induced losses

- Sending each remote computer **small** amounts of work:
  ☺ minimize vulnerability to interruption-induced losses
  ☹ maximize message packaging overhead

Chunking

- Sending each remote computer **large** amounts of work:
  ☺ decrease message packaging overhead
  ☹ maximize vulnerability to interruption-induced losses

- Sending each remote computer **small** amounts of work:
  ☺ minimize vulnerability to interruption-induced losses
  ☹ maximize message packaging overhead

Replication

- Replicating tasks (same work sent to $q \geq 2$ remote computers):
  ☺ lessen vulnerability to interruption-induced losses
  ☹ minimize opportunities for "parallelism" and productivity

- Communication/control to/of remote computers **costly**
  ⇒ orchestrate task replication statically
  ☹ duplicate work unnecessarily when few interruptions
  ☺ prevent server from becoming bottleneck

Replication

- Replicating tasks (same work sent to $q \geq 2$ remote computers):
  ☺ lessen vulnerability to interruption-induced losses
  ☹ minimize opportunities for "parallelism" and productivity
- Communication/control to/of remote computers **costly**
  $\Rightarrow$ orchestrate task replication statically
  ☹ duplicate work unnecessarily when few interruptions
  ☺ prevent server from becoming bottleneck

Risk increases with time

	A	B	C	D
P_1	1	2	3	4

Risk increases with time

	A	B	C	D
P_1	1	2	3	4
P_2				

Risk increases with time

	A	B	C	D
P_1	1	2	3	4
P_2	4	3	2	1

Risk increases with time

	A	B	C	D
P_1	1	2	3	4
P_2	4	3	2	1
P_3				

Risk increases with time

	A	B	C	D
P_1	1	2	3	4
P_2	4	3	2	1
P_3	4	3	2	1

Risk increases with time

	A	B	C	D
P_1	1	2	3	4
P_2	4	3	2	1
P_3	4	3	2	1

	A	B	C	D
P_1	1	2	3	4
P_2	4	3	1	2
P_3	3	2	4	1

Outline

Interruption model

$$dPr = \begin{cases} \kappa dt & \text{for} \quad t \in [0, 1/\kappa] \\ 0 & \text{otherwise} \end{cases}$$

$$Pr(w) = \min\left\{1, \int_0^w \kappa dt\right\} = \min\{1, \kappa w\}$$

Goal: maximize expected work production

Interruption model

$$dPr = \begin{cases} \kappa dt & \text{for} \quad t \in [0, 1/\kappa] \\ 0 & \text{otherwise} \end{cases}$$

$$Pr(w) = \min\left\{1, \int_0^w \kappa dt\right\} = \min\{1, \kappa w\}$$

Goal: maximize expected work production

Free-initiation model (1/2)

Regimen Θ: allocate whole workload on a single computer

$$E^{(\mathrm{f})}(\text{jobdone}, \Theta) \ = \int_0^{\infty} Pr(\text{jobdone} \geq u \text{ under } \Theta) \ du$$

Single chunk

$$E^{(\mathrm{f})}(W, \Theta_1) \ = \ W\,(1 - Pr(W))$$

Two chunks with $\omega_1 + \omega_2 = W$

$$E^{(\mathrm{f})}(W, \Theta_2) = \omega_1(1 - Pr(\omega_1)) \ + \ \omega_2(1 - Pr(\omega_1 + \omega_2))$$

Free-initiation model (1/2)

Regimen Θ: allocate whole workload on a single computer

$$E^{(\mathrm{f})}(\text{jobdone}, \Theta) \;=\; \int_0^\infty Pr(\text{jobdone} \geq u \text{ under } \Theta)\; du$$

Single chunk

$$E^{(\mathrm{f})}(W, \Theta_1) \;=\; W\,(1 - Pr(W))$$

Two chunks with $\omega_1 + \omega_2 = W$

$$E^{(\mathrm{f})}(W, \Theta_2) = \omega_1(1 - Pr(\omega_1)) \;+\; \omega_2(1 - Pr(\omega_1 + \omega_2))$$

Free-initiation model (1/2)

Regimen Θ: allocate whole workload on a single computer

$$E^{(\mathrm{f})}(\text{jobdone}, \Theta) \;=\; \int_0^\infty Pr(\text{jobdone} \geq u \text{ under } \Theta)\; du$$

