CR-08: Scheduling (Ordonnancement)

Loris Marchal, CR CNRS

Scheduling:

- part of Optimization Research
- allocate resources to tasks to optimize some performance metric

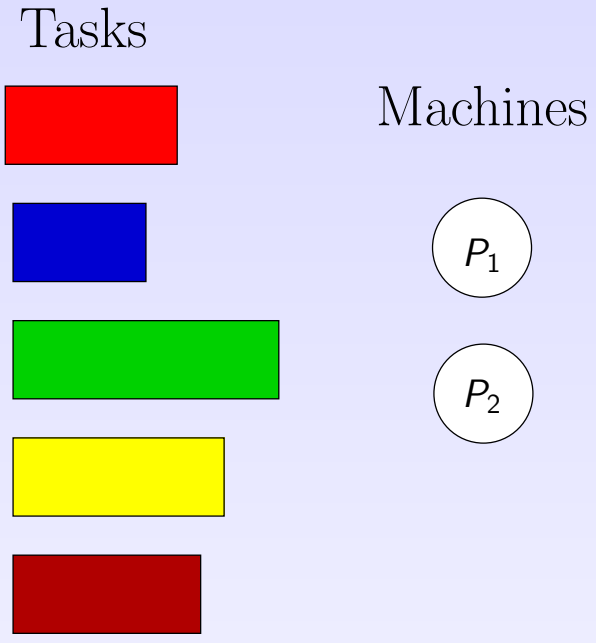
Context:

- computer systems
- distributed computing platforms

Pre-requisite (recommended):

- Parallel Algorithms (CB-04)

Exemple 1:

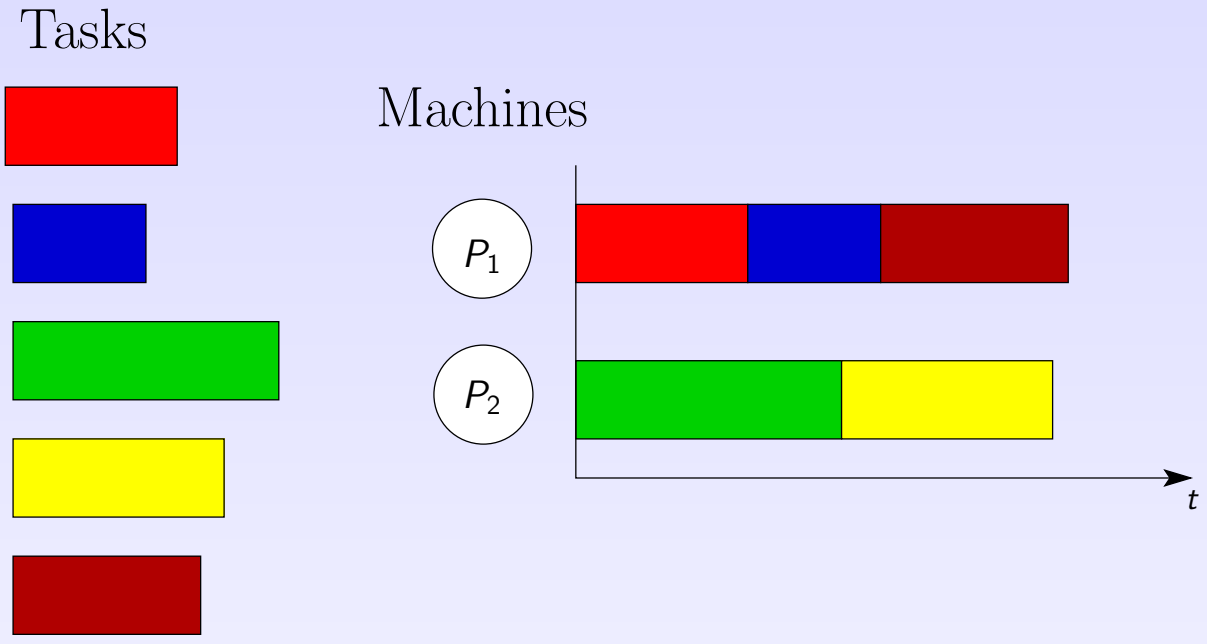


Objectives:

- Minimizing total execution time (C_{max})
- Minimizing average execution time $\sum_i C_i$
- Minimizing weighted sum of execution time $\sum_i w_i C_i$
- With deadlines, minimize the number of late jobs

Results: NP-completeness, algorithms, approximation algorithms,(in-)approximation bounds

Exemple 1:

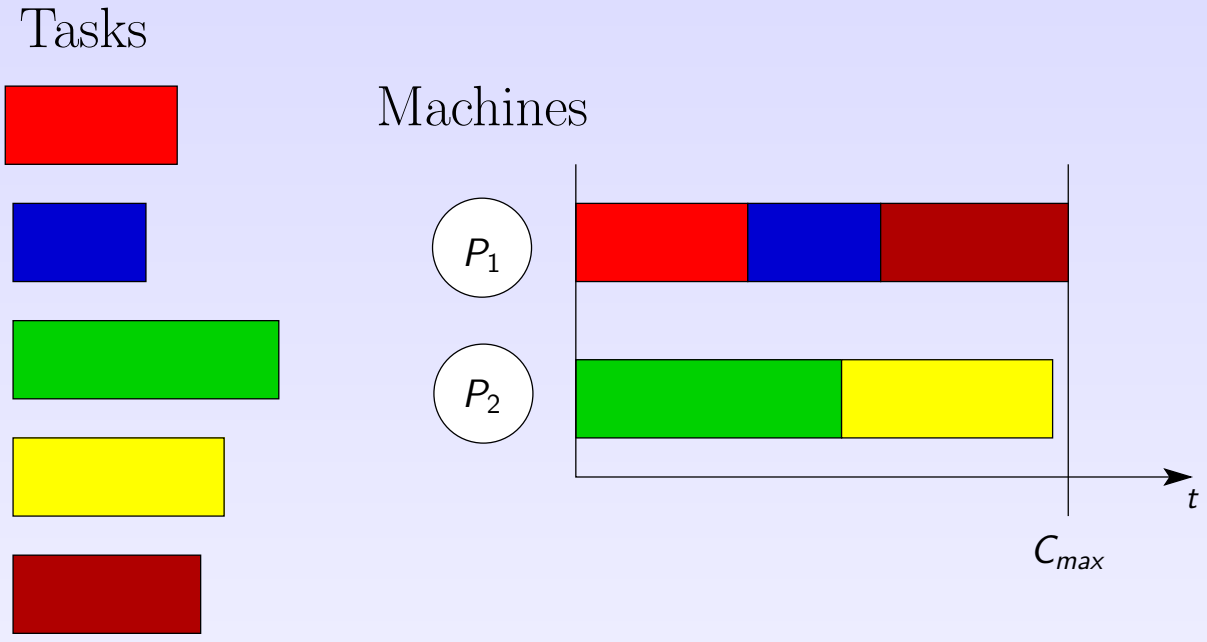


Objectives:

- Minimizing total execution time (C_{max})
- Minimizing average execution time $\sum_i C_i$
- Minimizing weighted sum of execution time $\sum_i w_i C_i$
- With deadlines, minimize the number of late jobs

Results: NP-completeness, algorithms, approximation algorithms,(in-)approximation bounds

Exemple 1:

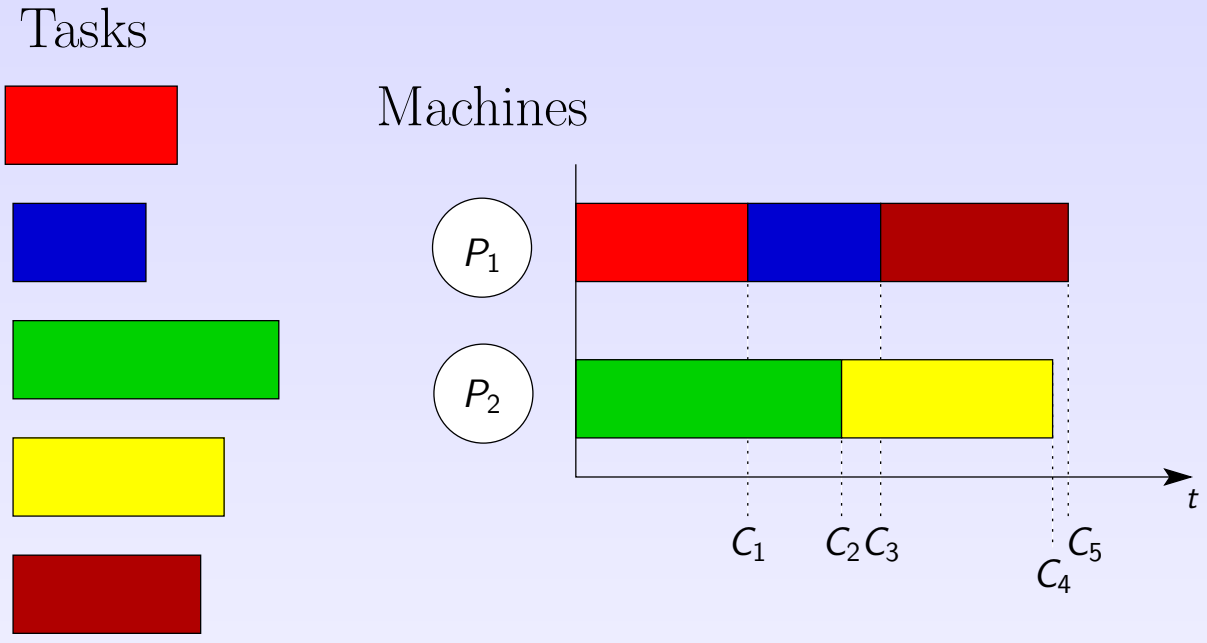


Objectives:

- Minimizing total execution time (C_{max})
- Minimizing average execution time $\sum_i C_i$
- Minimizing weighted sum of execution time $\sum_i w_i C_i$
- With deadlines, minimize the number of late jobs

Results: NP-completeness, algorithms, approximation algorithms,(in-)approximation bounds

Exemple 1:

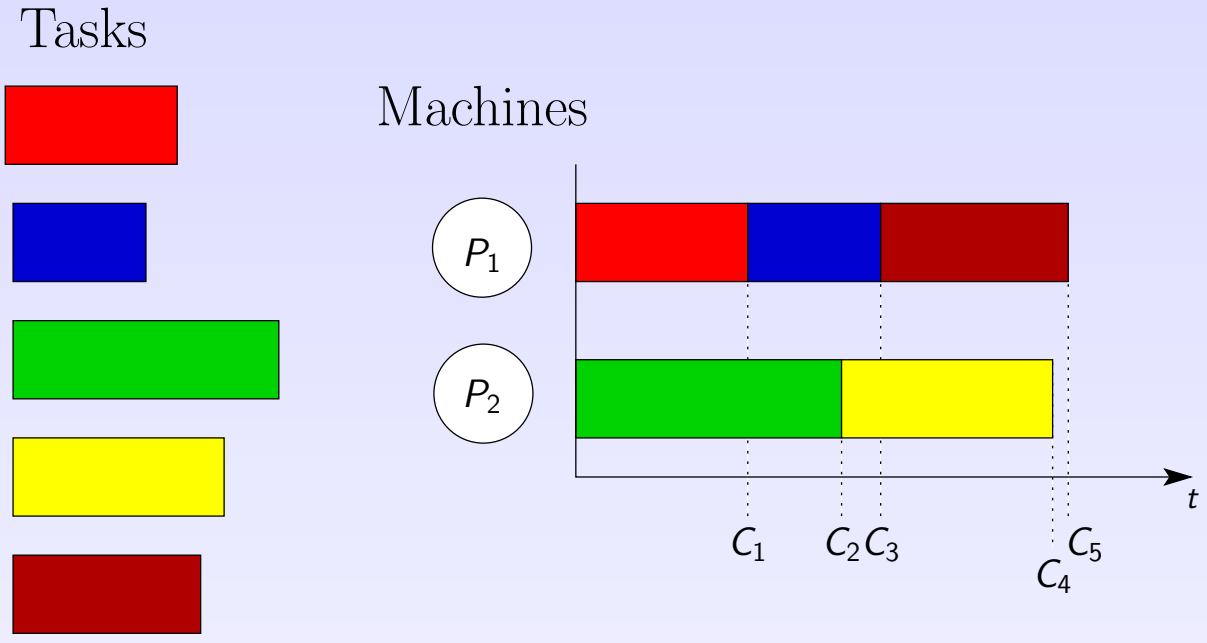


Objectives:

- Minimizing total execution time (C_{max})
- Minimizing average execution time $\sum_i C_i$
- Minimizing weighted sum of execution time $\sum_i w_i C_i$
- With deadlines, minimize the number of late jobs

Results: NP-completeness, algorithms, approximation algorithms,(in-)approximation bounds

Exemple 1:

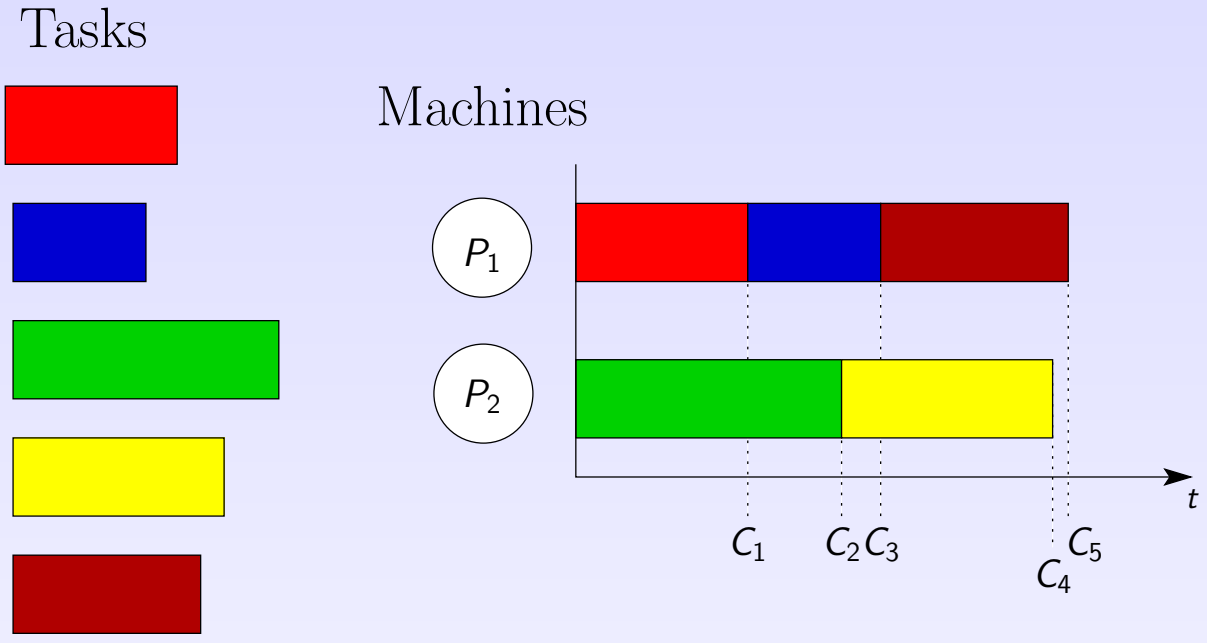


Objectives:

- Minimizing total execution time (C_{max})
- Minimizing average execution time $\sum_i C_i$
- Minimizing weighted sum of execution time $\sum_i w_i C_i$
- With deadlines, minimize the number of late jobs

Results: NP-completeness, algorithms, approximation algorithms,(in-)approximation bounds

