Multi-criteria scheduling

Denis Trystram
with the help of P-F. Dutot, K. Rzadca and E. Saule
LIG, Grenoble University, France

8 juin 2007

Outline

Outline

Outline

Outline

Outline

Outline

Outline

A preliminary story

Once upon a time two friends who want to gather some friends to share their interest for a topic (music, swimming, yoga or whatever) in a nice resort on the french riviera.

A preliminary story

Once upon a time two friends who want to gather some friends to share their interest for a topic (music, swimming, yoga or whatever) in a nice resort on the french riviera.

Official story :
Two senior researchers of a well-known Institut want to gather some colleagues and young researchers to disseminate the most recent scientific results on scheduling theory in a thematic school in the frame of the National Council of Research...

A preliminary story

The duration is 5 days.
They first selected some pairs (topic,speaker). Some are more popular than others (good topic with good speaker, but also bad topic with good speaker, etc.).
As they have to pay for the equipments for the whole week, they must determine the best repartition of talks for attracting the maximum number of participants.
Available time slots are not evenly distributed in the week (it is better to deliver a talk wenesday morning before the banquet instead of monday early morning when many participants are not still arrived or friday afternoon when most people already left...

Users and organizers point of view

The participants are very busy people, thus, they want to come the minimum time and attend the maximum of good talks.
The participants want to maximize the number of good pairs (topic,speaker) (they have payed for !).

The organizers want to maximize the number of participants in each lecture.

bi-criteria problem

This is a typical bi-criteria assignment problem.
It is easy to find a good solution for each criterion, however, it is not always satisfactory for the other...

Multi-objective optimization is a rather new topic which motivates a lot of people.

Optimizing one objective has been widely studied for many combinatorial problems including scheduling.

Scheduling is a problem that has many variants.
The most popular objective is the makespan which is informally defined as the time of the last finishing task (*completion time*) of an application represented by a precedence task graph.

Traditional scheduling – Framework

Application a weighted DAG $G = (V, E)$

- V = set of tasks (indexed from 1 to n)
- E = dependency relations
- p_i = computational cost of task i (execution time)
- $c(i, j)$ = communication cost (data sent from task i to j)

Platform Set of m processors (identical, uniform, dedicated, ...)

Schedule

- $\sigma(i)$ = date to start the execution of task i
- $\pi(i)$ = processor assigned to it

Traditional scheduling – Constraints

Data dependencies If $(i,j) \in E$ then

$$\text{if } \pi(i) = \pi(j) \text{ then } \sigma(i) + p_i \leq \sigma(j)$$
$$\text{if } \pi(i) \neq \pi(j) \text{ then } \sigma(i) + p_i + c(i,j) \leq \sigma(j)$$

Resource constraints

$$\pi(i) = \pi(j) \Rightarrow [\sigma(i), \sigma(i) + p_i[\bigcap [\sigma(j), \sigma(j) + p_j[= \emptyset$$

Traditional scheduling – Objective functions

Completion time of task i : $C_i = \sigma(i) + p_i$

Makespan or total execution time (the most studied one)

$$C_{max}(\sigma) = \max_{i \in V}(C_i)$$

Other classical objectives : Sum of completion times (minsum)

With arrival times : maximum flow or sum flow

Stretch

Tardiness

Fairness

Single objective problem

Scheduling Problem.

Determine σ : when and where the computational units (tasks) will be executed.

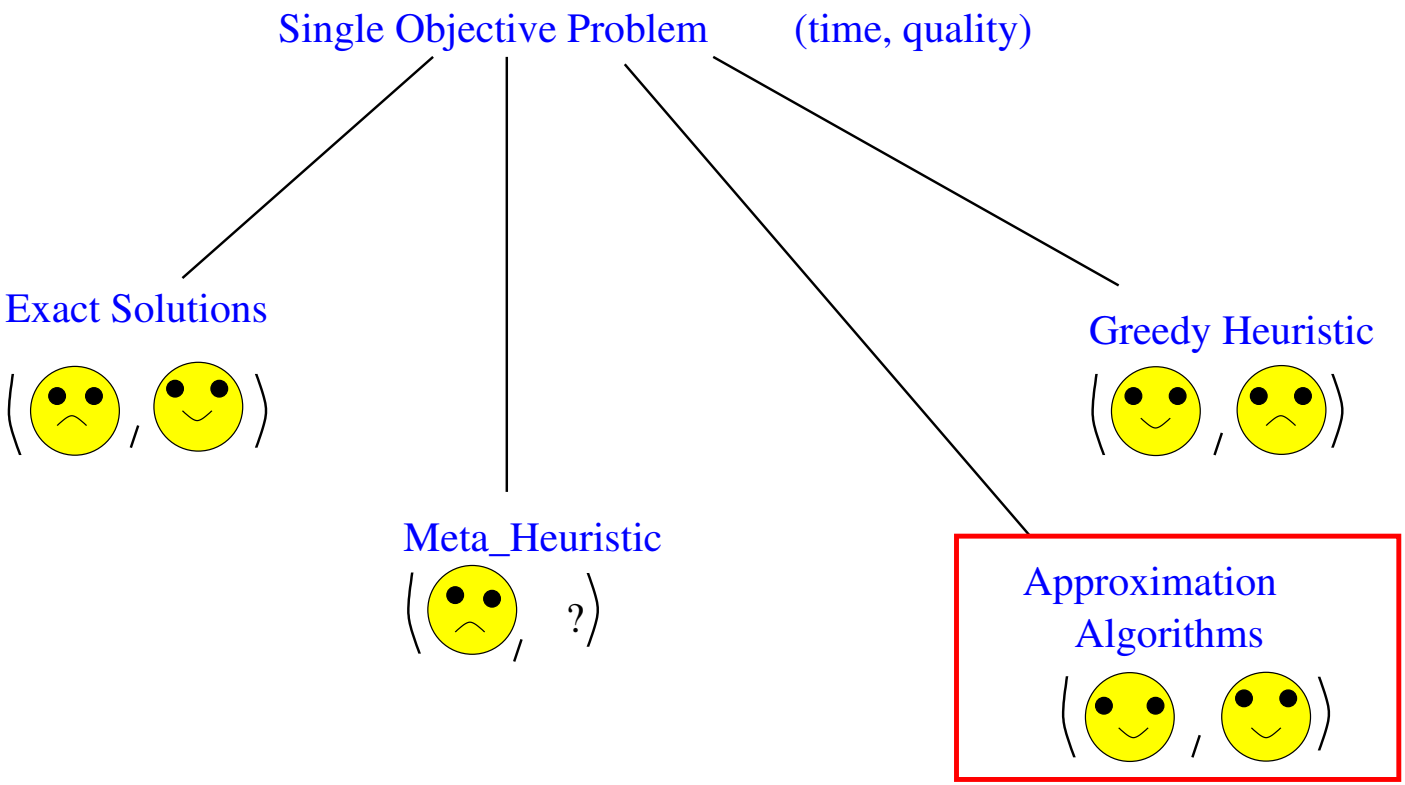
Theorem

Minimizing the makespan (basic problem) is NP-Hard [Ullman75]

Solutions may be obtained by exact methods, purely heuristic methods, or approximation methods.

Solving the single objective problem

Let us recall briefly the various possible ways for solving the problem :



Approximation algorithms

See the lecture of Jean-Claude Konig.
Problems are classified by approximation classes (FPTAS, PTAS, APX).

Well-established techniques since years.
We focus on ρ-approximation algorithms. In particular, dual-approximation [Hochbaum].

Introduction

Today, there is an increasing interest in considering a big variety of criteria. Taking into account the diversity of points of view is a recent challenge.
In the jungle of criteria, the tendency is to imagine his own criteria which would not be considered usually.

Previous works

- C_{max} and tardy jobs [Hoogeven 1995].
- Web access problem (tri criteria) [Papadimitriou in FOCS 2000]. Selection of Web sites for sources of an information. Three objectives : access time, cost and success probability.
- C_{max} versus minsum for Parallel jobs in computational grids.
- Not linear criteria : Reliability or Energy versus performance.
- Fairness [Agnetis 2004].

Philosophical discusssion

Make clear first that that multi-objective problem is an optimization problem, and not really a decision one.