Single chunk

$$E^{(\mathrm{f})}(W, \Theta_1) \;=\; W\,(1 - Pr(W))$$

Two chunks with $\omega_1 + \omega_2 = W$

$$E^{(\mathrm{f})}(W, \Theta_2) = \omega_1(1 - Pr(\omega_1)) \;+\; \omega_2(1 - Pr(\omega_1 + \omega_2))$$

Free-initiation model (2/2)

With n chunks, maximize

$$E^{(\mathrm{f})}(W,n) = \omega_1(1 - Pr(\omega_1)) \; + \; \omega_2(1 - Pr(\omega_1 + \omega_2)) \\ \cdots + \; \omega_n(1 - Pr(\omega_1 + \cdots + \omega_n))$$

where

$$\omega_1 > 0, \; \omega_2 > 0, \ldots, \; \omega_n > 0$$

$$\omega_1 + \omega_2 + \cdots + \omega_n \leq W$$

Free-initiation model (2/2)

With n chunks, maximize

$$E^{(\mathrm{f})}(W,n) = \omega_1(1 - Pr(\omega_1)) \; + \; \omega_2(1 - Pr(\omega_1 + \omega_2))$$
$$\cdots + \; \omega_n(1 - Pr(\omega_1 + \cdots + \omega_n))$$

where

$$\omega_1 > 0, \; \omega_2 > 0, \ldots, \; \omega_n > 0$$

$$\omega_1 + \omega_2 + \cdots + \omega_n \leq W$$

Charged-initiation model

$$E^{(c)}(\text{jobdone}) = \int_0^\infty Pr(\text{jobdone} \geq u + \varepsilon)\; du.$$

Single chunk

$$E^{(c)}(W,1) = W\,(1 - Pr(W+\varepsilon))$$

Two chunks with $\omega_1 + \omega_2 \leq W$

$$E^{(c)}(W,2) = \omega_1(1 - Pr(\omega_1+\varepsilon)) + \omega_2(1 - Pr(\omega_1+\omega_2+2\varepsilon))$$

Charged-initiation model

$$E^{(c)}(\text{jobdone}) \;=\; \int_0^\infty Pr(\text{jobdone} \geq u + \varepsilon)\; du.$$

Single chunk

$$E^{(c)}(W, 1) \;= W\,(1 - Pr(W + \varepsilon))$$

Two chunks with $\omega_1 + \omega_2 \leq W$

$$E^{(c)}(W, 2) \;= \omega_1(1 - Pr(\omega_1 + \varepsilon)) + \omega_2(1 - Pr(\omega_1 + \omega_2 + 2\varepsilon))$$

Charged-initiation model

$$E^{(c)}(\text{jobdone}) \;=\; \int_0^{\infty} Pr(\text{jobdone} \geq u + \varepsilon)\; du.$$

Single chunk

$$E^{(c)}(W,1) \;= W\,(1 - Pr(W+\varepsilon))$$

Two chunks with $\omega_1 + \omega_2 \leq W$

$$E^{(c)}(W,2) \;= \omega_1(1 - Pr(\omega_1+\varepsilon)) + \omega_2(1 - Pr(\omega_1+\omega_2+2\varepsilon))$$

Relating the two models

Theorem

$$E^{(\mathrm{f})}(W,n) \;\geq\; E^{(\mathrm{c})}(W,n) \;\geq\; E^{(\mathrm{f})}(W,n) - n\varepsilon$$

Outline

Free-initiation model

$$E^{(f)}(W, \Theta_1) = W - \kappa W^2$$

$$\begin{aligned} E^{(f)}(W, \Theta_2) &= \omega_1(1 - \omega_1\kappa) + \omega_2(1 - (\omega_1 + \omega_2)\kappa)) \\ &= E^{(f)}(W, \Theta_1) + \omega_1\omega_2\kappa \end{aligned}$$

Theorem

Optimal schedule to deploy $W \in [0, \frac{1}{\kappa}]$ units of work in n chunks: use identical chunks of size Z/n:

$$Z = \min\left\{W, \frac{n}{n+1}\frac{1}{\kappa}\right\}$$

$$E^{(f)}(W, n) = Z - \frac{n+1}{2n}Z^2\kappa$$

Free-initiation model

$$E^{(\mathrm{f})}(W,\Theta_1) = W - \kappa W^2$$

$$\begin{aligned} E^{(\mathrm{f})}(W,\Theta_2) &= \omega_1(1-\omega_1\kappa) + \omega_2(1-(\omega_1+\omega_2)\kappa)) \\ &= E^{(\mathrm{f})}(W,\Theta_1) + \omega_1\omega_2\kappa \end{aligned}$$

