The Geometry of Truthfulness

Angelina Vidali

Max Planck Institut für Informatik, Saarbrücken

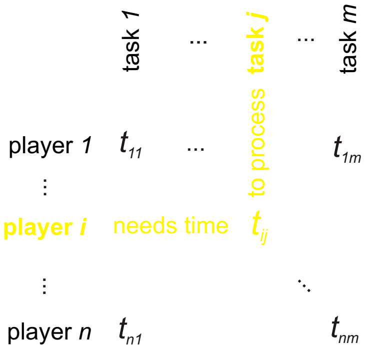
16th December 2009

Scheduling unrelated machines

[Nisan and Ronen STOC '99]

The matrix of processing times

We want to process m tasks using n machines(/selfish players).
We have the following matrix of processing times:

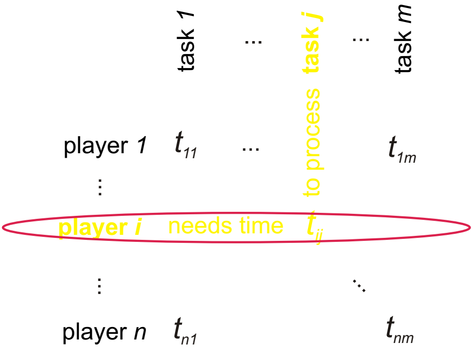


Scheduling unrelated machines

[Nisan and Ronen STOC '99]

The matrix of processing times

We want to process m tasks using n machines(/selfish players).
We have the following matrix of processing times:



Only player i knows
the values of his line.
He can report a false value!

Scheduling unrelated machines

Protocol

- The players declare their values
- The mechanism allocates all tasks (allocation algorithm)
- The mechanism pays the players based on the declared values and the allocation (payment algorithm)

The objective of each player: utility maximization

maximize{payment−processing time}

MinMax objective of the mechanism designer:

Finish with all tasks as soon as possible! i.e., minimize the makespan

makespan=the time of the player that finishes last

Theorem (Nisan and Ronen, STOC '99)

No truthful mechanism can achieve this objective.

They conjectured the best possible approximation ratio is n. They showed it cannot be better than 2.

Now the lower bound is 2.618 [Koutsoupias–Vidali MFCS '07]

Brief history of scheduling unrelated machines

Restricted to the objective of minimizing the makespan

- It is a well-studied NP-hard problem. Lenstra, Shmoys, and Tardos showed that its approximation ratio is between $3/2$ and 2.
- Nisan and Ronen in 1998 initiated the study of its mechanism-design version.
  - They gave a mechanism with approximation ratio n.
  - They showed a lower bound of 2.
  - They conjectured that the right answer is n.
  - They also gave a randomized mechanism with approximation ratio $7/4$ for 2 players.
- Archer and Tardos considered the related machines problem. In this case, for each machine there is a single value (instead of a vector), its speed.

max planck institut informatik

Recent results [1]

Deterministic

- The lower bound was improved from 2 to 2.41
(Christodoulou - Koutsoupias - Vidali, SODA 2007)
- . . . and then to 2.61 for many machines
(Koutsoupias - Vidali, MFCS 2007)
- For 2 machines the only truthful mechanisms with bounded approximation ratio are task-independent.
(Dobzinski - Sundararajan, EC 2008)
- For 2 machines the only decisive truthful mechanisms are either affine minimizers or threshold mechanisms.
(Christodoulou - Koutsoupias - Vidali, ESA 2008)

Recent Results [2]

Fractional

- The approximation ratio is between $2 - 1/n$ and $(n+1)/2$ (Christodoulou - Koutsoupias - Kovacs, ICALP 2007)

Randomized

- The approximation ratio is between $2 - 1/n$ and $7/8\, n$ (Mu'alem and Schapira, SODA 2007).
- A 1.59-approximation mechanism for 2 machines [improving on $7/4 = 1.75$ Nisan-Ronen] (Lu - Yu, STACS 2008, WINE 2008)

Discrete (only two values: high and low)

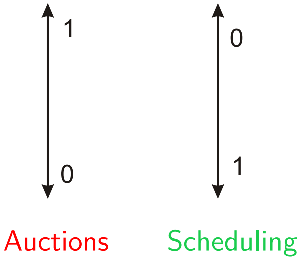
Mechanism with approximation ratio 2 (Lavi - Swamy EC 2007).

Anonymous

A lower bound of n. (Ashlagi, Dobizinski and Lavi EC 2009)

Auctions or Scheduling?