Philosophical discusssion

Make clear first that that multi-objective problem is an optimization problem, and not really a decision one.
The central problem here is the trade-off between all feasible solutions. There are a lot of "good" solutions...

Property

The only reasonable answer is to give **all** the good solutions.
Determining one of them is external to the optimization.

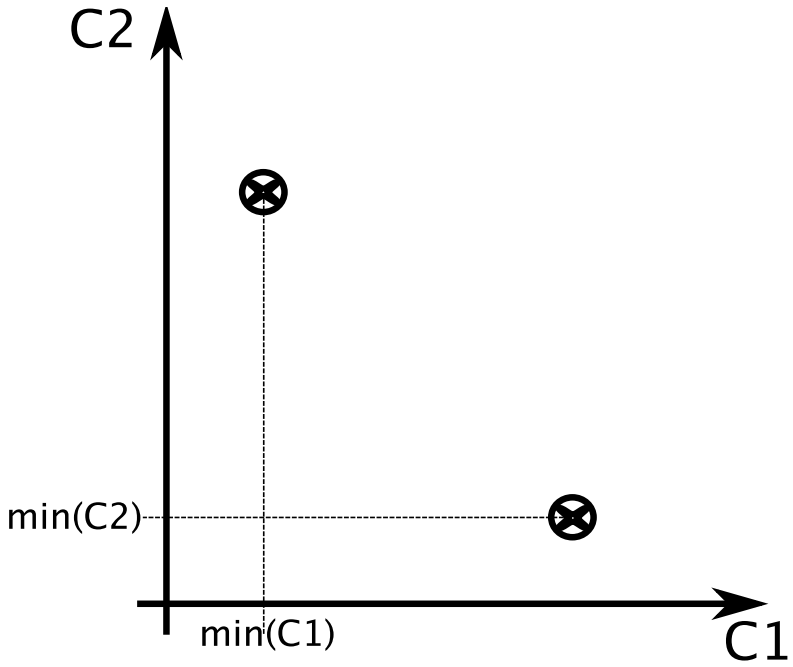
Good solutions are Pareto optimal solutions.

Definition

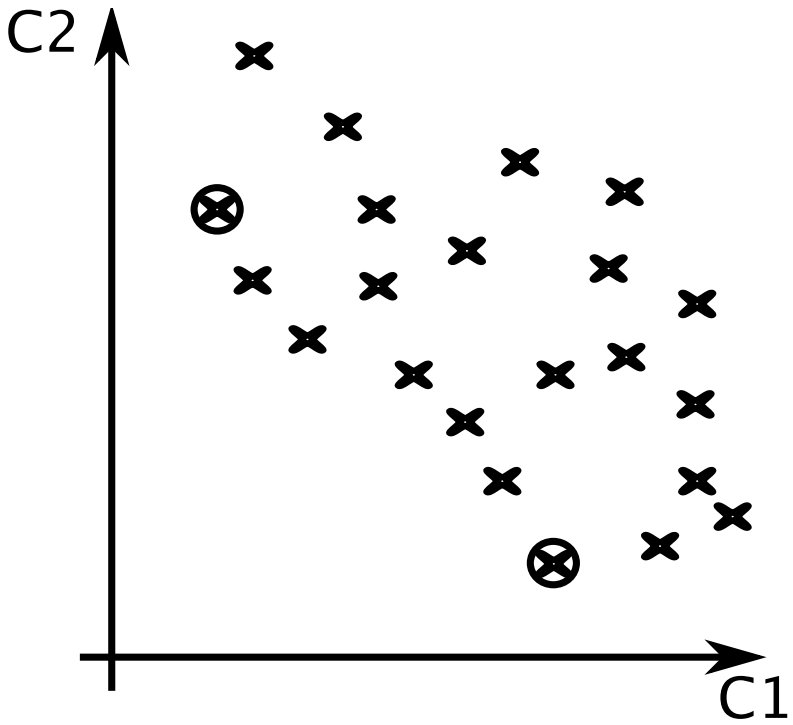
Definition.

A solution is Pareto optimal iif no solution is as good as it is for all the objectives and is better for at least one objective.

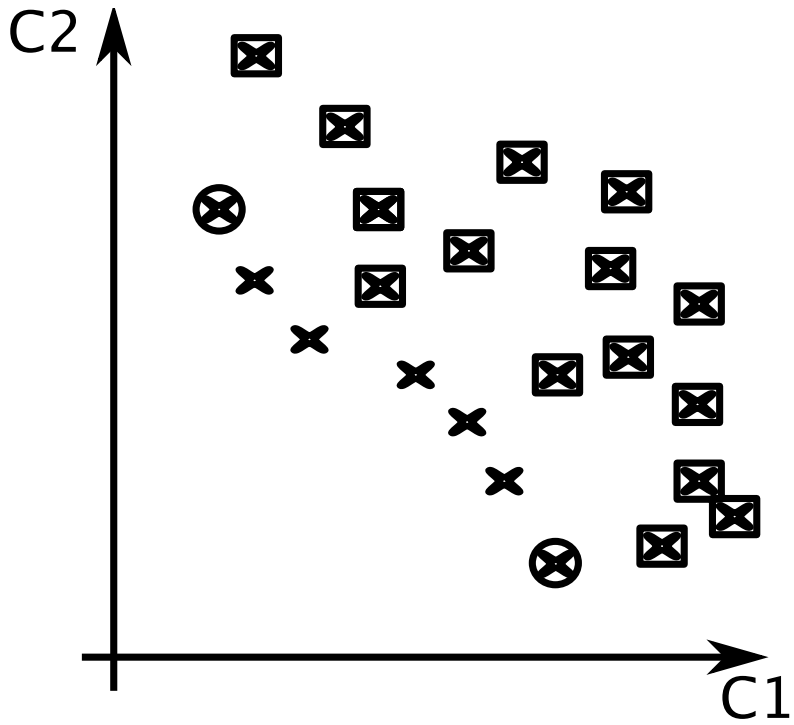
min(C1,C2)



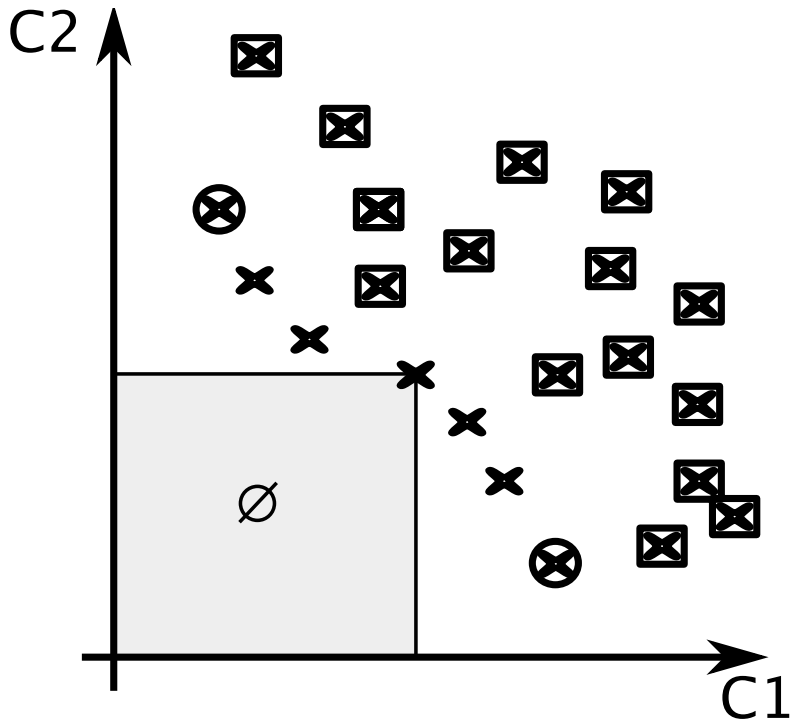
min(C1,C2)