Theorem

Optimal schedule to deploy $W \in [0, \frac{1}{\kappa}]$ units of work in n chunks: use identical chunks of size Z/n:

$$Z = \min\left\{W, \frac{n}{n+1}\frac{1}{\kappa}\right\}$$

$$E^{(\mathrm{f})}(W,n) = Z - \frac{n+1}{2n} Z^2 \kappa$$

Charged-initiation model

Theorem

Optimal schedule to deploy $W \in [0, \frac{1}{\kappa}]$ units of work in n chunks (assume $\min(W, \frac{1}{\kappa}) \geq \frac{n(n+1)}{2}\varepsilon$):

$$\omega_{1,n} = \frac{Z}{n} + \frac{n+1}{2}\varepsilon - \varepsilon$$

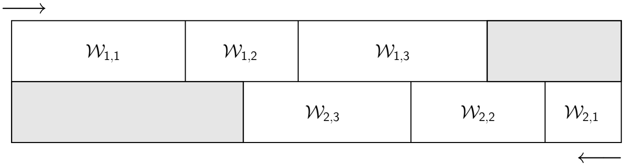
$$\omega_{i+1,n} = \omega_{i,n} - \varepsilon$$

$$Z = \min\left\{W, \frac{n}{n+1}\frac{1}{\kappa} - \frac{n}{2}\varepsilon\right\}$$

$$E^{(c)}(W, n) = Z - \frac{n+1}{2n}Z^2\kappa - \frac{n+1}{2}Z\varepsilon\kappa + \frac{(n-1)n(n+1)}{24}\varepsilon^2\kappa$$

Outline

General shape of optimal solution



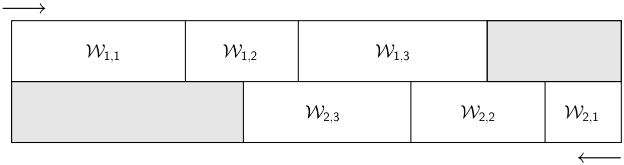
Theorem

W_1 and $W2$ assigned workloads in optimal solution:

1. Either $W_1 \bigcap W2 = \emptyset$ or $W_1 \bigcup W2 = W$
2. P_1 processes $W_1 \setminus W2$ before $W_1 \bigcap W2$
3. P_1 and P_2 process $W_1 \bigcap W2$ in reverse order

☹ **Optimal out of reach even for 2 or 3 chunks per processor**

General shape of optimal solution



Theorem

W_1 and $W2$ assigned workloads in optimal solution:

1. Either $W_1 \bigcap W2 = \emptyset$ or $W_1 \bigcup W2 = W$
2. P_1 processes $W_1 \setminus W2$ before $W_1 \bigcap W2$
3. P_1 and P_2 process $W_1 \bigcap W2$ in reverse order

☹ **Optimal out of reach even for 2 or 3 chunks per processor**

Algorithm (at most n chunks per computer)

If $W \geq \frac{2}{\kappa}$ **then**

$\forall i \in [1, n]$, $\mathcal{W}_{1,i} = \left[\frac{i-1}{n} \frac{n}{n+1} \frac{1}{\kappa}, \frac{i}{n} \frac{n}{n+1} \frac{1}{\kappa}\right]$

$\forall i \in [1, n]$, $\mathcal{W}_{2,i} = \left[W - \frac{i}{n} \frac{n}{n+1} \frac{1}{\kappa}, W - \frac{i-1}{n} \frac{n}{n+1} \frac{1}{\kappa}\right]$

If $W \leq \frac{1}{\kappa}$ **then**

$\forall i \in [1, n]$, $\mathcal{W}_{1,i} = \mathcal{W}_{2,n-i+1} = \left[\frac{i-1}{n} W, \frac{i}{n} W\right]$

