A quick tour of multiagent resource allocation

Preference representation & aggregation, complexity, algorithms...

Sylvain Bouveret, Onera Toulouse

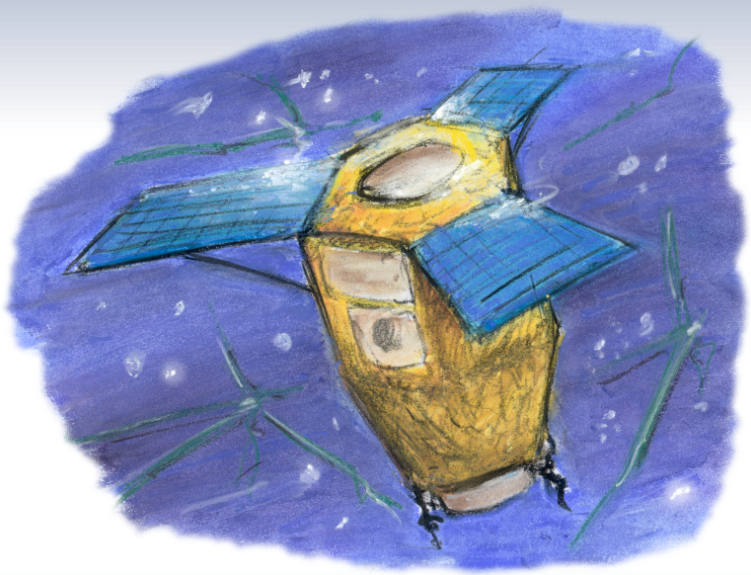
sylvain.bouveret@onera.fr

Invited lecture

University of Girona, December 14–16, 2010

Pleiades: an Earth Observing Satellite

The satellite...



A short animation...

Pleiades: an Earth Observing Satellite

A picture...

A short animation...

EOS: How does it work ?

The mission of Earth Observing Satellites :
to acquire **images**, in response to **requests** from **clients**.

