Scheduling jobs under a variable number of processors

Joachim Cendrier[1] Anne Benoit[1] Frédéric Vivien[1]

[1]ENS Lyon, LIP, ROMA

Aussois, June 27, 2024

Motivation

- Growing concern about energy consumption and environmental impact of data centers
- Data centers with renewable energies ⇒ variable power
- Current algorithms inefficient: designed to run with a fixed amount of energy

- Our work: Conception and development of new scheduling algorithms to optimize Goodput and Yield, while handling power variations

Table of Contents

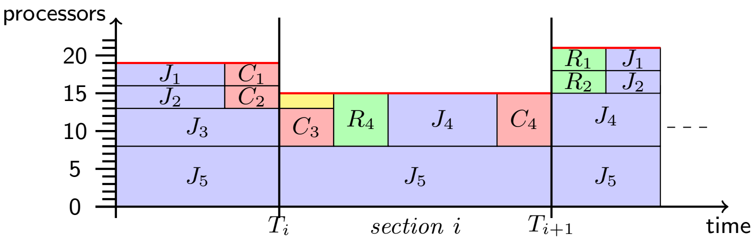
Model

Problem: Scheduling m infinite parallel rigid jobs under variable number of processors, in each *section*

- A job J_j can be checkpointed and recovered with cost in time C and R
- Knowledge of the length T_i and the number of available processors P_i for each section
- P is bounded in $[P_{\min}, P_{\max}]$ and $|P_{i+1} - P_i| \leq \delta$

Additional constraint:

- Never lose work (i.e., checkpoint enough before section change)



Goodput objective function

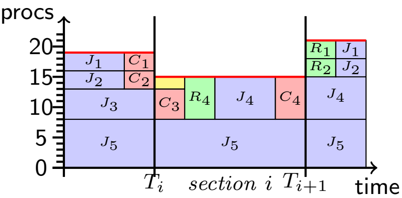
Goal: Maximize processor utilization during a section

$$Goodput([T_i, T_{i+1}]) = \frac{\sum_{j=1}^{m} p_j W_{j,i}}{P_i(T_{i+1} - T_i)}$$

where $W_{j,i}$ is the duration of the useful work done by job J_j during section $[T_i, T_{i+1}]$

Goodput maximization:

- Using as many processors as possible
- Minimizing time spent on checkpoints and recoveries
- **Consequence:** Some jobs may be **constantly running**, others can always be **idle**



Yield objective function

Goal: Maximize, at the end of each section, the minimum progress of all jobs

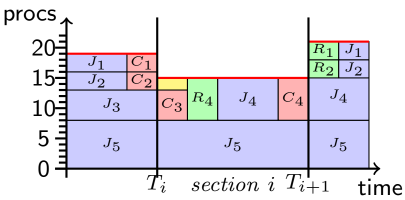
$$minYield(t) = \min_{j \in [1,m]} Yield(t,j)$$

$$Yield(t,j) = \frac{W_j(t)}{t}$$

where $W_j(t)$ is the duration of the useful work done by job J_j since the beginning

MinYield maximization:

- Changing running jobs
- **Consequence:** Higher total **checkpoint and recovery costs**

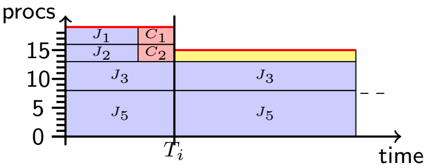


Algorithms

Goodput Algorithms

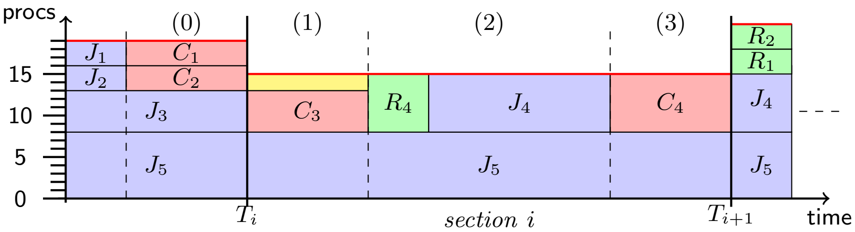
Optimize Goodput at the end of the section, with decisions at the end and at the beginning of the section:

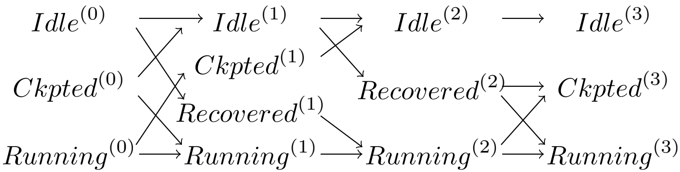
- Greedy Goodput:
  - End: checkpoint some jobs to ensure at least δ processors available
  - Beginning: sort jobs by decreasing number of processors, then select them as long as there are free processors for the section
  - Temporal Complexity: $\mathcal{O}(m)$
- Dynamic Programming by section (DP Goodput): DP that **maximizes the goodput**, deciding which jobs will be executed during the section
  - Temporal Complexity slow version: $\mathcal{O}(mP_{\max}^3)$
  - Temporal Complexity fast version: $\mathcal{O}(mP_{\max}^2)$