The world upside down



- Auction: sell the objects to bidder who values them high
- Scheduling: allocate the task to machines with small processing times
  Change max→ min
- The only difference is the objective but here we want to find all truthful mechanisms regardless of objective.

An auction for selling multiple items [The input]

Auction (A bidder gets an item if his valuation is high enough)

Possible Outcomes:	{only }	{only }	{both  }
valuation of player 1:	10	6	10+6
valuation of player 2:	3	5	3+5
valuation of player 3:	2	9	2+9

Scheduling (A machine gets a job if its processing time is low)

Possible Outcomes:	{only task 1}	{only task 2}	{both tasks}
Valuation of player 1:	10	6	10+6
valuation of player 2:	3	5	3+5
valuation of player 3:	2	9	2+9

- If the valuations are additive (or linear) we don't need the last column, we can compute it if we know the first two columns.

max planck institut informatik

Truthful mechanisms

"A player has nothing to gain by lying."

Definition (Truthful mechanisms)

A mechanism is truthful(/incentive compatible) if revealing the true values is dominant strategy of each player.

i.e. if one player lies (and the other players stick to their values) he cannot increase his utility

Truthful = Monotone

Definition (Monotonicity Property)

An allocation algorithm is monotone if for every two inputs t and t' which differ only on machine i (i.e., on the i-the row) the associated allocations a and a' satisfy

$$(a_i - a_i') \cdot (t_i - t_i') \leq 0,$$

where $\cdot$ denotes the dot product of the vectors.

Theorem (Saks, Lan Yu EC 2005)

The Monotonicity Property is a necessary and sufficient condition (without any reference to payments!) for truthfulness.

The monotonicity property

Take

$$(input, output) = \left(\begin{pmatrix} t_{11} & t_{12} & \cdots & t_{1m} \\ \cdots & & & \\ \mathbf{t_{i1}} & \mathbf{t_{i2}} & \cdots & \mathbf{t_{im}} \\ \cdots & & & \\ t_{n1} & t_{n2} & \cdots & t_{nm} \end{pmatrix}, \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ \cdots & & & \\ \mathbf{a_{i1}} & \mathbf{a_{i2}} & \cdots & \mathbf{a_{im}} \\ \cdots & & & \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} \right)$$

the *row vectors of player i* satisfy the following inequality

$$(\mathbf{a_i} - \mathbf{a_i'}) \cdot (\mathbf{t_i} - \mathbf{t_i'}) \leq 0.$$

- The other rows do not have to satisfy any condition
- We don't need to care about the payments.

The monotonicity property

Take

$$(\textit{input}', \textit{output}') = \left(\begin{pmatrix} t_{11} & t_{12} & \cdots & t_{1m} \\ \cdots & & & \\ \mathbf{t'_{i1}} & \mathbf{t'_{i2}} & \cdots & \mathbf{t'_{im}} \\ \cdots & & & \\ t_{n1} & t_{n2} & \cdots & t_{nm} \end{pmatrix}, \begin{pmatrix} a'_{11} & a'_{12} & \cdots & a'_{1m} \\ \cdots & & & \\ \mathbf{a'_{i1}} & \mathbf{a'_{i2}} & \cdots & \mathbf{a'_{im}} \\ \cdots & & & \\ a'_{n1} & a'_{n2} & \cdots & a'_{nm} \end{pmatrix} \right)$$

the *row vectors of player i* satisfy the following inequality

$$(\mathbf{a_i} - \mathbf{a'_i}) \cdot (\mathbf{t_i} - \mathbf{t'_i}) \leq 0.$$

- The other rows do not have to satisfy any condition
- We don't need to care about the payments.

To grasp truthfulness we can assume w.l.o.g. a single-player model

Our results apply directly to multi-player situations

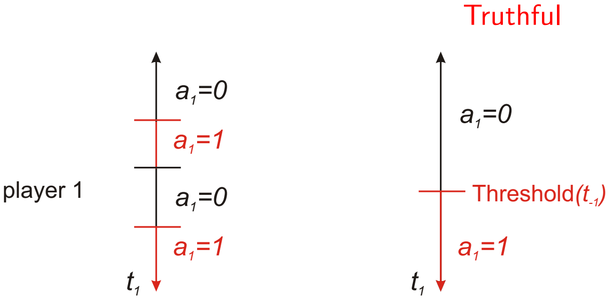
- "If the values of all other players are fixed, is it better to report my true values or is it better to lie?"
- Studying the mechanism for fixed values of the other players is like studying a single-player model.
- Truthfulness determines the geometry of the projections of the mechanism for fixed values of the all players except for one.

max planck institut informatik

A Geometrical Interpretation of Monotonicity

Singe-task case

suppose there is only one task. Fix t_{-1} (i.e. the values of all players except for player 1).