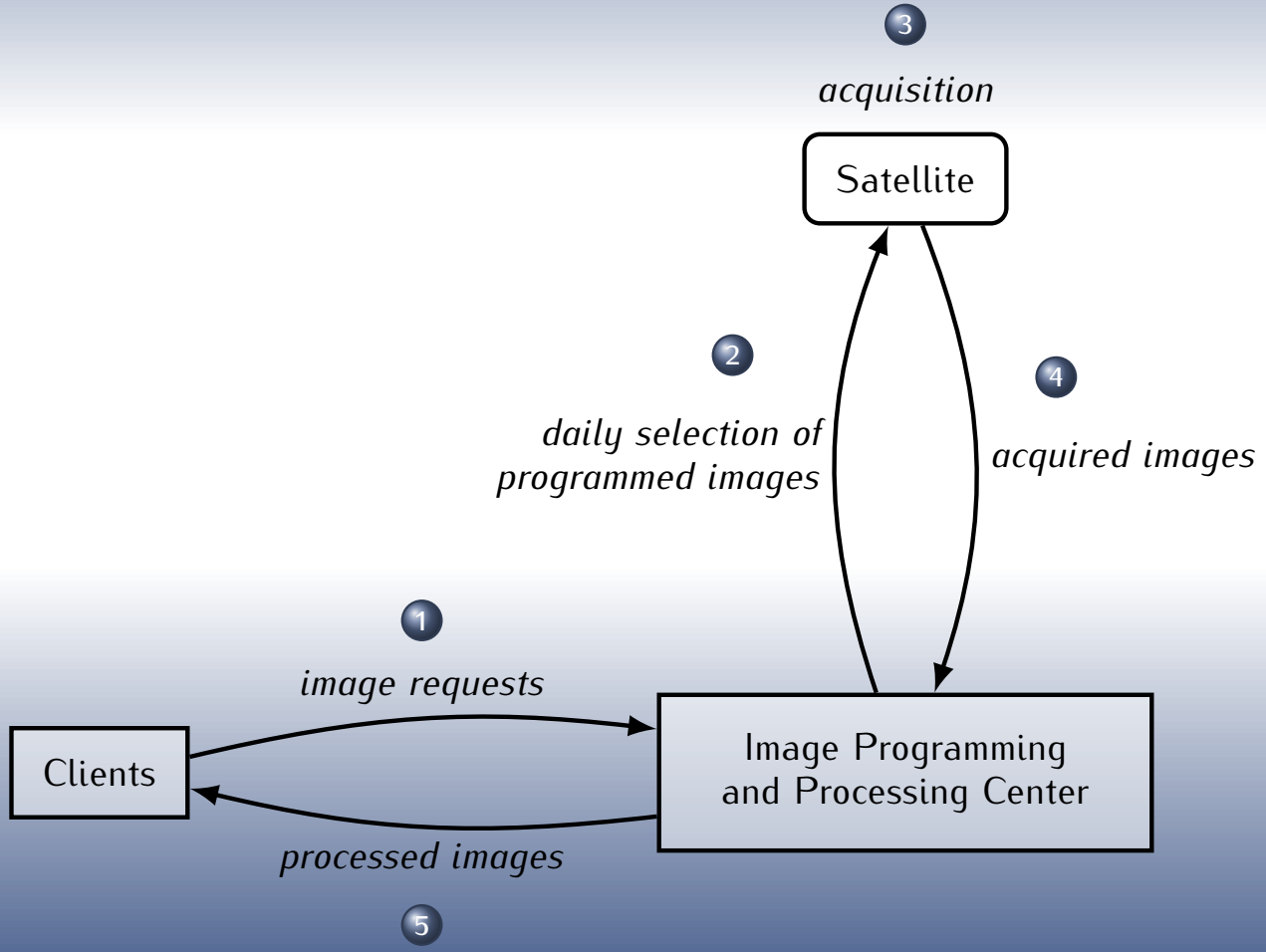


Image requests

A requested image is characterized by :

- the requesting client
- its location, size, ...
- its imaging **constraints** (ex : mono or stereo, shooting angle ...) and validity window (ex : from next June 15 to August 30)
- its **weight** given by the client $\rightsquigarrow$ expression of **preferences**

Generally, all requested images cannot be processed the same day, due to **conflicts** between them
(respect of physical and imaging constraints, minimum transition time between images ...).

The problem (informal) 1/2

The satellite (or a constellation of satellites)
is **co-funded by several agents** ...

... and then **exploited in common**.

ex : PLEIADES satellite $\rightsquigarrow$ France/Italy/Spain, civil/defense

The daily problem : select each day, among the set of valid image requests, **a subset** of images to be programmed this day.

The (daily) selection of programmed images is seen as **an allocation problem of indivisible objects (images) to agents (clients)**.

The problem (informal) 2/2

Selections of programmed images must be

- **admissible** : respect constraints
- **efficient** : the satellite(s) must not be under-exploited
- **fair** : for each agent, its "return on investment" should be proportional to its financial contribution.

The problem (informal) 2/2

Selections of programmed images must be

- **admissible** : respect constraints
- **efficient** : the satellite(s) must not be under-exploited
- **fair** : for each agent, its "return on investment" should be proportional to its financial contribution.

Every day, hundreds of requests to deal with ...

1. choose the principles of fairness
2. formalize the model
3. analyze the model (properties, complexity...)
4. design methods/algorithms following the principles.

Context

Studies for the french space agency (CNES) by ONERA Centre de Toulouse.

Participating people :

- N. Bataille, J.-M. Lachiver, CNES
- S. Bouveret, M. Lemaître, G. Verfaillie, ONERA
- H. Fargier, J. Lang, CNRS / IRIT

Sylvain Bouveret's thesis.

Some work supported by the ANR-PHAC project (French Research National Agency).