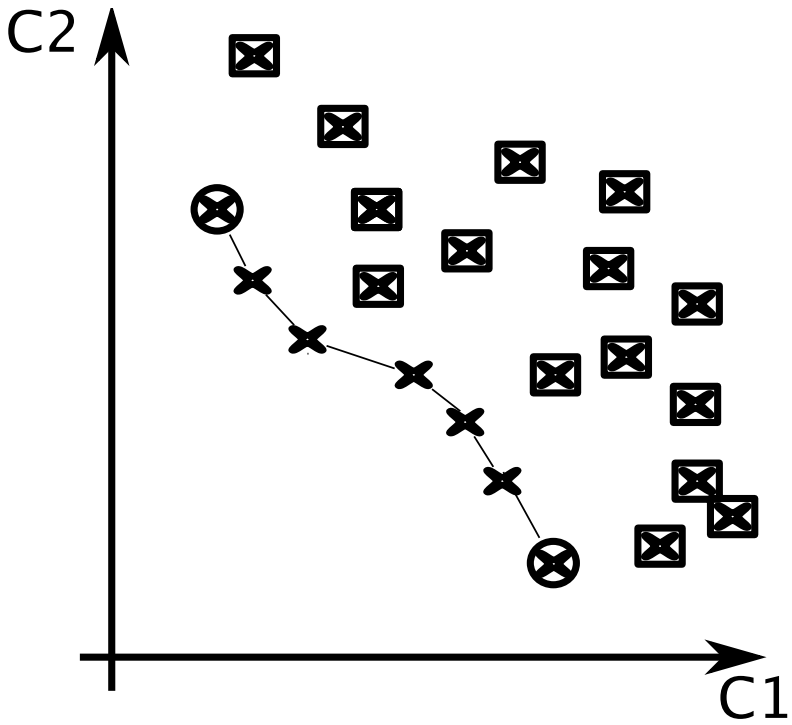
Dominated solutions



Pareto optimal points and curve



Pareto optimal points and curve



Complexity (1)

The theoretical framework on multi-objectives in recent, and thus, the domain lacks of concepts, tools, etc..

Complexity (1)

The theoretical framework on multi-objectives in recent, and thus, the domain lacks of concepts, tools, etc..
For instance, how to define the complexity classes in multi-objective optimization ? Some tentatives like [T'Kindt 2004].

Complexity (1)

The theoretical framework on multi-objectives in recent, and thus, the domain lacks of concepts, tools, etc..
For instance, how to define the complexity classes in multi-objective optimization ? Some tentatives like [T'Kindt 2004].
Similarly for approximation classes.

Complexity (2)

Two easy single objective problems can lead to a hard bi-objective problem :

Complexity (2)

Two easy single objective problems can lead to a hard bi-objective problem :
Let consider two sets of independent jobs J1 and J2 to schedule on 1 machine. The single objective problem is to minimize the minsum over each set Ji ($i = 1, 2$).

Complexity (2)

Two easy single objective problems can lead to a hard bi-objective problem :
Let consider two sets of independent jobs J1 and J2 to schedule on 1 machine. The single objective problem is to minimize the minsum over each set Ji ($i = 1, 2$).
The solution is easy by SPT (Shortest Processing Time) list algorithm.

Complexity (2)

Two easy single objective problems can lead to a hard bi-objective problem :
Let consider two sets of independent jobs J1 and J2 to schedule on 1 machine. The single objective problem is to minimize the minsum over each set Ji ($i = 1, 2$).
The solution is easy by SPT (Shortest Processing Time) list algorithm.
The bi-objective problem is to minimize the minsum on both sets.
The idea of the proof is to show that the decision of the problem is NP-hard (given a pair of values, deciding if there is at least one point which is better than this pair).
Remark that we can easily check if a given solution of the bi-objective problem is optimal (by SPT).

Complexity (3)

What is the size of the optimum of an optimization problem ?

- **Single objective problems :** usually, only one value of the solution
- **multi-objective problems :** exponential number of solutions (or even infinite number of solutions). Sometimes, there are only a few solutions...

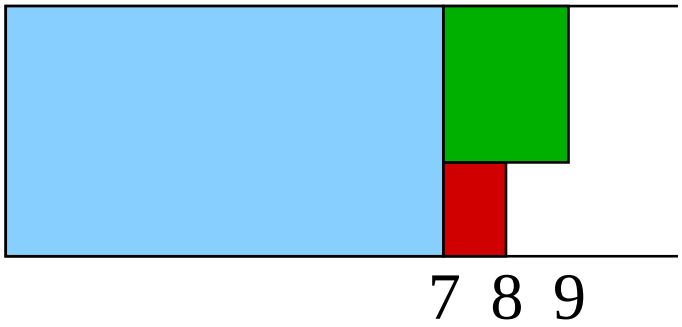
It is linked to the cardinality of the Pareto curve.
For instance, the cardinality of the curve on the previous example is exponential. Remark that even if the cardinal of the curve is polynomial, the problem can be hard...

Overview of the existing approaches

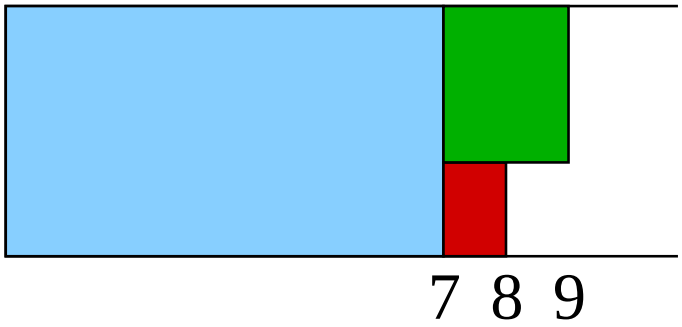
- Just observe.
- Aggregation (linear or convex) of objectives. How to do it ? Remark that we implicitly take position while aggregating. The solution is Pareto optimal but we obtain only those on the convex hull (and not all these points).
- Change some objectives into constraints. Popular in Linear Programming.
- Hierarchy between objectives : give them different priorities
- True multi-objective analysis : find approximations.

Summarize : We focus on the last approach, remark that it is not contradictory with the others.

Preliminary about minsum and Cmax



Preliminary about minsum and Cmax



$$\max_i(C_i) = 9$$

$$\sum_i C_i = 24$$

$$\max_i(C_i) = 9$$

$$\sum_i C_i = 12$$

A first case study : related objectives (1)

Preliminary remark : C_{max} seems to be "easier" than $\sum C_i$ (but it is not).
1 machine scheduling and independent tasks : SPT (Shortest Processing Time) is polynomial for $\sum C_i$ All compact schedules are optimal ones for C_{max}.
For any fixed m, SPT is optimal for $\sum C_i$ (argument : total order on the starting times).
and C_{max} is NP-complete (reduction form Partition).

A first case study : related objectives (2)

Remark : the principle of list algorithms is to select jobs. Thus, it is easy to mix the choices.
The idea here is that there is the same variable behind both objectives. Thus, we can design "good" greedy algorithms in optimizing one objective while controlling the impact on the other. The degradation will be controlled locally.

A first static scheme

Stein and Wein have proposed a nice bi-objective construction by combining two known algorithm (one for each objective).

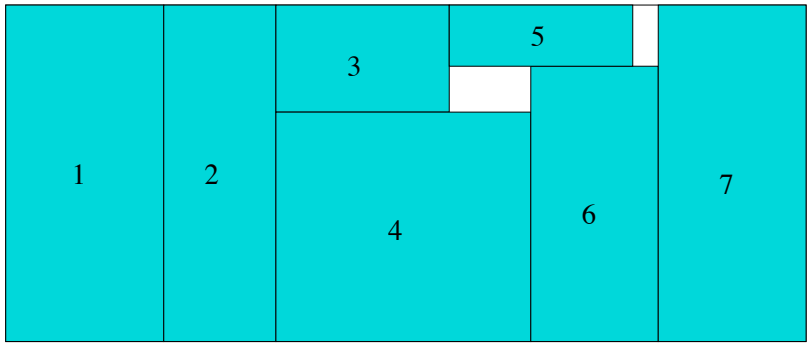
A first static scheme

Stein and Wein have proposed a nice bi-objective construction by combining two known algorithm (one for each objective).
More precisely :

(minsum,makespan) guarantee

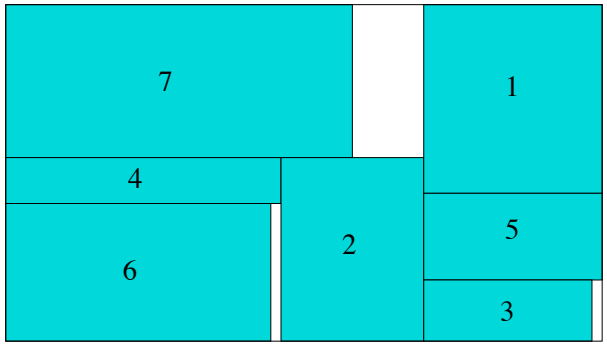
Let r and r' the approximation ratio of scheduling algorithms A and A' on m machines relatively to minsum and C_{max} for parallel moldable tasks.