A truthful allocation should be decreasing with respect to the processing time of player 1.

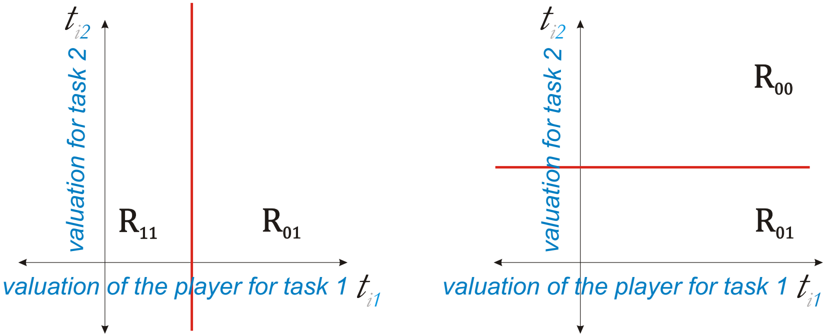
A Geometrical Interpretation of Monotonicity

Monotonicity determines the slope of the separating hyperplane between each pair of regions.

$(a_{i1} - a'_{i1})t_{i1} + (a_{i1} - a'_{i2})t_{i2} = f_{a:a'}(t_{-i})$ separates $R_a, R_{a'}$

$t_{i1} = f_{11:01}(t_{-i})$ separates R_{11} and R_{01}

Allocations in *Hd*=1



R_{01} := the region where player i has assignment 01

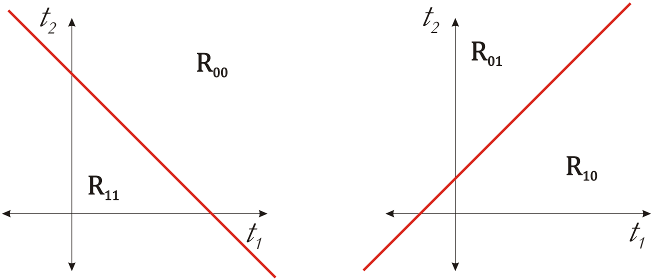
A Geometrical Interpretation of Monotonicity

Monotonicity determines the slope of the separating hyperplane between each pair of regions.

$(a_{i1} - a'_{i1})t_{i1} + (a_{i1} - a'_{i2})t_{i2} = f_{a:a'}(t_{-i})$ separates $R_a, R_{a'}$

$t_{i1} + t_{i2} = f_{11:00}(t_{-i})$ separates R_{11} and R_{00}

Allocations in *Hd*=2

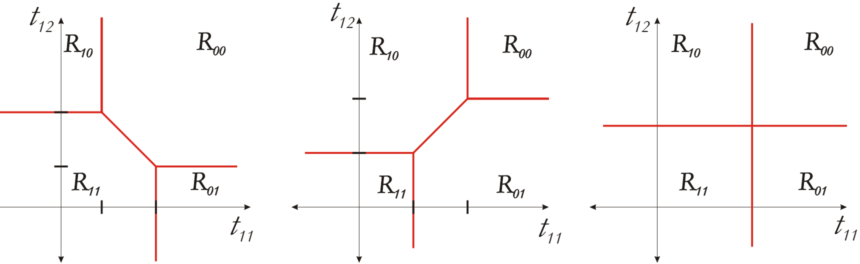


R_{01} := the region where player i has assignment 01

A Geometrical Interpretation of Monotonicity

And which are the possible shapes for the case of two tasks?

Fix t_{-1} (i.e. the values of all players except for player 1),
$R_{10} :=$ the region where player 1 has assignment 10



$(a_1 - a'_1)t_{11} + (a_2 - a'_2)t_{12} = f_{a:a'}(t_{-1})$ separates $R_a, R_{a'}$

$t_{11} + t_{12} = f_{11:00}(t_{-1})$ separates R_{11} and R_{00}.

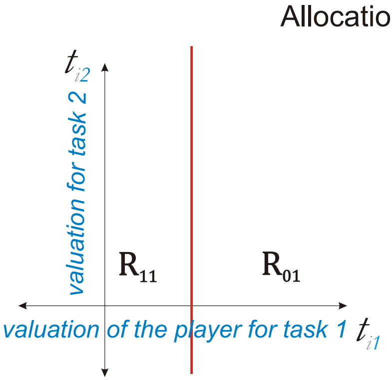
Monotonicity determines the slope of the separating hyperplane.