Outline

1. **The basic ingredients**:
   where we will introduce the different elements of a resource allocation problem.
2. **Preference aggregation**:
   where we will formally define efficiency and fairness, based on microeconomics.
3. **Preference representation**:
   where we will see why we need compact representation languages, and speak about complexity.
4. **Solving multiagent resource allocation problems**:
   where we will see how to solve multiagent resource allocation problems.
5. **Conclusions, other applications, other issues**:
   where we will briefly discuss some real-world applications, and some issues that we did not discuss in the lecture.

Prerequisites

- Basic mathematical notions (functions, set theory, binary relations, ...)
- Basic notions of graph theory
- Basic notions of algorithmics and computer science
- Some notions of complexity theory and propositional logic would help
- Basic level of English...

Talk inspired from...

- **Tutorial on MultiAgent Resource Allocation** given by *Ulle Endriss and Nicolas Maudet* at the European Agent Systems Summer School 2006-2007

  http://staff.science.uva.nl/ ulle/teaching/easss-2007/

- **Tutorial on Fair Division** given by *Ulle Endriss* at the COST-ADT Doctoral School on COMSOC in 2010

  http://staff.science.uva.nl/ ulle/teaching/cost-adt-2010/

- **Lecture on Combinatorial Auctions** given by **Ulle Endriss** at the ILLC Amsterdam

  http://staff.science.uva.nl/ ulle/teaching/comsoc/2009/slides/comsoc-auctions.pdf

Some books and articles...

Chevaleyre, Y., Dunne, P. E., Endriss, U., Lang, J., Lemaître, M., Maudet, N., Padget, J., Phelps, S., Rodríguez-Aguilar, J. A., and Sousa, P. (2006).
Issues in multiagent resource allocation.
Informatica, 30:3–31.

Cramton, P., Shoham, Y., and Steinberg, R., editors (2006).
Combinatorial Auctions.
MIT Press.

Lang, J. (2004).
Logical preference representation and combinatorial vote.
Annals of Mathematics and Artificial Intelligence, 42(1):37–71.

Moulin, H. (1988).
Axioms of Cooperative Decision Making.
Cambridge University Press.

Moulin, H. (2003).
Fair Division and Collective Welfare.
MIT Press.

Uckelman, J. (2008).
More Than the Sum of Its Parts. Compact Preference Representation of Combinatorial Domains.
PhD thesis, Universiteit van Amsterdam.

Part 1

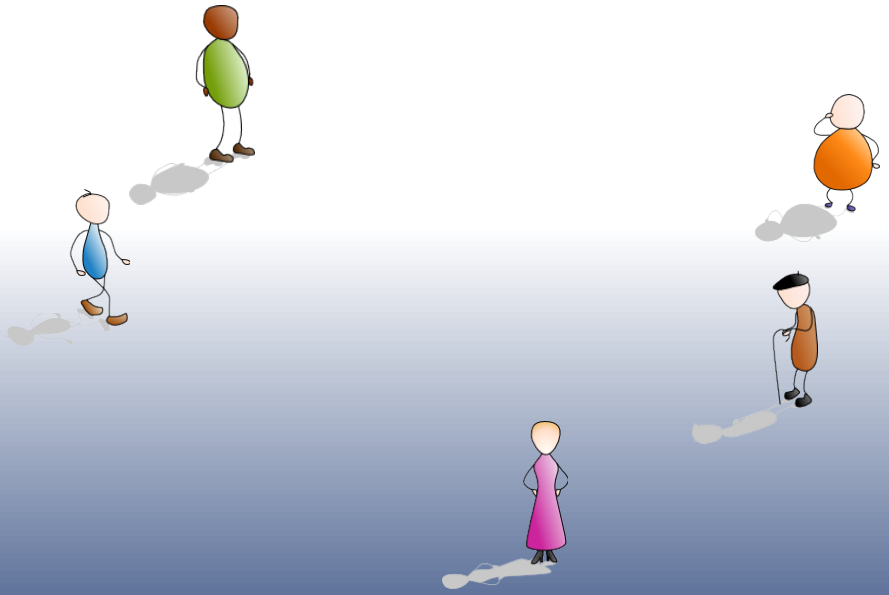
The basic ingredients

The basic ingredients

The resource allocation problem...