Stein and Wein's algorithm



Schedule S

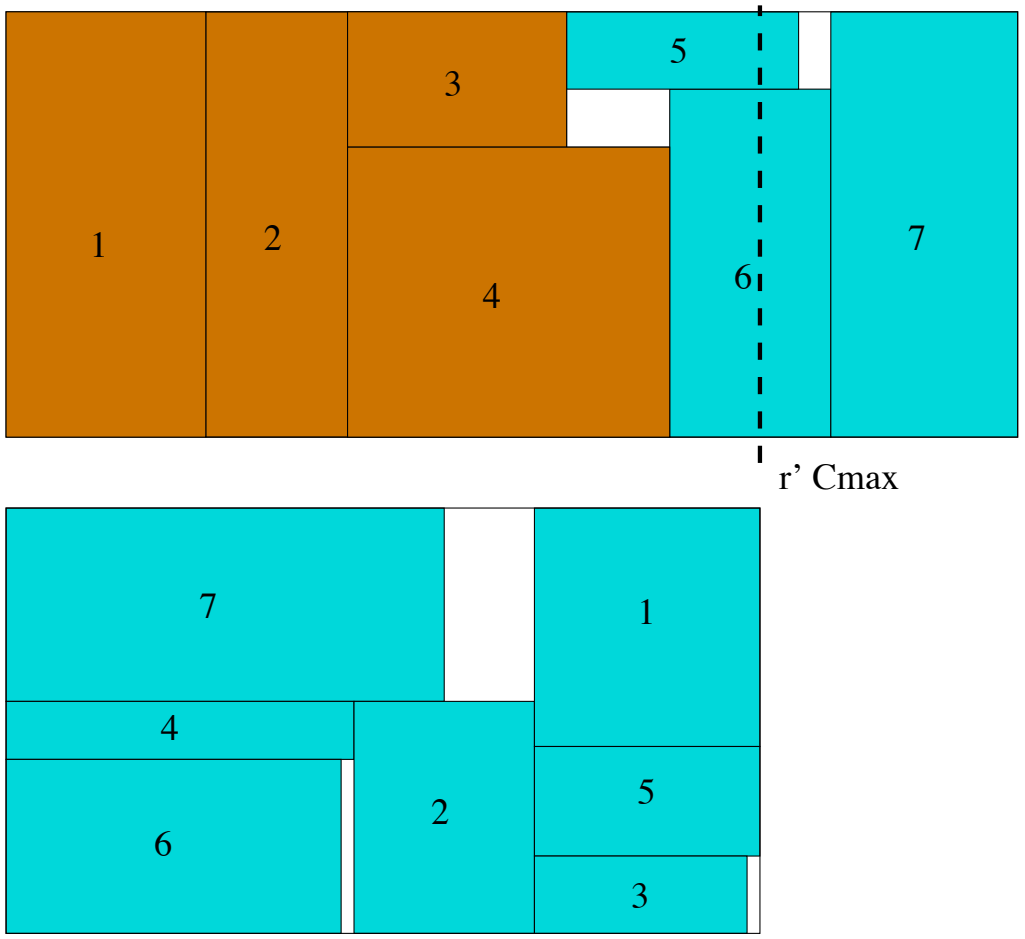
(minsum)



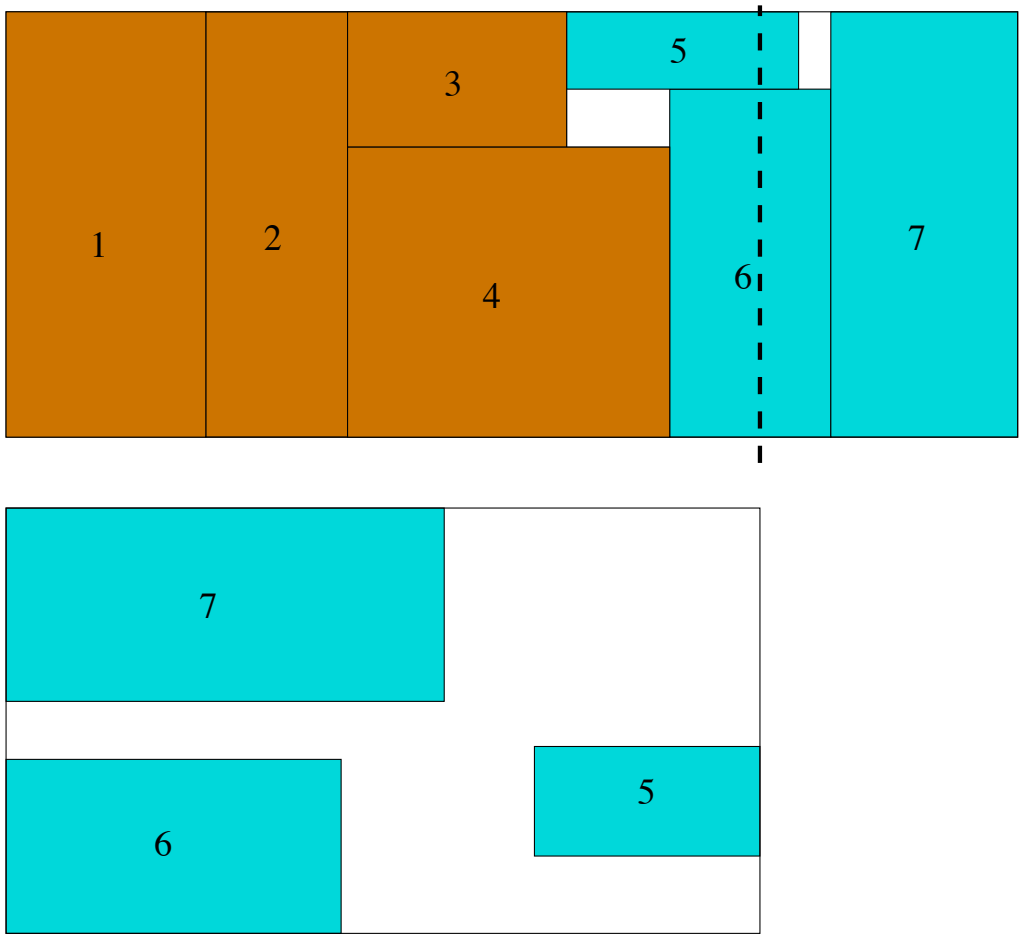
Schedule S'

(Makespan)