A Geometrical Interpretation of Monotonicity

Monotonicity determines the slope of the separating hyperplane between each pair of regions.

$(a_1 - a'_1)t_1 + (a_2 - a'_2)t_2 = f_{a:a'}$ separates $R_a, R_{a'}$

$t_1 = f_{11:01}$ separates R_{11} and R_{01}



R_{01} := the region where the player has assignment 01

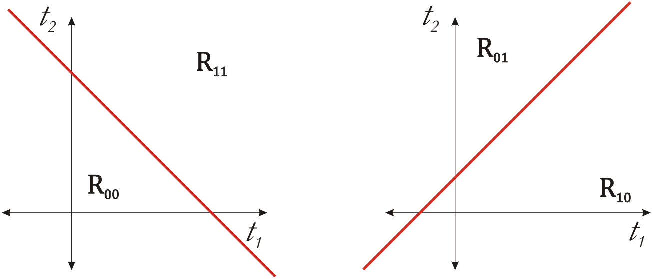
A Geometrical Interpretation of Monotonicity

Monotonicity determines the slope of the separating hyperplane between each pair of regions.

$(a_1 - a'_1)t_1 + (a_2 - a'_2)t_2 = f_{a:a'}$ separates $R_a, R_{a'}$

$t_1 + t_2 = f_{11:00}$ separates R_{11} and R_{00}

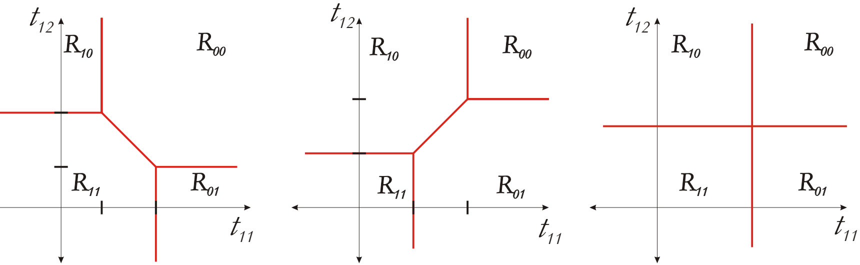
Allocations in *Hd*=2



R_{01} := the region where the player has assignment 01

And which are the possible shapes?

The case of two items

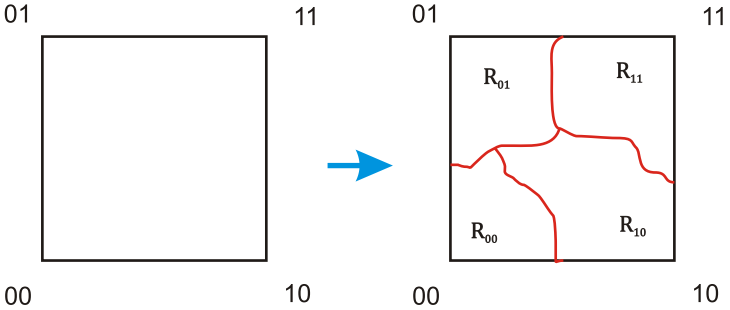


What if there are more tasks??? $m \geq 3$

Alternative statement of the problem

Forgetting everything about game theory!

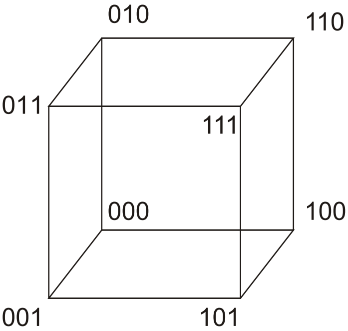
Given the cube $[0,1]^m$ partition it to regions having the labels of the cube vertices so that each pair of points and their corresponding labels satisfy the monotonicity property.



Alternative statement of the problem

Forgetting everything about game theory!

Given the cube $[0,1]^m$ partition it to regions having the labels of the cube vertices so that each pair of points and their corresponding labels satisfy the monotonicity property.