Algorithms

Dynamic Programming



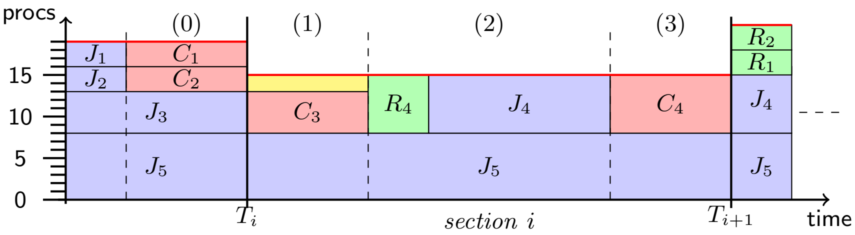


- 13 different state sequences for the jobs

Algorithms

Dynamic Programming

$$G_j(P_i^{(1)}, P_i^{(2)}, P_i^{(3)}) = G_{j-1}(P_i^{(1)} - \alpha p_j, P_i^{(2)} - \beta p_j, P_i^{(3)} - \gamma p_j) + p_j(\alpha W_j^{(1)} + \beta W_j^{(2)} + \gamma W_j^{(3)})$$



- Dynamic programming by adding jobs and processors of each phase one by one

Algorithms

Yield algorithms: similar than goodput algorithms.

Bi-criteria algorithms:

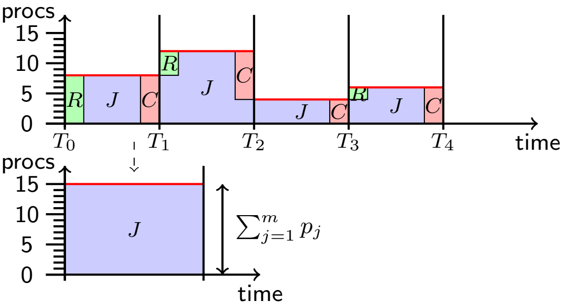
- Target Yield $target$: If target is satisfied, it is a DPG, otherwise it is a DPY
  - Temporal Complexity: $\mathcal{O}(mP_{\max}^2)$
- Target Goodput $target$: DP with an upper bound on **idle time** (unused processors and time spent on checkpoint and recover) per section
  - Temporal Complexity: $\mathcal{O}(mP_{\max}^3 T)$
- Dynamic Programming Bi by section (DP BiC Y): Same idea as for goodput, with a multiplying coefficient for Yield:
  - Temporal Complexity: $\mathcal{O}(mP_{\max}^2)$

$$G_j(args) = G_{j-1}(args) + \overbrace{p_j W_j}^{GoodputPart} \underbrace{(2 - Yield(j))^Y}_{YieldPart}$$

where W_j is the potentiel duration of the useful work for job J_j

Bounds

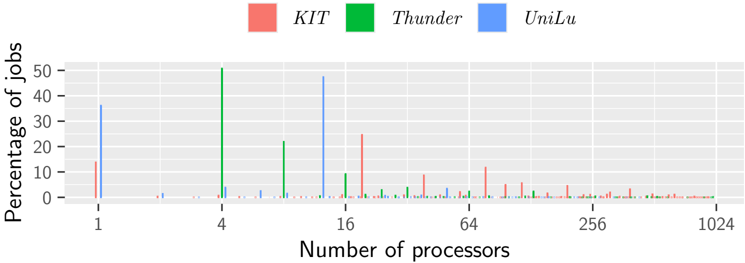
- **Max Goodput:** all processors are constantly used for useful work
- **Max minYield:** modular work area



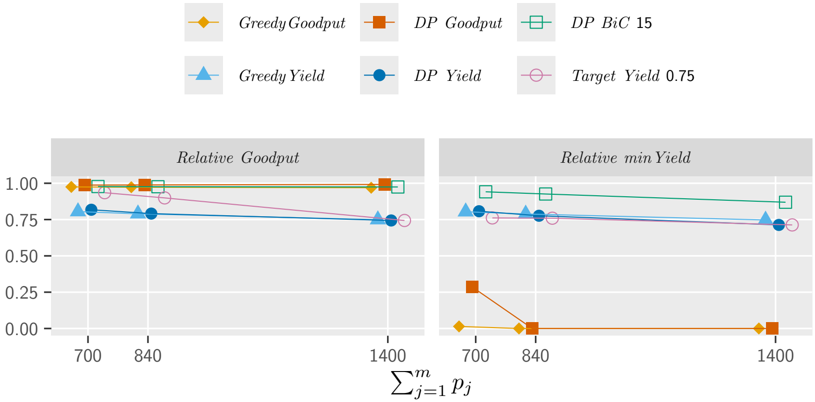
Goodput and minYield divided by Max Goodput and Max minYield respectively to obtain **relative** Goodput and **relative** minYield