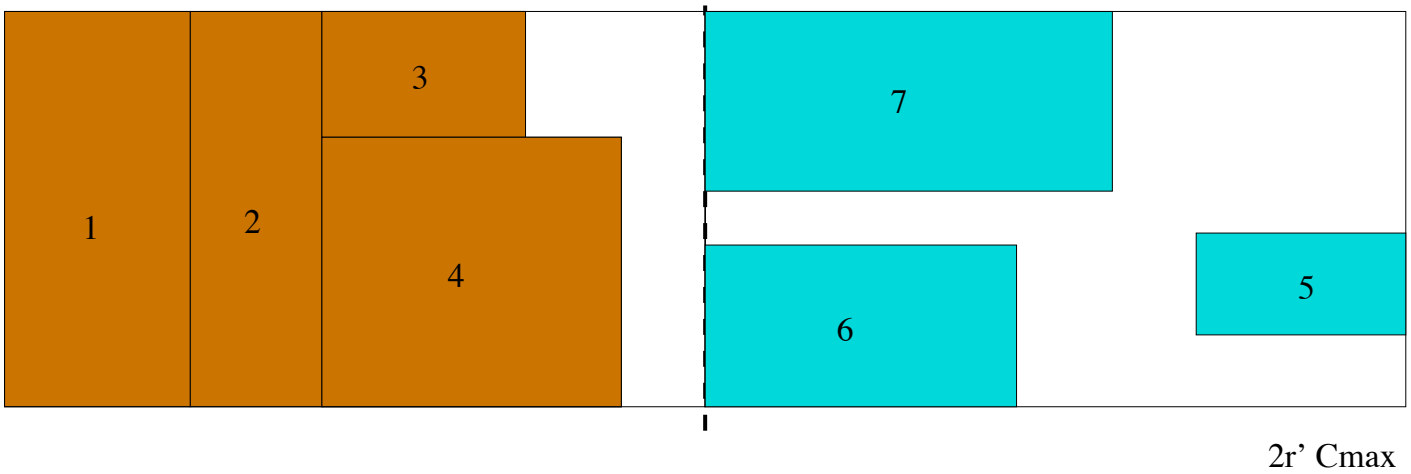
Stein and Wein's algorithm



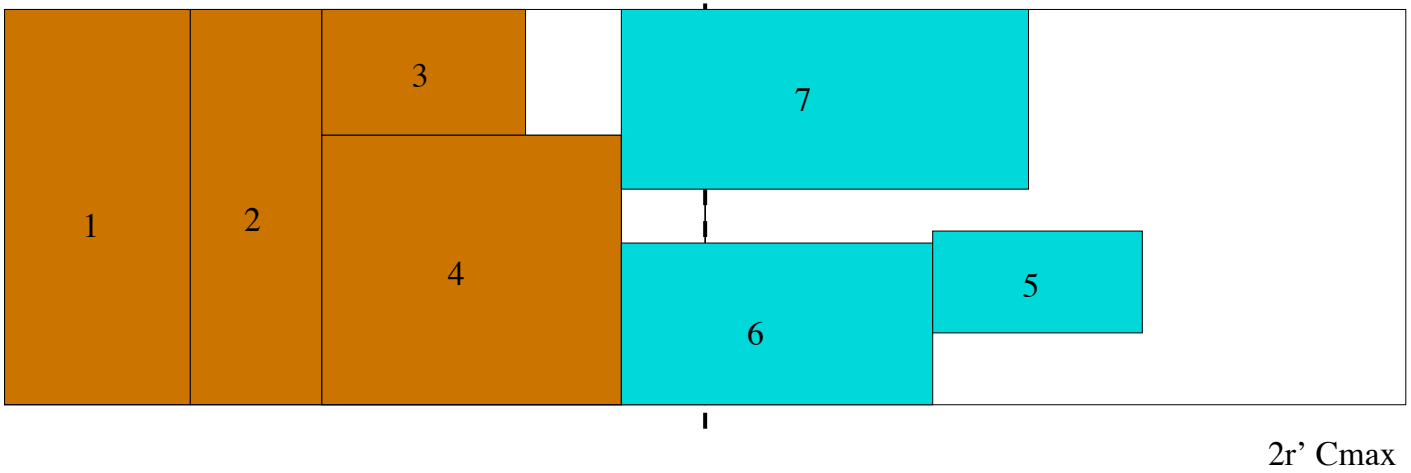
Stein and Wein's algorithm



Stein and Wein's algorithm



Stein and Wein's algorithm



Analysis

Property

The previous algorithm is a $(2r, 2r')$-approximation.

Analysis

Property

The previous algorithm is a $(2r, 2r')$-approximation.

C_{max} is multiplied by a factor 2 (by construction) leading to $2r'$-approximation.

Analysis

Property

The previous algorithm is a $(2r, 2r')$-approximation.

C_{max} is multiplied by a factor 2 (by construction) leading to $2r'$-approximation.
minsum is also no more than twice the minsum of the initial algorithm.

Application

Property.

Let us apply the previous scheme to the best known existing algorithms.
It leads to (8.53)-approximation of minsum and $\frac{3}{2}$-approximation for C_{max}.

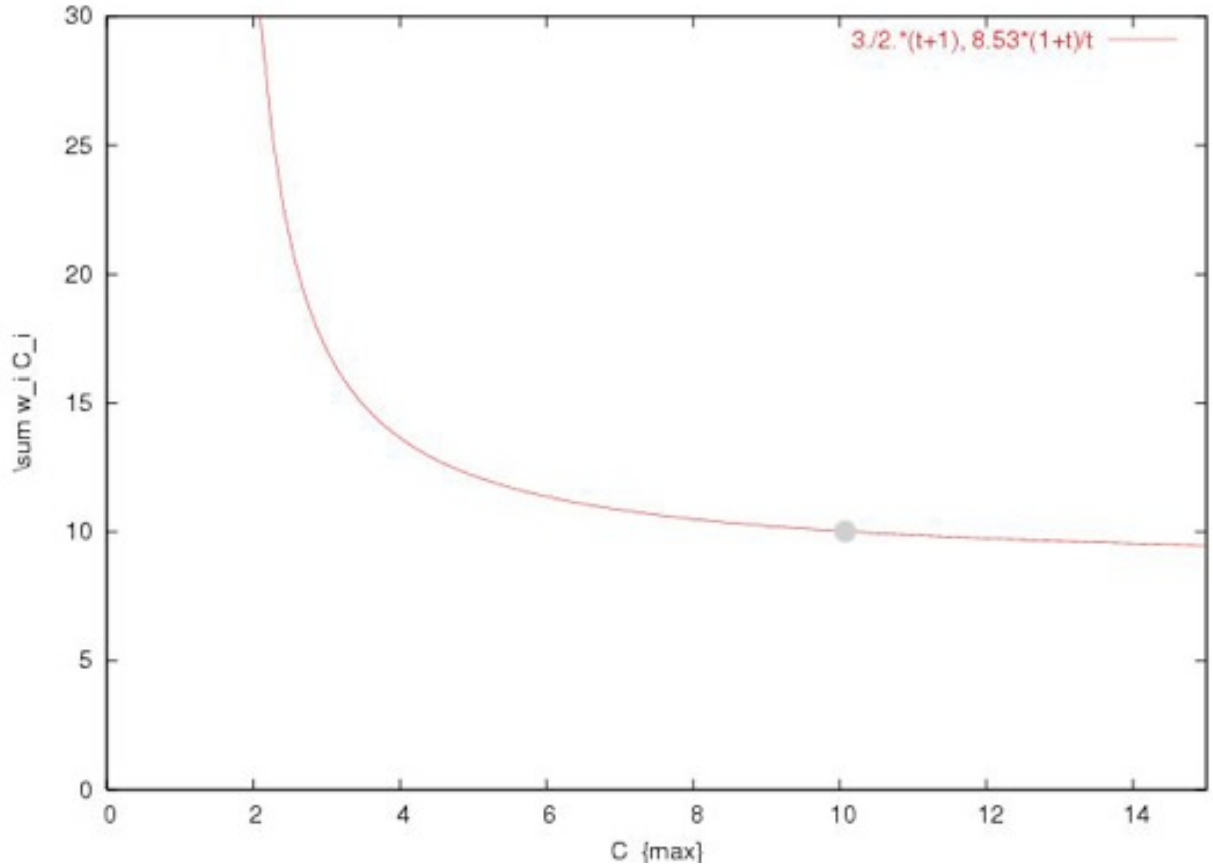
Application

Property.

Let us apply the previous scheme to the best known existing algorithms.
It leads to (8.53)-approximation of minsum and $\frac{3}{2}$-approximation for C_{max}.

We can refine the previous scheme by cutting before $\frac{3}{2}C^*_{max}$: at $t\frac{3}{2}C^*_{max}$ $(t \leq 1)$.

Summary



A dynamic scheme

Problem

We consider :

- independent moldable tasks
- identical processors
- fully connected
- objective function : makespan and minsum

Now, let us describe a dynamic scheme.

Preliminary definition

ρ-MSWP

A ρ-approximation algorithm solving the Maximum Scheduled Weight Problem (*MSWP*) takes as input :

- a set of weighted jobs
- a deadline D

Preliminary definition

ρ-MSWP

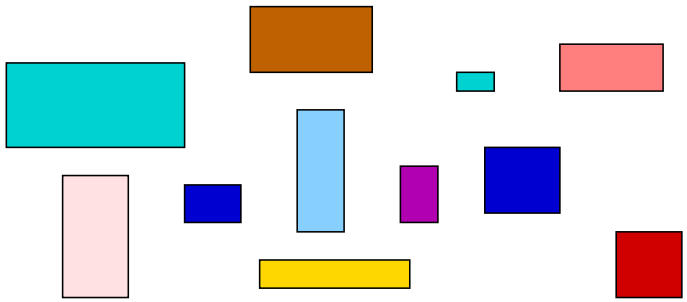
A ρ-approximation algorithm solving the Maximum Scheduled Weight Problem (*MSWP*) takes as input :

- a set of weighted jobs
- a deadline D

Selects some jobs, and produces :

- a schedule of length ρD
- with as much weight as the optimal schedule does in D units of time.