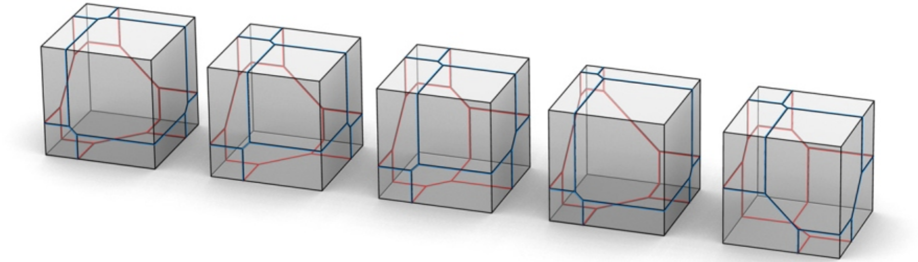


The Geometry of Truthfulness

The case of 3 tasks

Theorem

The truthful mechanisms for the case of three tasks are the five following shapes & all their rotations:

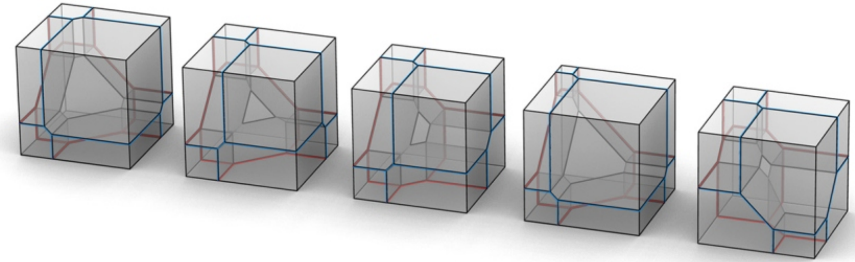


The Geometry of Truthfulness

The case of 3 tasks

Theorem

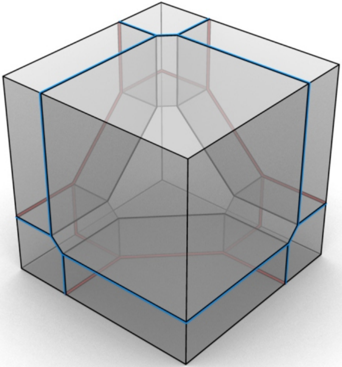
The truthful mechanisms for the case of three tasks are the five following shapes & all their rotations:



The Geometry of Truthfulness

The case of 3 tasks

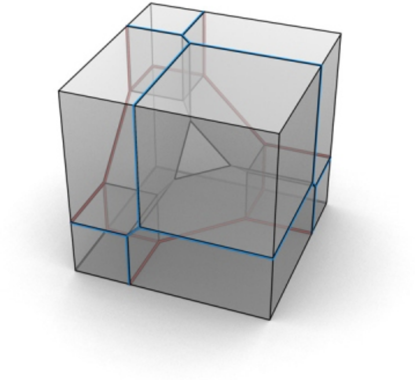
5 possible shapes & all their rotations



The Geometry of Truthfulness

The case of 3 tasks

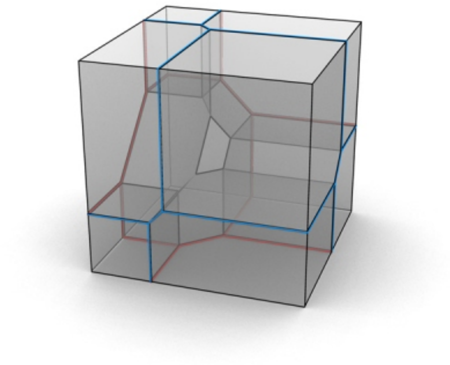
5 possible shapes & all their rotations

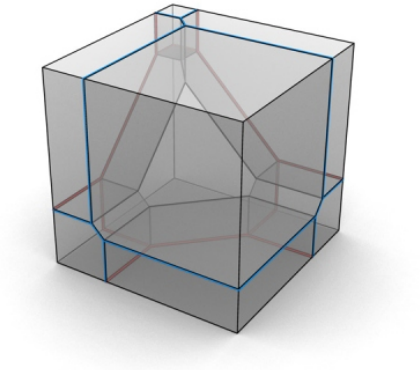


The Geometry of Truthfulness

The case of 3 tasks

5 possible shapes & all their rotations

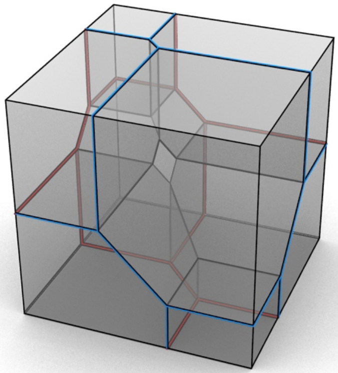




The Geometry of Truthfulness

The case of 3 tasks

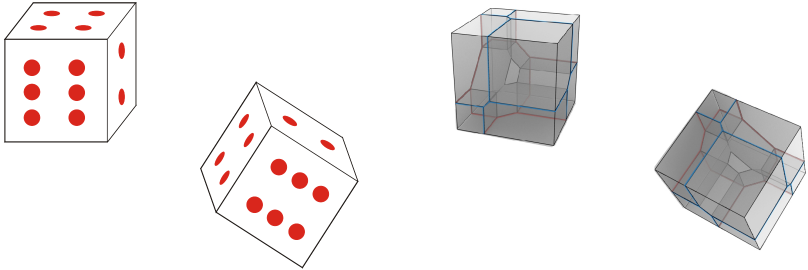
5 possible shapes & all their rotations



Throw one truthful mechnansm like a dice!

You get another truthful mechanism.

What do we mean by rotation?



The too many possible shapes are a drawback if we would like to use this characterization for obtaining lower bounds.

The question is fundamental but what about the scheduling problem?

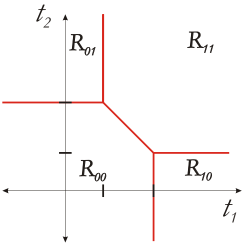
- The geometry of the 2-item case was an important tool for:
    - the characterization of 2-player mechanisms,
    - the $1 + \sqrt{2}$ lower bound for $n \geq 3$ players.
- The characterization for the 3-item case can help us improve the lower bound for $n \geq 4$ players to a better constant.
- The characterization for the m-item case might help us to find a better lower bound.
- An affine maximizer for 3 tasks (weighted VCG mechanism) is these polytopes!

The Allocation Graph

[Easier to study: you get algebraic proofs]

Edge weighted, directed graph.

- One vertex for each allocation.

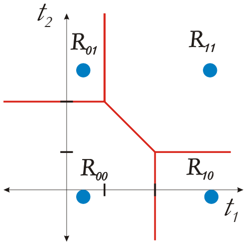


The Allocation Graph

[Easier to study: you get algebraic proofs]

Edge weighted, directed graph.

- One vertex for each allocation.
- There is an edge between two vertices of the graph iff the corresponding regions of the partition share a common boundary.

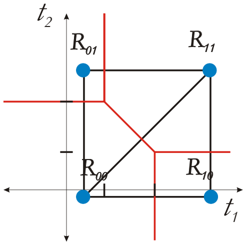


The Allocation Graph

[Easier to study: you get algebraic proofs]

Edge weighted, directed graph.

- One vertex for each allocation.
- There is an edge between two vertices of the graph iff the corresponding regions of the partition share a common boundary.

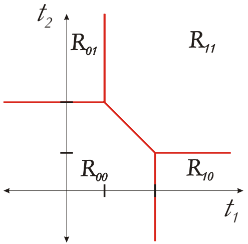


The Allocation Graph

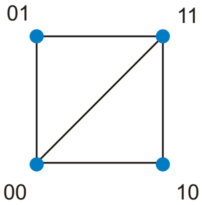
[Easier to study: you get algebraic proofs]

Edge weighted, directed graph.

- One vertex for each allocation.
- There is an edge between two vertices of the graph iff the corresponding regions of the partition share a common boundary.
- The weight of the edge between allocations a, a' is $f_{a:a'} := \sup\{(a - a')t | t \in R_a\}$.



Mechanism



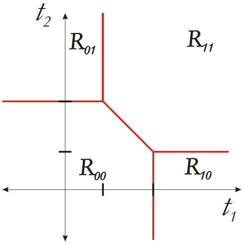
Allocation graph

The Allocation Graph

[Easier to study: you get algebraic proofs]

Edge weighted, directed graph.

- One vertex for each allocation.

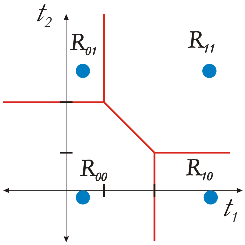


The Allocation Graph

[Easier to study: you get algebraic proofs]

Edge weighted, directed graph.

- One vertex for each allocation.
- There is an edge between two vertices of the graph iff the corresponding regions of the partition share a common boundary.

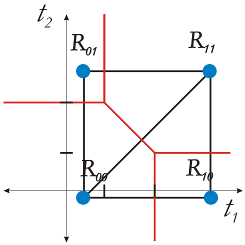


The Allocation Graph

[Easier to study: you get algebraic proofs]

Edge weighted, directed graph.