Methodology

- 3 differents workloads for job sizes
- 1000 sections of average length egals to 5 checkpoints/recoveries
- Number of available processors is in $[140, 700]$ with variation up to 70 processors between two sections
- System is slightly overloaded
- Geometric mean on 10 instances

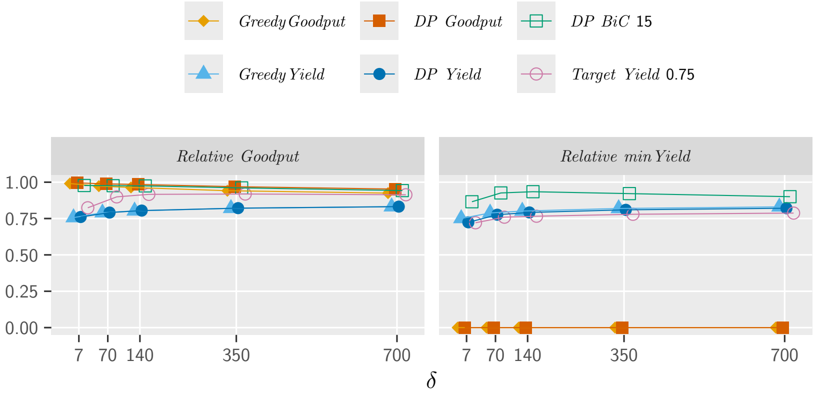


Impact of the overall load



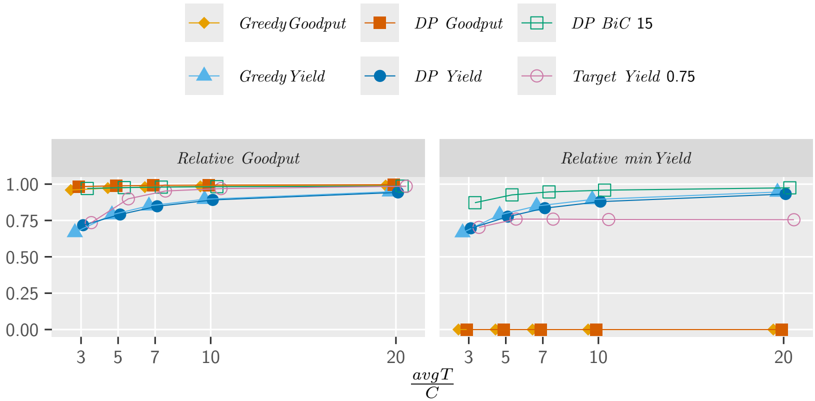
- Decrease in relative Goodput of target Yield
- Decrease of all relative minYield

Impact of variability on number of processors



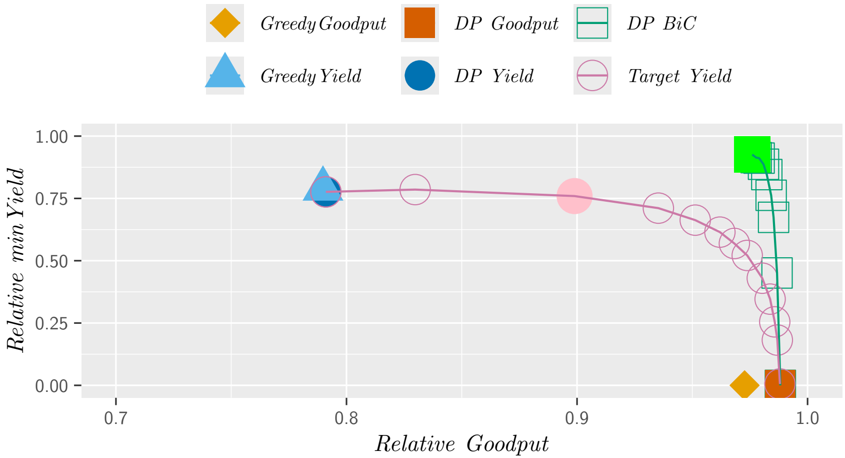
- Balancing of the relative Goodput for all algorithms
- Higher relative minYield

Impact of the average section length



- Small section duration implies greater impact of decisions

Pareto comparaison



- Trade-off Goodput-minYield
- But DP BiC achieves to maintain high Goodput with high Yield

Conclusion

- Modelisation of a scheduling problem with variable number of processors
- Design of multiple dynamic programming algorithms
- Simulations that give results close to the maximum bounds

- Future work:
  - Have jobs with a limited duration, release date, deadlines,...
  - Relaxing hypotheses on section