If $\frac{1}{\kappa} < W \frac{2}{\kappa}$ **then**

$l \leftarrow \left\lfloor \frac{n}{3} \right\rfloor$

$\forall i \in [1, l]$, $\mathcal{W}_{1,i} = \left[\frac{i-1}{l}(W - \frac{1}{\kappa}), \frac{i}{l}(W - \frac{1}{\kappa})\right]$

$\forall i \in [1, l]$, $\mathcal{W}_{2,i} = \left[W - \frac{i}{l}(W - \frac{1}{\kappa}), W - \frac{i-1}{l}(W - \frac{1}{\kappa})\right]$

$\forall i \in [1, 2l]$, $\mathcal{W}_{1,l+i} = \mathcal{W}_{2,3l-i+1} =$
$\left[(W - \frac{1}{\kappa}) + \frac{i-1}{2l}(\frac{2}{\kappa} - W), (W - \frac{1}{\kappa}) + \frac{i}{2l}(\frac{2}{\kappa} - W)\right]$

Algorithm (at most n chunks per computer)

Theorem

Previous algorithm is:

1. Optimal when $W \geq 2\frac{1}{\kappa}$:

$$E^{(\mathrm{f},2)}(W,n) = \frac{n-1}{n}\frac{1}{\kappa} \xrightarrow[n\to\infty]{} \frac{1}{\kappa};$$

2. Asymptotically optimal when $W \leq \frac{1}{\kappa}$

$$E^{(\mathrm{f},2)}(W,n) = W - \frac{W^3\kappa^2}{6}\left(1+\frac{3}{n}+\frac{2}{n^2}\right) \xrightarrow[n\to\infty]{} W - \frac{W^3\kappa^2}{6};$$

3. Asymptotically optimal when $\frac{1}{\kappa} < W < 2\frac{1}{\kappa}$

horrible formula for $E^{(\mathrm{f},2)}(W,n)$

$$E^{(\mathrm{f},2)}(W,n) \xrightarrow[n\to\infty]{} 2W - \frac{1}{3}\frac{1}{\kappa} - W^2\kappa + \frac{W^3\kappa^2}{6}.$$

Algorithm (at most n chunks per computer)

Theorem

Previous algorithm is:

1. Optimal when $W \geq 2\frac{1}{\kappa}$:

$$E^{([illegible])}(W, n) = \frac{n-1}{n}\frac{1}{\kappa} \xrightarrow[n\to\infty]{} \frac{1}{\kappa};$$

2. Asymptotically optimal when $W \leq \frac{1}{\kappa}$

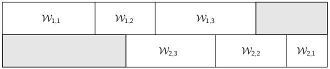
$$E^{(\mathrm{f},2)}(W, n) = W - \frac{W^3\kappa^2}{6}\left(1 + \frac{3}{n} + \frac{2}{n^2}\right) \xrightarrow[n\to\infty]{} W - \frac{W^3\kappa^2}{6};$$

3. Asymptotically optimal when $\frac{1}{\kappa} < W < 2\frac{1}{\kappa}$

Getting lost?!

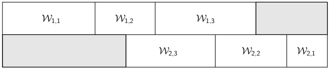
$$E^{(\mathrm{f},2)}(W, n) \xrightarrow[n\to\infty]{} 2W - \frac{1}{3}\frac{1}{\kappa} - W^2\kappa + \frac{W^3\kappa^2}{6}$$

Asymptotically optimal solution when $W \leq \frac{1}{\kappa}$

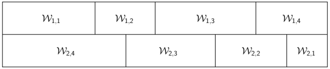


Optimal scheduling with n chunks

Asymptotically optimal solution when $W \leq \frac{1}{\kappa}$

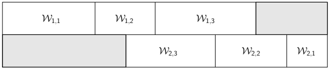


Optimal scheduling with n chunks

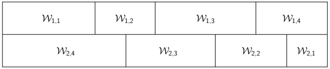


Solution extended with $(n+1)$-st chunk

Asymptotically optimal solution when $W \leq \frac{1}{\kappa}$



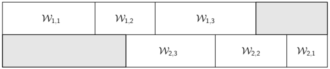
Optimal scheduling with n chunks



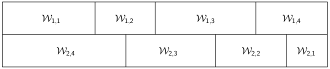
Solution extended with $(n+1)$-st chunk

Dividing chunks so that boundaries coincide

Asymptotically optimal solution when $W \leq \frac{1}{\kappa}$

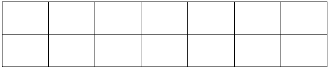


Optimal scheduling with n chunks



Solution extended with $(n+1)$-st chunk

Dividing chunks so that boundaries coincide



Solution returned by algorithm with $2n+1$ equal-size chunks

Outline

Pragmatic approach

- Difficult ⇒ only heuristics!
- Partition
  - workload into slices
  - resources into groups
- Replicate each slice on every processor in its group