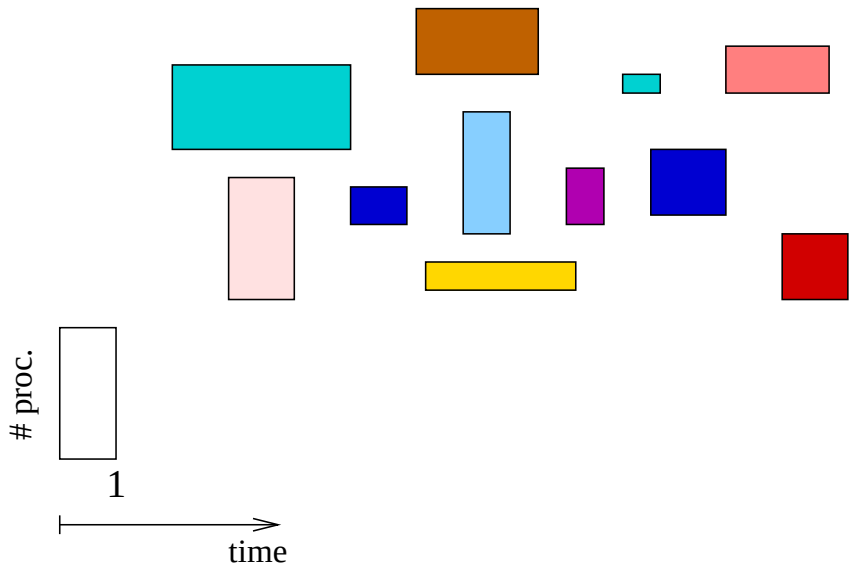
We improved an execution scheme presented by [Hall et al. 96] :



Algorithm

- find the smallest possible execution time t_{min}
- make a box of size $2\rho t_{min}$
- fill the box with as much weight as possible (with a ρ-*MSWP* algorithm)
- double the size of the box and continue

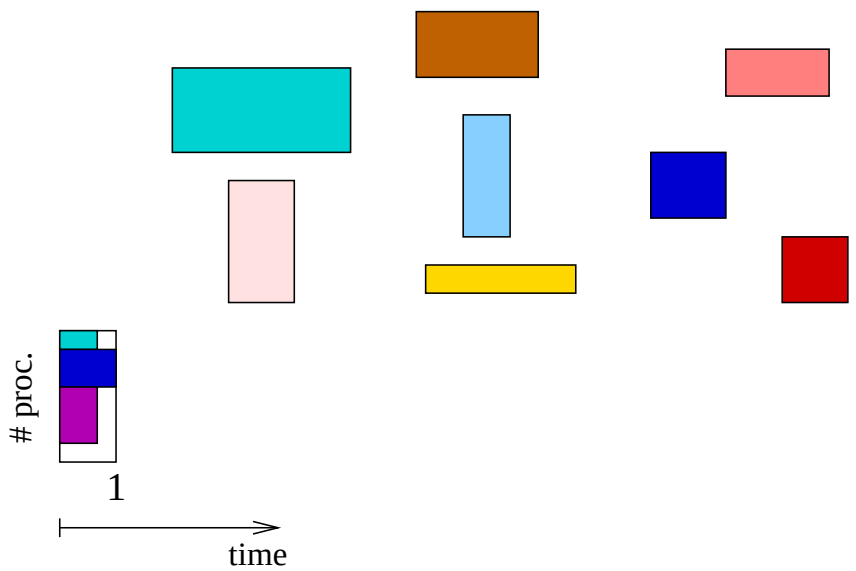
We improved an execution scheme presented by [Hall et al. 96] :



Algorithm

- find the smallest possible execution time t_{min}
- make a box of size $2\rho t_{min}$
- fill the box with as much weight as possible (with a ρ-*MSWP* algorithm)
- double the size of the box and continue

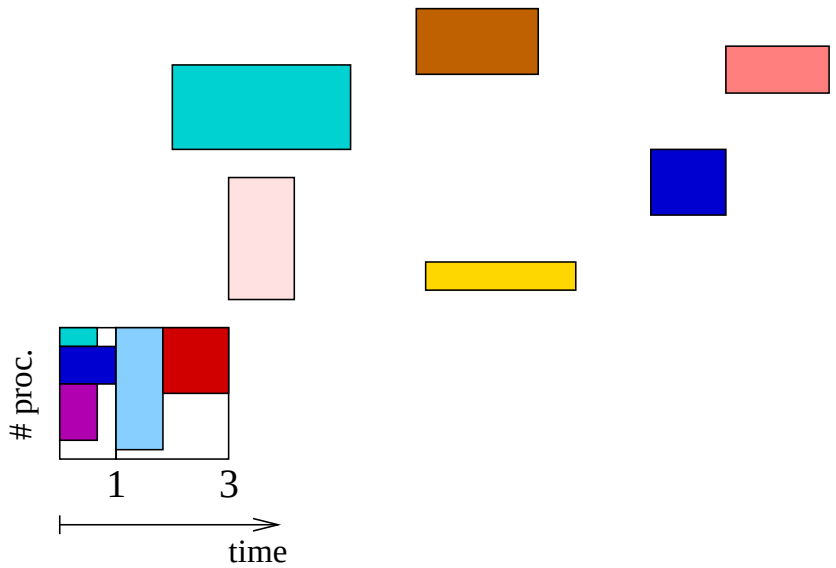
We improved an execution scheme presented by [Hall et al. 96] :



Algorithm

- find the smallest possible execution time t_{min}
- make a box of size $2\rho t_{min}$
- fill the box with as much weight as possible (with a ρ-*MSWP* algorithm)
- double the size of the box and continue

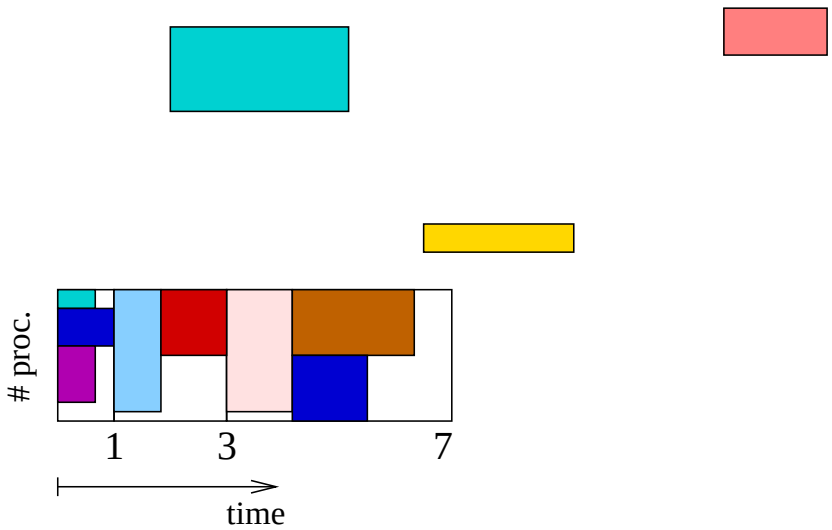
We improved an execution scheme presented by [Hall et al. 96] :



Algorithm

- find the smallest possible execution time t_{min}
- make a box of size $2\rho t_{min}$
- fill the box with as much weight as possible (with a ρ-*MSWP* algorithm)
- double the size of the box and continue