- One vertex for each allocation.
- There is an edge between two vertices of the graph iff the corresponding regions of the partition share a common boundary.

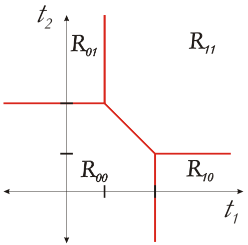


The Allocation Graph

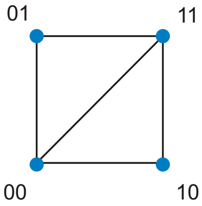
[Easier to study: you get algebraic proofs]

Edge weighted, directed graph.

- One vertex for each allocation.
- There is an edge between two vertices of the graph iff the corresponding regions of the partition share a common boundary.



Mechanism



Allocation graph

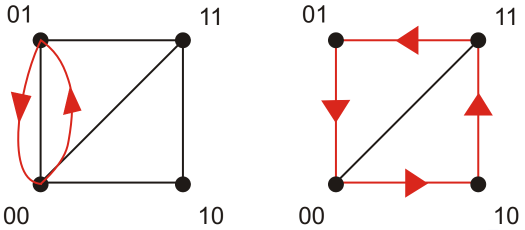
Cycle Monotonicity [Rochet 1987]

The weight of the edge between allocations a, a' is
$f_{a:a'} := \sup\{(a - a')t | t \in R_a\}$.

Monotonicity: Every two-cycle on the allocation graph has non-negative length.

What we will use: If the consequtive nodes in the cycle share a common edge then the length of the cycle is zero. (Nodes in Hd=1 allways share a common boundary!)

$$f_{00:10} + f_{10:11} + f_{11:01} + f_{01:00} = 0$$



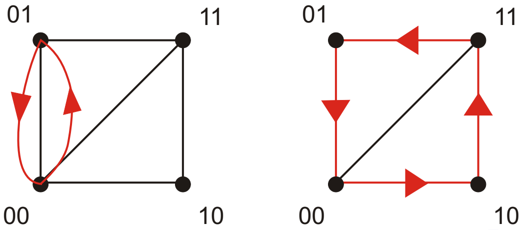
Cycle Monotonicity [Rochet 1987]

The weight of the edge between allocations a, a' is $f_{a:a'} := \sup\{(a-a')t | t \in R_a\}$.

Cycle Monotonicity: Every cycle on the allocation graph has non-negative length.

What we will use: If the consequtive nodes in the cycle share a common edge then the length of the cycle is zero. (Nodes in Hd=1 allways share a common boundary!)

$$f_{00:10} + f_{10:11} + f_{11:01} + f_{01:00} = 0$$

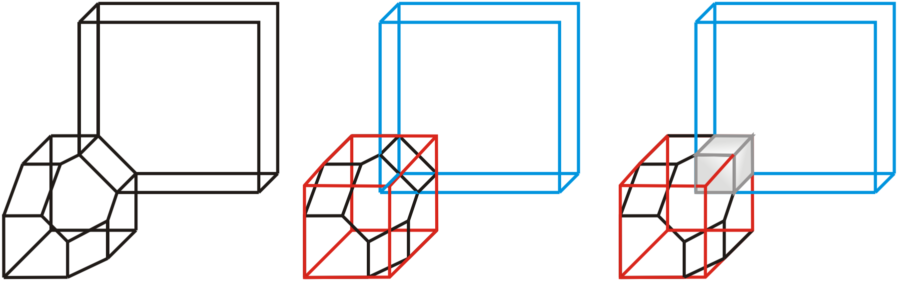


Cubism

Instead of dealing with complicated polytopes we deal with boxes!
For each region of the mechanism we define a box that contains it.

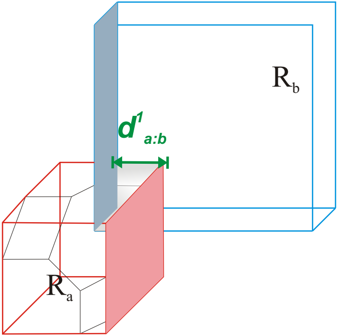
Lemma

Two regions intersect iff their coresponding boxes intersect.



Deciding if two regions intersect

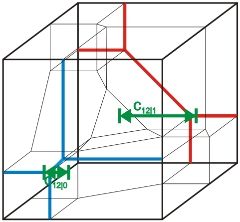
We have to measure the distances between parallel to each other hyperplanes. (We have to do this for each one of the axes.)