The resource allocation problem...

- A finite set $\mathcal{N}$ of **agents** .

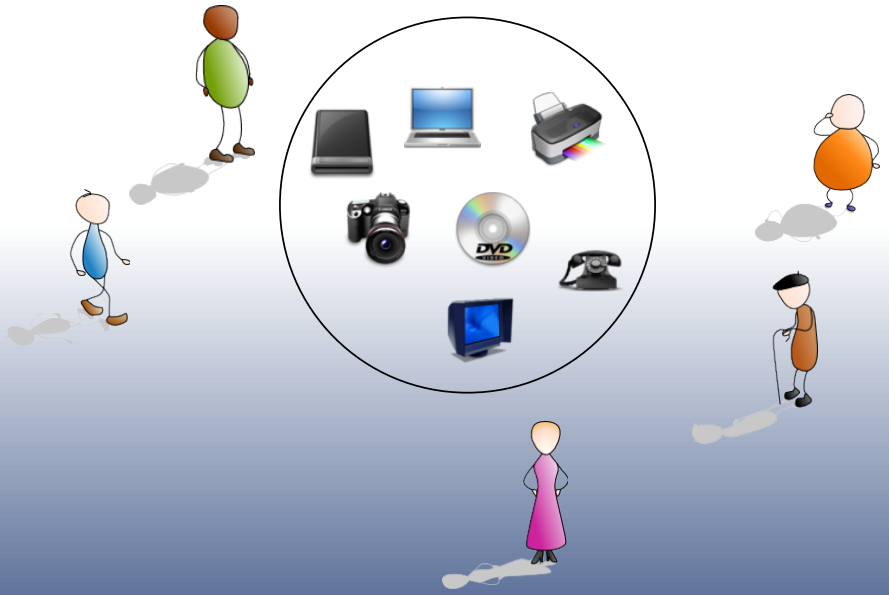


The resource allocation problem...

- A finite set $\mathscr{N}$ of **agents** .
- A limited common **resource**.

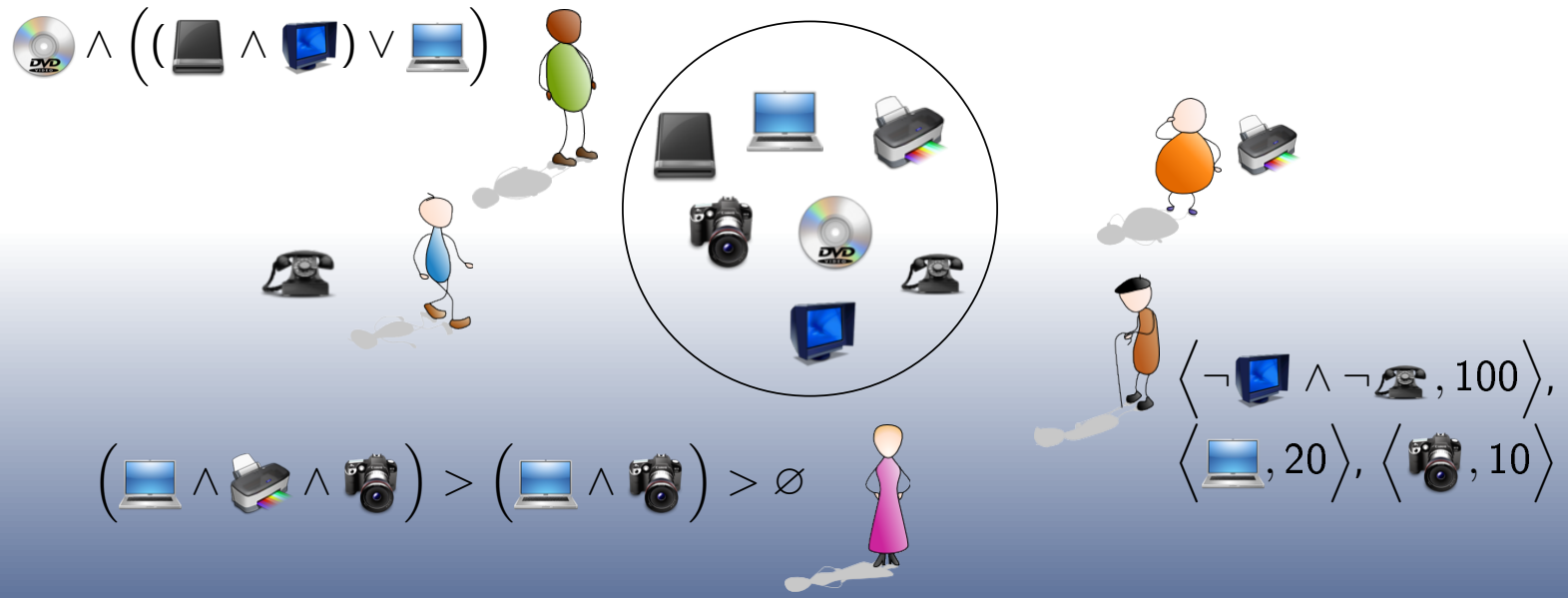
The resource allocation problem...