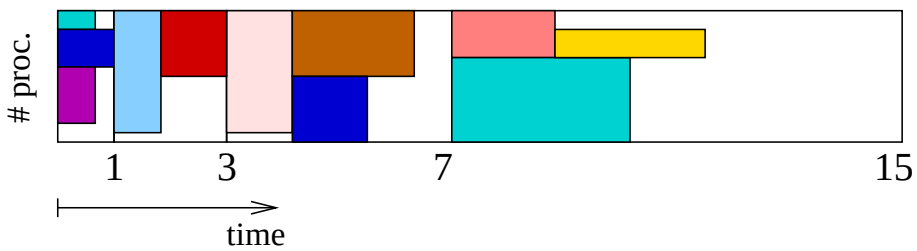
We improved an execution scheme presented by [Hall et al. 96] :



Algorithm

- find the smallest possible execution time t_{min}
- make a box of size $2\rho t_{min}$
- fill the box with as much weight as possible (with a ρ-*MSWP* algorithm)
- double the size of the box and continue

We improved an execution scheme presented by [Hall et al. 96] :



Algorithm

- find the smallest possible execution time t_{min}
- make a box of size $2\rho t_{min}$
- fill the box with as much weight as possible (with a ρ-*MSWP* algorithm)
- double the size of the box and continue

Improvements

1. off-line
2. better ρ-*MSWP* algorithm
3. parameter α

(makespan ;minsum) guaranty

$$\left(\frac{\alpha}{\alpha-1}\rho; \frac{\alpha^2}{\alpha-1}\rho\right)$$

This scheme can be used in several cases, depending on the underlying ρ-*MSWP* algorithm :

- rigid parallel tasks
- moldable tasks
- hierarchical moldable tasks

We may also use it in an on-line setting

Setting the problem

Let us consider several users of a computational grid.
Each one has his-her own criterion :

- minimum completion time
- 10 percents of the jobs completed as soon as possible
- all results no later than 2 days
- etc..

Informal discussion

The Pareto optimal solutions are not all equivalent for the agents. This is not our problem and we have no tool for comparing the pairs (Cmax, minsum).

Informal discussion

The Pareto optimal solutions are not all equivalent for the agents.
This is not our problem and we have no tool for comparing the pairs (Cmax, minsum).
We restrict the presentation to bi-objective problems.
Each objective is associated to a "agent" (a player who can not take any decision).

Setting of the problem

We need a "tool" for comparing the solutions between them : this is the **Utility** function.
It reflects a preference relation for ordering the solutions for each agent.

Setting of the problem

We need a "tool" for comparing the solutions between them : this is the **Utility** function.
It reflects a preference relation for ordering the solutions for each agent.
Let $\succeq_i$ denote the reference relation between solutions (pairs of values of objectives) for i.

Definition.

Utility for i (or playoff) u_i $(i = 1, 2)$ is an application which verifies : $u_i(a) \geq u_i(b)$ iif $a \succeq_i b$

Fair solution

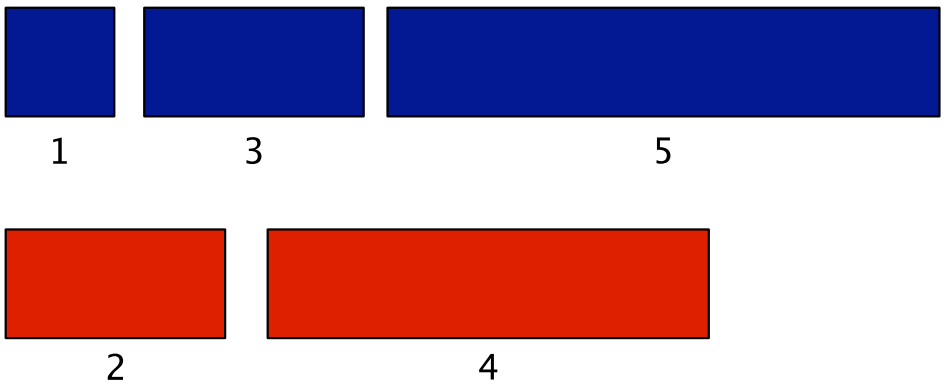
Is the notion of utility enough ?
Hard for the previous (C_{max},minsum) problem. It is possible to order all local solutions, but the ordering is only ordinal and not cardinal.

Fair solution

Is the notion of utility enough ?
Hard for the previous (C_{max},minsum) problem. It is possible to order all local solutions, but the ordering is only ordinal and not cardinal.
We restrict the analysis to optimization problems with the same objective applied on several (here 2) sub-sets (see Agnetis for more details).
Fair solutions are defined by a principle of transfer between Pareto optimal solutions.

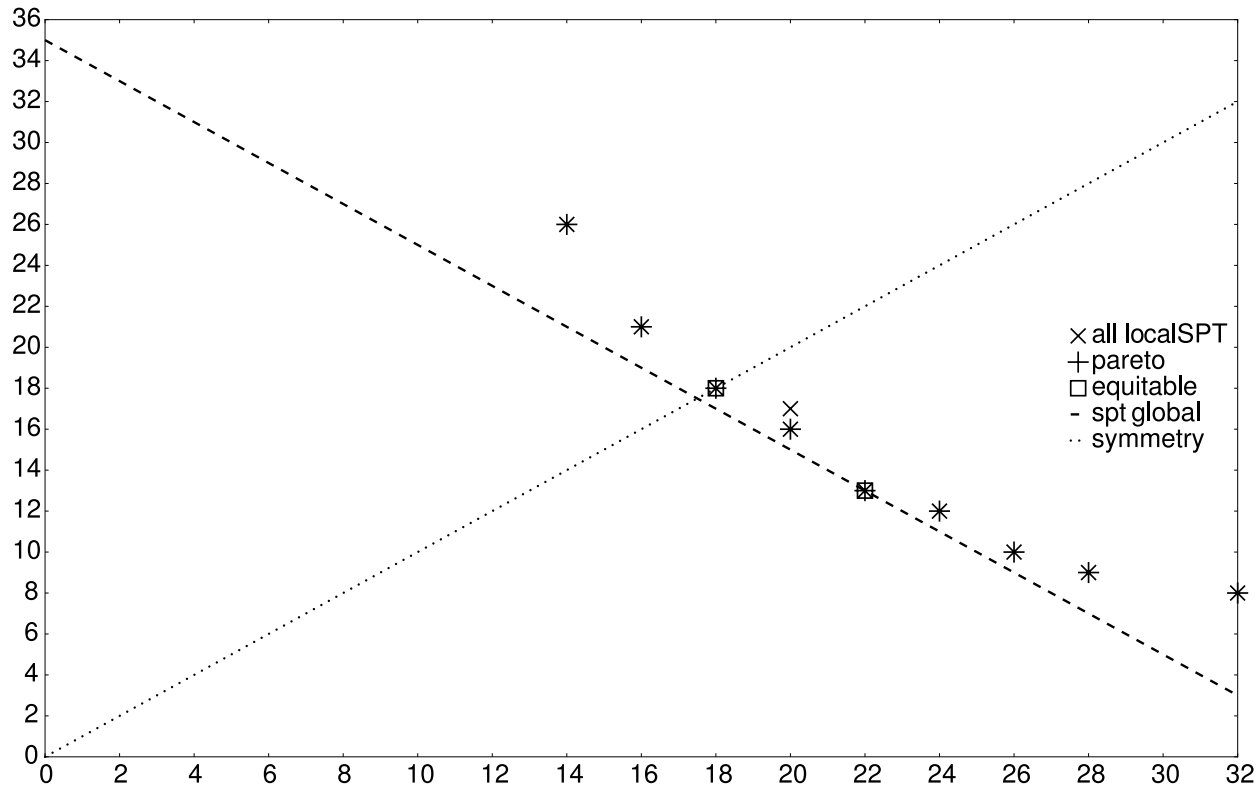
Example (Agnetis)

2 agents (blue and red) aiming at minimizing minsum on their own subset of jobs.