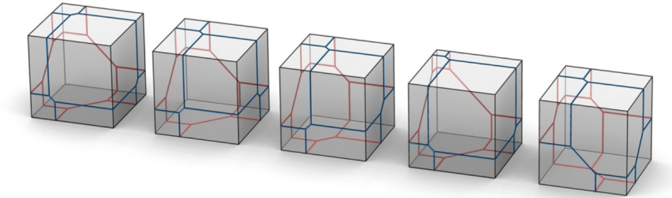


We compute these distance for any number of tasks *m*. This can be a starting point for extending the theorem to *m* tasks. But. . . we don't know which are the possible allocation graphs. . .

The unit for measurement

We will express all these distances as sums involving 6 constants, one for each projection. (We also calculate the distances for the case of *m* tasks.)



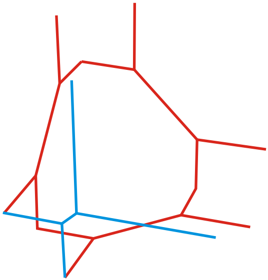


Guess the sketch!

Knowing 4 of the projections you can draw the whole mechanism.

In fact using cycle-monotonicity it turns out that we only need 4 instead of 6 constants. The reason is the following relation:

$$c_{12|0} - c_{12|1} = c_{13|0} - c_{13|1} = c_{23|0} - c_{23|1}$$

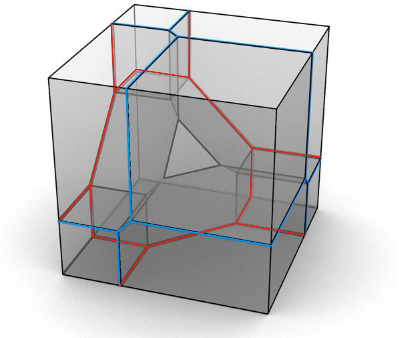


Guess the sketch!

Knowing 4 of the projections you can draw the whole mechanism.

In fact using cycle-monotonicity it turns out that we only need 4 instead of 6 constants. The reason is the following relation:

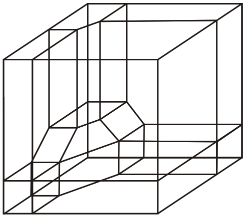
$$c_{12|0} - c_{12|1} = c_{13|0} - c_{13|1} = c_{23|0} - c_{23|1}$$



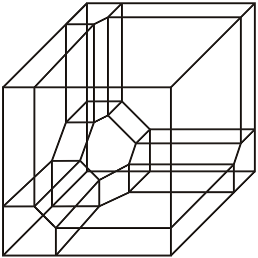
Degenerate cases [1]

An example

In fact there are more shapes. . . luckily these are degenerate cases of the five shapes we have already found.



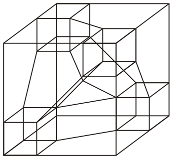
is a degenerate version of



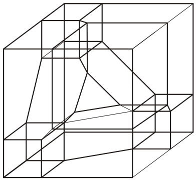
Degenerate cases [2]

Another example

In fact there are more shapes. . . luckily these are degenerate cases of the five shapes we have already found.



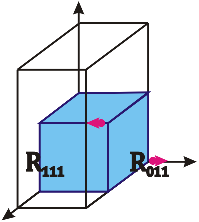
is a degenerate version of



Lower bounds for some scheduling mechanisms

Theorem

Every mechanism for which $R_{1\ldots1}$ is a box has approximation ratio at least $1+\sqrt{n}$.



A lower bound of n for non-penalizing mechanisms

Definition

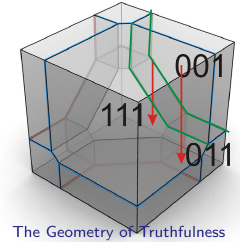
We will say that a mechanism is non-penalizing if in the allocation graph no pair of regions of the form R_{a10}, R_{b01}, where a, b are $(m-2)$-dimensional allocation vectors, share a common boundary.

If a player becomes faster for the tasks he gets he doesn't loose what he already got.

Theorem

Every non-penalizing mechanism has approximation ratio at least n.

Proof idea: Falling from 001 he can only fall in 111 or 011.



Future directions

- Generalize this characterization for more tasks and perhaps use it for improving the existing $1+\varphi \approx 2.618$ lower bound.
- Improve the lower bound for the case of 4 players using the characterization we gave here. (Generalizing the proof of the $1+\sqrt{2}$)
- Find a complete characterization of mechanisms for scheduling .