- A finite set $\mathcal{N}$ of **agents** .
- A limited common **resource**.



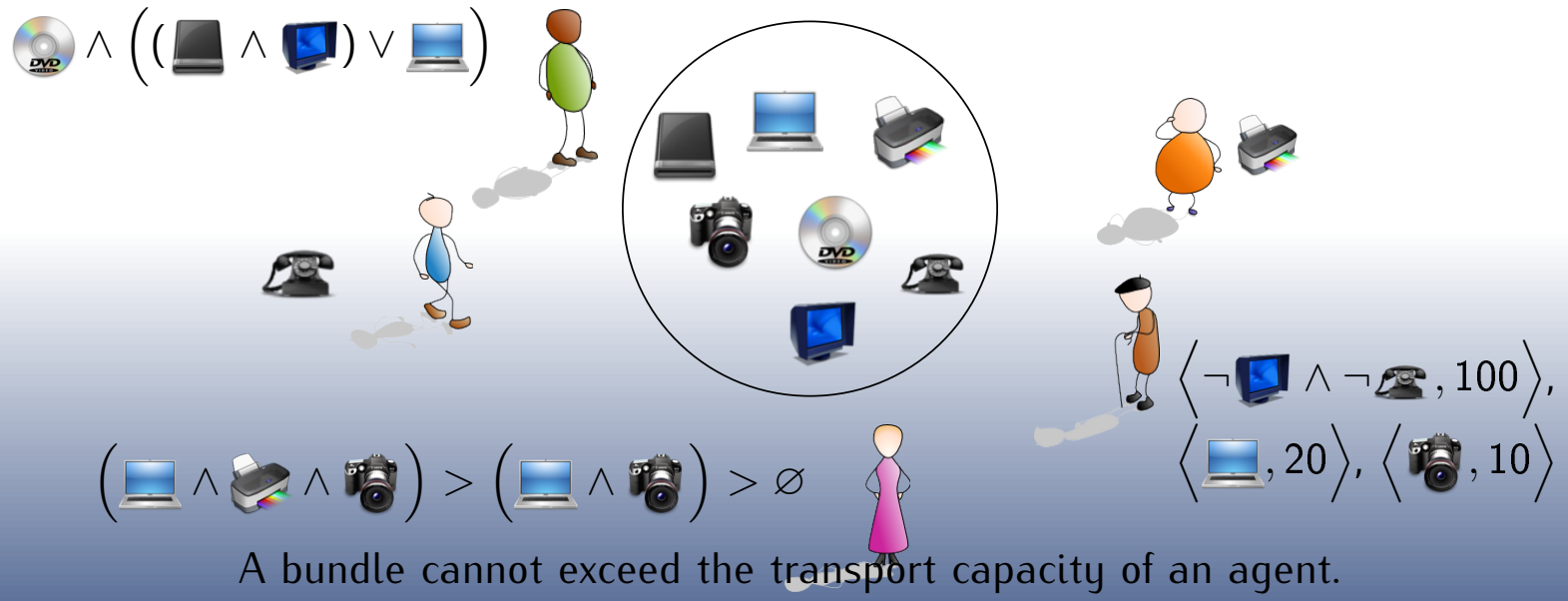
The resource allocation problem...