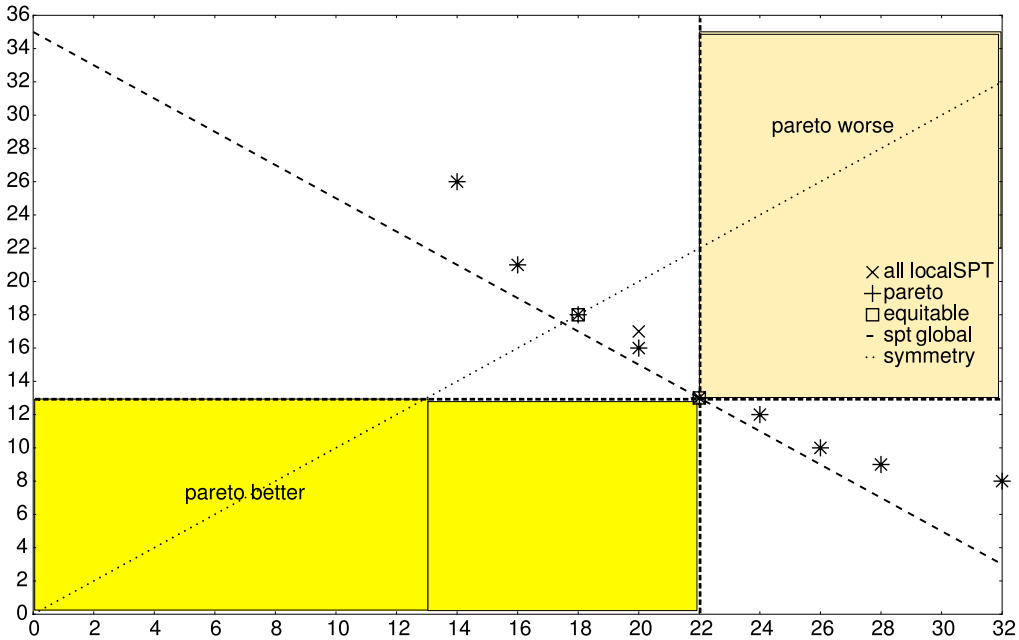


Example (Agnetis)

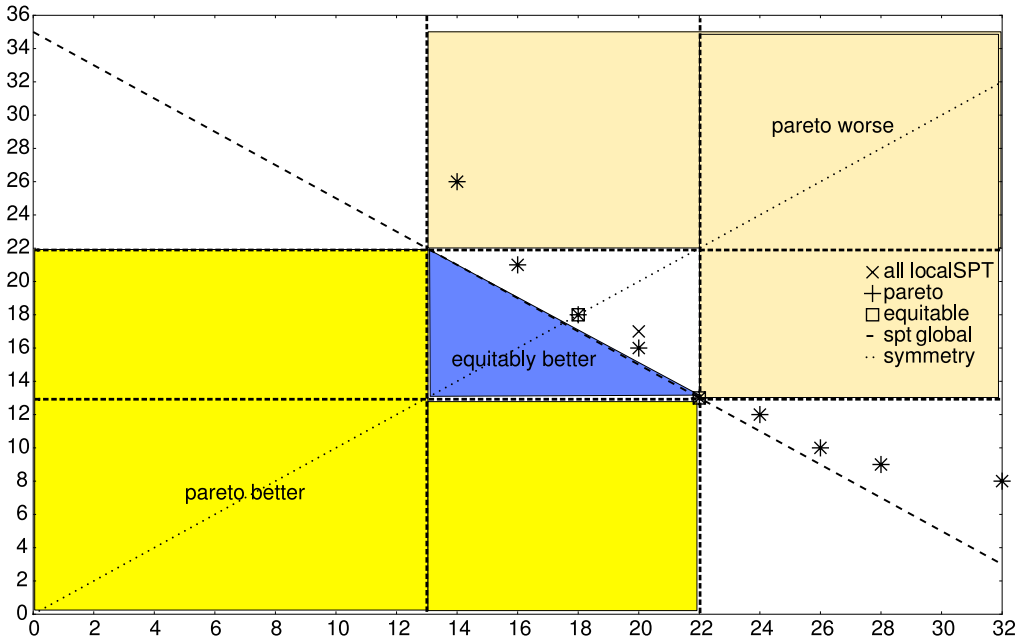
2 agents (blue and red) aiming at minimizing minsum on their own subset of jobs.



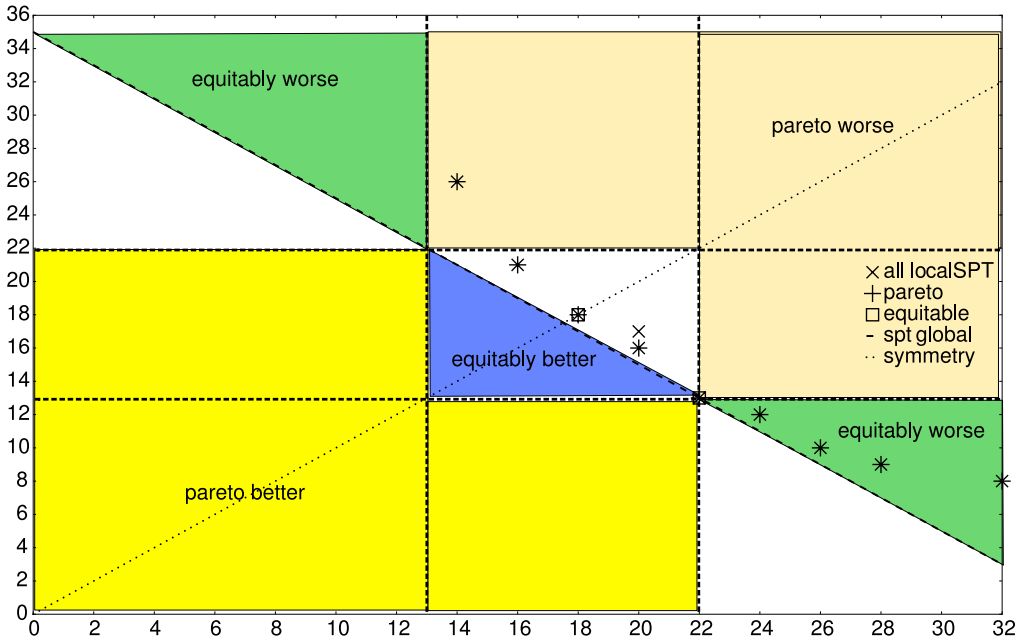
Example (Analysis)



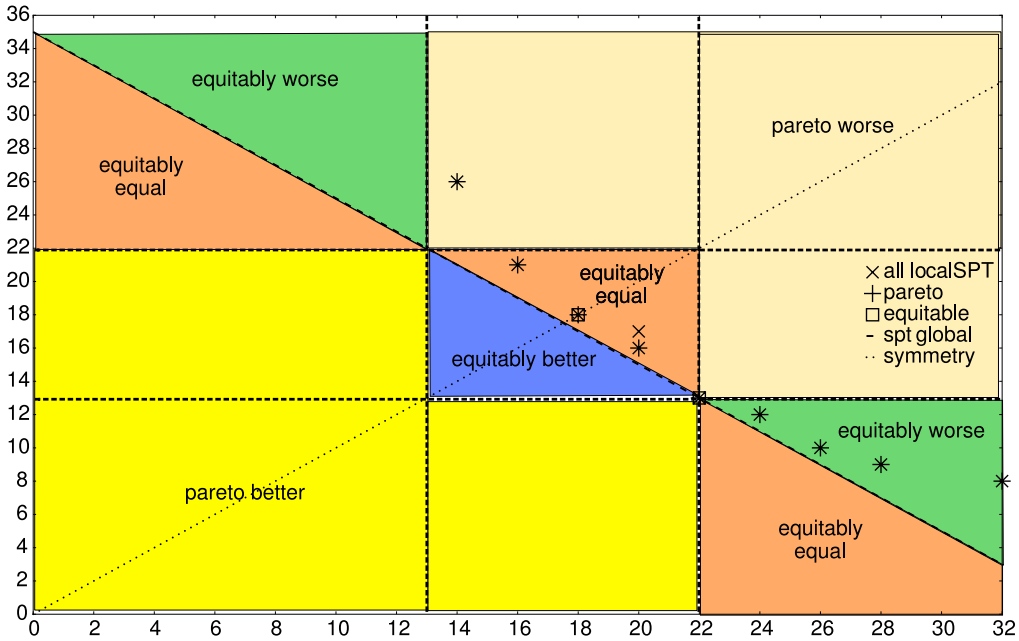
Example (Analysis)



Example (Analysis)



Example (Analysis)



General Conclusion

Increasing interest for the multi-objective analysis.
Much harder problem which still lacks of tools.

General Conclusion

Increasing interest for the multi-objective analysis.
Much harder problem which still lacks of tools.
After the break, let us present another view on the fairness : Cake division.