Exemple 1:

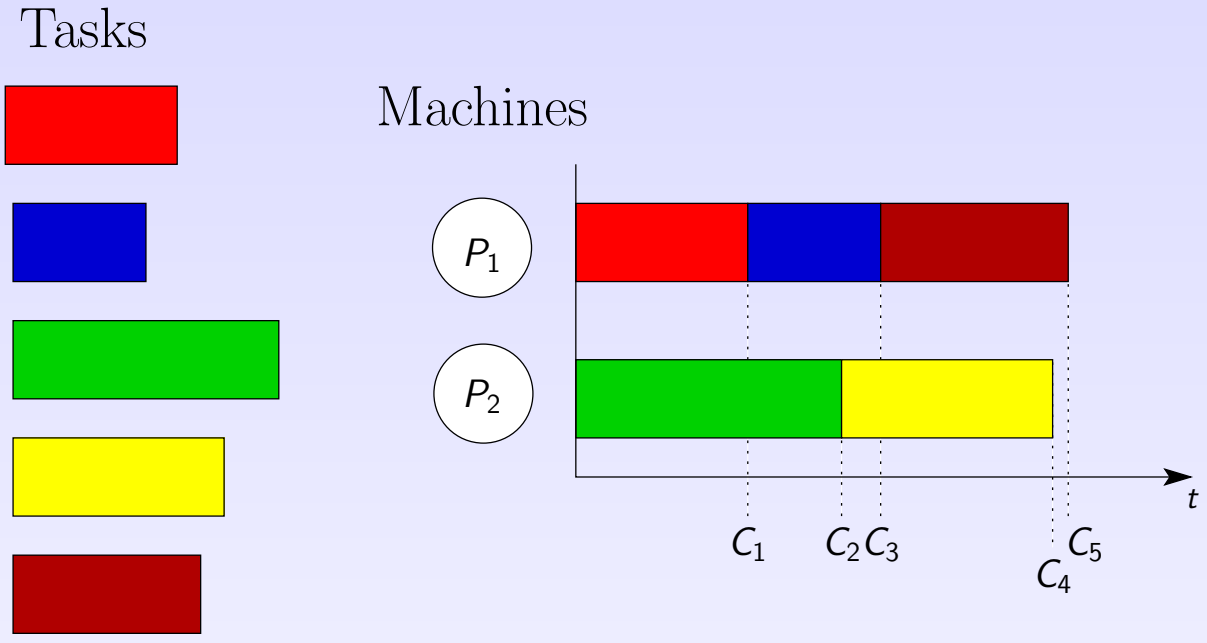


Objectives:

- Minimizing total execution time (C_{max})
- Minimizing average execution time $\sum_i C_i$
- Minimizing weighted sum of execution time $\sum_i w_i C_i$
- With deadlines, minimize the number of late jobs

Results: NP-completeness, algorithms, approximation algorithms,(in-)approximation bounds

Exemple 1:

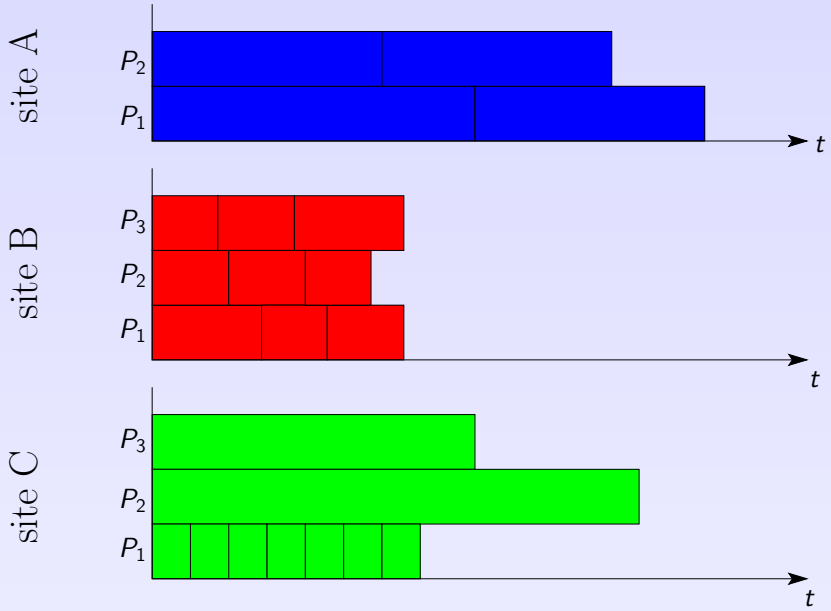


Objectives:

- Minimizing total execution time (C_{max})
- Minimizing average execution time $\sum_i C_i$
- Minimizing weighted sum of execution time $\sum_i w_i C_i$
- With deadlines, minimize the number of late jobs

Results: NP-completeness, algorithms, approximation algorithms,(in-)approximation bounds

Exemple 2:



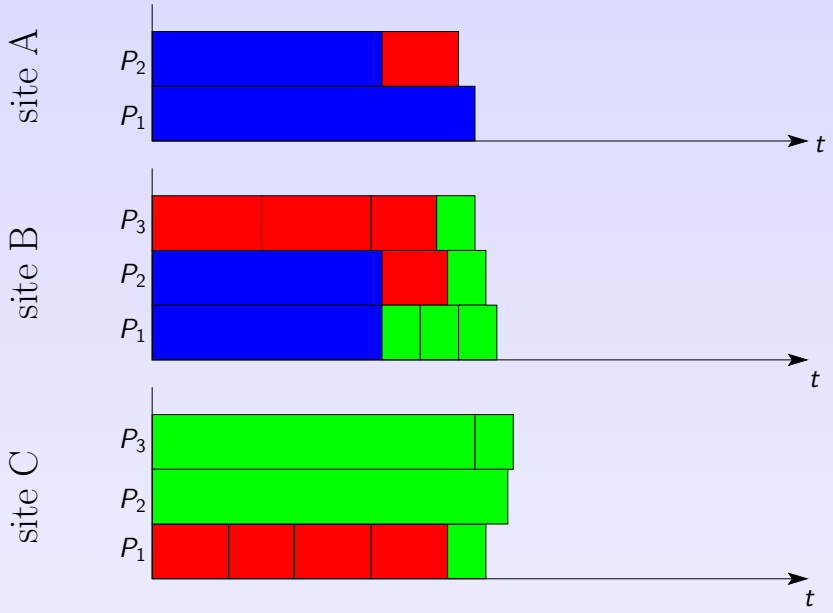
Context:

- Multiple organization have their own cluster
- Sharing resources to increase performance
- Selfishness: organization will not participate if the performance for its tasks is decreased

Results:

- Bounds on ratio with/without selfishness
- Approximation algorithms
- Use of game theory to model users...

Exemple 2:



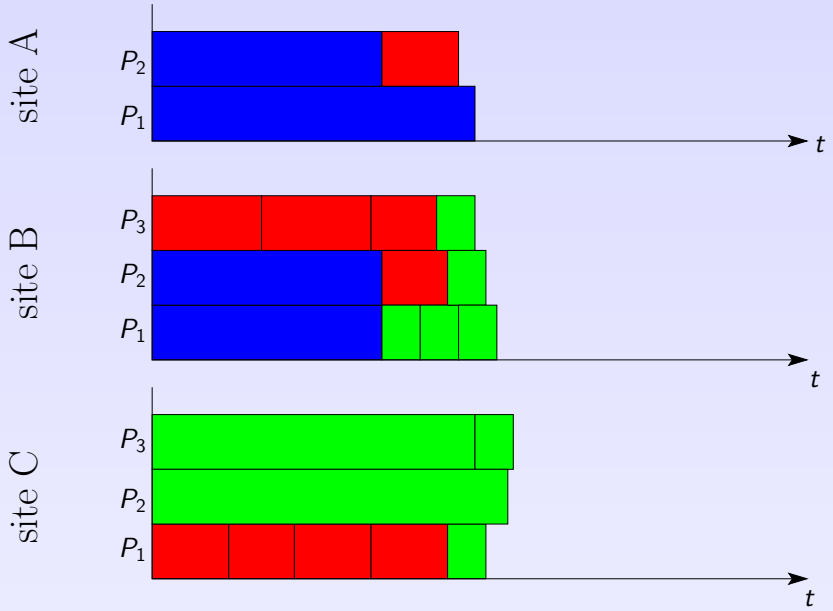
Context:

- Multiple organization have their own cluster
- Sharing resources to increase performance
- Selfishness: organization will not participate if the performance for its tasks is decreased

Results:

- Bounds on ratio with/without selfishness
- Approximation algorithms
- Use of game theory to model users...

Exemple 2:



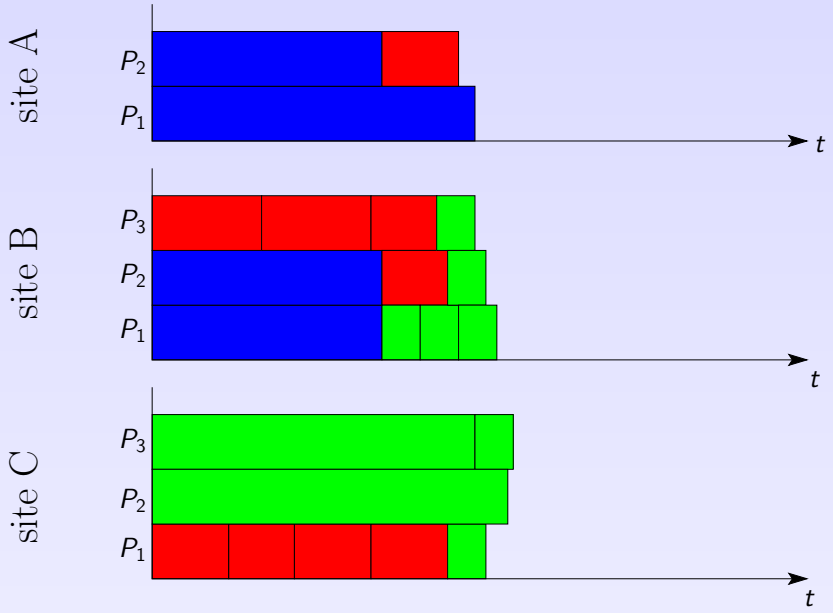
Context:

- Multiple organization have their own cluster
- Sharing resources to increase performance
- Selfishness: organization will not participate if the performance for its tasks is decreased

Results:

- Bounds on ratio with/without selfishness
- Approximation algorithms
- Use of game theory to model users...

Exemple 2:



Context:

- Multiple organization have their own cluster
- Sharing resources to increase performance
- Selfishness: organization will not participate if the performance for its tasks is decreased

Results:

- Bounds on ratio with/without selfishness
- Approximation algorithms
- Use of game theory to model users...

General outline:

1. Classical scheduling methods and results ($\sim$ 3 courses)
   - Classical problems (List scheduling, etc.)
   - Application modeling and corresponding results
   - NP-completness, approximation algorithms
   - Online and non-clairvoyant scheduling

2. Better models for computing platforms ($\sim$ 4 courses)
   - Introducing communication costs
   - Divisible Load Scheduling, Steady-State
   - Communication/Computation Interference

3. New objectives and trends in scheduling ($\sim$ 6 courses)
   - Fault tolerance, robustness, energy consumption
   - Multi-organization scheduling
   - Game theory and scheduling
   - Stochastic methods
   - Multi-criteria optimization

Organization

- French or English (on demand)
- A few courses to practice article analysis

Exams:

- Bibliographic study, on (provided) articles ($\sim$ 2 per student)

mail: Loris.Marchal@ens-lyon.fr or room 335, 3rd floor

More info: graal.ens-lyon.fr/~lmarchal/scheduling/