Pragmatic approach

- Difficult ⇒ only heuristics!
- Partition
  - workload into slices
  - resources into groups
- Replicate each slice on every processor in its group

Pragmatic approach

- Difficult ⇒ only heuristics!
- Partition
  - workload into slices
  - resources into groups
- Replicate each slice on every processor in its group

Pragmatic approach

- Difficult ⇒ only heuristics!
- Partition
  - workload into slices
  - resources into groups
- Replicate each slice on every processor in its group

. . . and orchestrate execution!

	A	B	C	D
P_1	1	2	3	4
P_2	4	3	1	2
P_3	3	2	4	1

Partitioning

- Small $W \leq \frac{1}{\kappa}$: single slice, replicated on all p computers

- Large $W \geq p\frac{1}{\kappa}$: p independent slices of size $\frac{1}{\kappa}$

- General case $\frac{1}{\kappa} < W < p\frac{1}{\kappa}$:
  - partition work into $q = \lceil W\kappa \rceil$ slices of size $\mathsf{sl} = W/q$
  - deploy these q slices to disjoint subsets of computers
  - replicate each slice on either $\lfloor p/q \rfloor$ or $\lceil p/q \rceil$ computers

Partitioning

- Small $W \leq \frac{1}{\kappa}$: single slice, replicated on all p computers
- Large $W \geq p\frac{1}{\kappa}$: p independent slices of size $\frac{1}{\kappa}$
- General case $\frac{1}{\kappa} < W < p\frac{1}{\kappa}$:
  - partition work into $q = \lceil W\kappa \rceil$ slices of size sl $= W/q$
  - deploy these q slices to disjoint subsets of computers
  - replicate each slice on either $\lfloor p/q \rfloor$ or $\lceil p/q \rceil$ computers

Partitioning

- Small $W \leq \frac{1}{\kappa}$: single slice, replicated on all p computers
- Large $W \geq p\frac{1}{\kappa}$: p independent slices of size $\frac{1}{\kappa}$
- General case $\frac{1}{\kappa} < W < p\frac{1}{\kappa}$:
  - partition work into $q = \lceil W\kappa \rceil$ slices of size $\mathsf{sl} = W/q$
  - deploy these q slices to disjoint subsets of computers
  - replicate each slice on either $\lfloor p/q \rfloor$ or $\lceil p/q \rceil$ computers

Orchestrating

Chunk	1	2	3	4	5	6	7	8	9	10	11	12
P_1	1	6	9	12	2	5	8	11	3	4	7	10
P_2	12	1	6	9	11	2	5	8	10	3	4	7
P_3	9	12	1	6	8	11	2	5	7	10	3	4
P_4	6	9	12	1	5	8	11	2	4	7	10	3

Time-steps for execution of n = 12 chunks with g = 4 processors

Group schedules

Chunk	1	2	3	4	5	6	7	8	9	10	11	12
P_1	1	6	9	12	2	5	8	11	3	4	7	10
P_2	12	1	6	9	11	2	5	8	10	3	4	7
P_3	9	12	1	6	8	11	2	5	7	10	3	4
P_4	6	9	12	1	5	8	11	2	4	7	10	3

Group 1 chunks 1-4	Group 2 chunks 5-8	Group 3 chunks 9-12
1	2	3
6	5	4
9	8	7
12	11	10

Time-steps for group execution

Group schedules

Group 1	Group 2	Group 3
1	2	3
6	5	4
9	8	7
12	11	10

Group schedules

Group 1	Group 2	Group 3
1	2	3
6	5	4
9	8	7
12	11	10

↓

All four executions fail with probability proportional to $1 \times 6 \times 9 \times 12$

Group schedules

Group 1	Group 2	Group 3
1	2	3
6	5	4
9	8	7
12	11	10

↓

All four executions fail with probability proportional to $2 \times 5 \times 8 \times 11$

Group schedules

Group 1	Group 2	Group 3
1	2	3
6	5	4
9	8	7
12	11	10

↓

All four executions fail with probability proportional to $3 \times 4 \times 7 \times 10$

Group schedules

Group 1	Group 2	Group 3
1	2	3
6	5	4
9	8	7
12	11	10

All four executions fail with probability proportional to $3 \times 4 \times 7 \times 10$

$$K = \sum_{j=1}^{\frac{n}{g}} \prod_{i=1}^{g} G_{i,j} = 1.6.9.12 + 2.5.8.11 + 3.4.7.10$$

Better performance for small K

Scheduling objective

$$E(\mathsf{sl},\mathsf{n}) = \mathsf{sl}\left(1 - \frac{\mathsf{g}}{\mathsf{n}}\left(\frac{\mathsf{sl}\kappa}{\mathsf{n}}\right)^{\mathsf{g}} \sum_{j=1}^{\frac{\mathsf{n}}{\mathsf{g}}} \prod_{i=1}^{\mathsf{g}} G_{i,j}\right)$$

Problem

Minimize

$$\mathsf{K} = \sum_{j=1}^{\frac{\mathsf{n}}{\mathsf{g}}} \prod_{i=1}^{\mathsf{g}} G_{i,j}$$

where entries of G are a permutation of $[1..n]$

Bound

$$\mathsf{K}_{\mathsf{min}} = \left\lceil \frac{\mathsf{n}}{\mathsf{g}} (\mathsf{n}!)^{\frac{\mathsf{g}}{\mathsf{n}}} \right\rceil$$

Heuristics (1/3)

Group 1	Group 2	Group 3
1	2	3
4	5	6
7	8	9
10	11	12

(a) Cyclic: K = 3104

Group 1	Group 2	Group 3
1	2	3
6	5	4
9	8	7
12	11	10

(b) Reverse: K = 2368

Heuristics (2/3)

Group 1	Group 2	Group 3
1	2	3
4	5	6
9	8	7
12	11	10

(c) Mirror: K = 2572

Group 1	Group 2	Group 3
1	2	3
6	5	4
7	8	9
12	11	10

(d) Snake: K = 2464

Heuristics (3/3)

Group 1	Group 2	Group 3
1	2	3
8	6	4
9	7	5
10	11	12

(e) Worm: $K = 2364$

Step 1	1	2	3
CCP	1	2	3
Step 2	6	5	4
CCP	6	10	12
Step 3	9	8	7
CCP	54	80	84
Step 4	12	11	10

(f) Greedy: $K = 2368 \geq K_{min} = 2348$

Comparing group schedules for n = 9 and g = 3

1	2	3	1	2	3	1	2	3
4	5	6	6	5	4	6	5	4
7	8	9	7	8	9	9	8	7
$K_{cyclic} = 270$			$K_{snake} = 230$			$K_{reverse} = K_{greedy} = 218$		

1	2	3	1	2	3
8	6	4	8	5	4
9	7	5	9	7	6
$K_{worm} = 216$			$K_{optimal} = K_{min} = 214$		

Comparing group schedules for n = 20 and g = 4

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

$K_{cyclic} = 34104$

1	2	3	4	5
6	7	8	9	10
15	14	13	12	11
20	19	18	17	16

$K_{mirror} = 27284$

1	2	3	4	5
10	9	8	7	6
15	14	13	12	11
20	19	18	17	16

$K_{reverse} = 24396$

1	2	3	4	5
10	9	8	7	6
11	12	13	14	15
20	19	18	17	16

$K_{snake} = 25784$

1	2	3	4	5
14	12	10	8	6
15	13	11	9	7
16	17	18	19	20

$K_{worm} = 24276$

1	2	3	4	5
10	9	8	7	6
15	14	13	12	11
20	19	18	16	17

$K_{greedy} = 24390$

$K_{min} = 23780$

More on group schedules!

- Lower and upper performance bounds
- Extensive comparisons against greedy (re-balancing row-by-row)
- Lots of simulation results

Please see paper or ask us ☺

Conclusion

- Turned out much more difficult than expected (☺ or ☹?)
- Extension to resources with different risk functions
- Extension to resources with different computation capacities
- Master-slave approach with communication costs
- Comparison with dynamic approaches