- A finite set $\mathscr{N}$ of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.



The resource allocation problem...

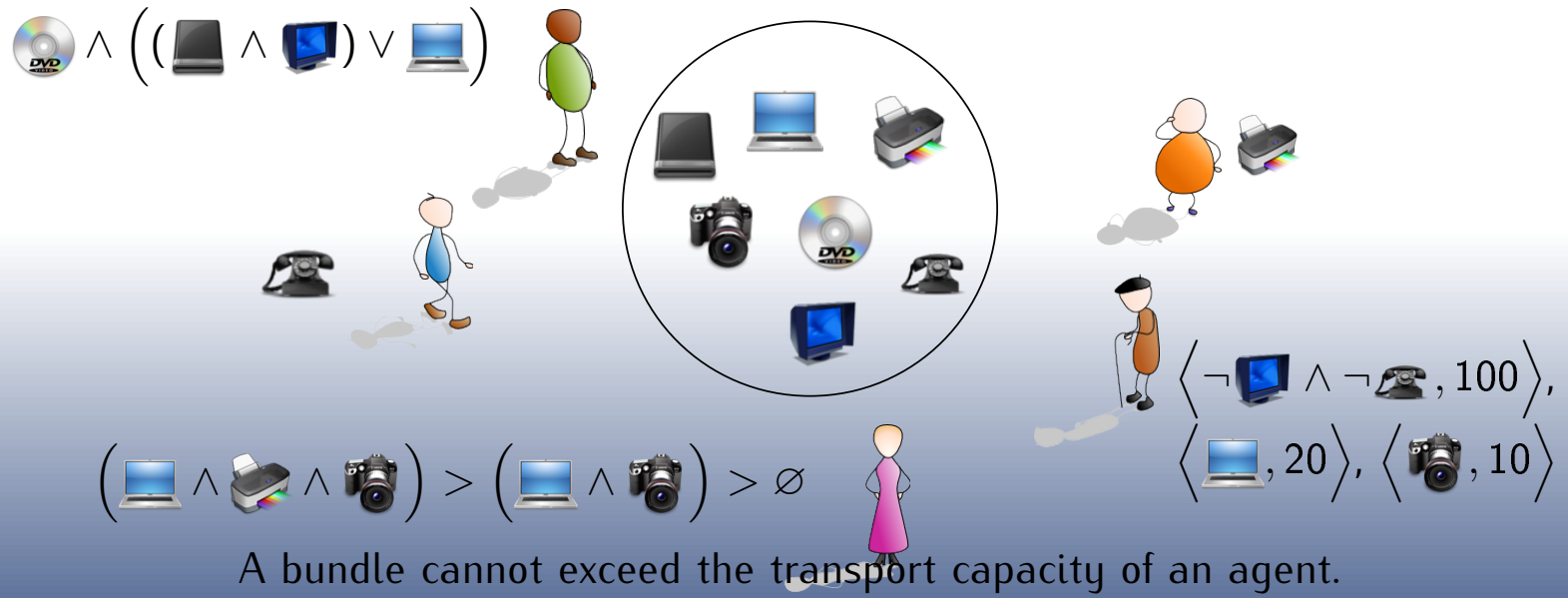
- A finite set $\mathcal{N}$ of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.
- A set of **constraints** (physical, legal, moral,...).



A bundle cannot exceed the transport capacity of an agent.

The resource allocation problem...

- A finite set $\mathcal{N}$ of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.
- A set of **constraints** (physical, legal, moral,...).
- An **optimization** or **decision** criterion.

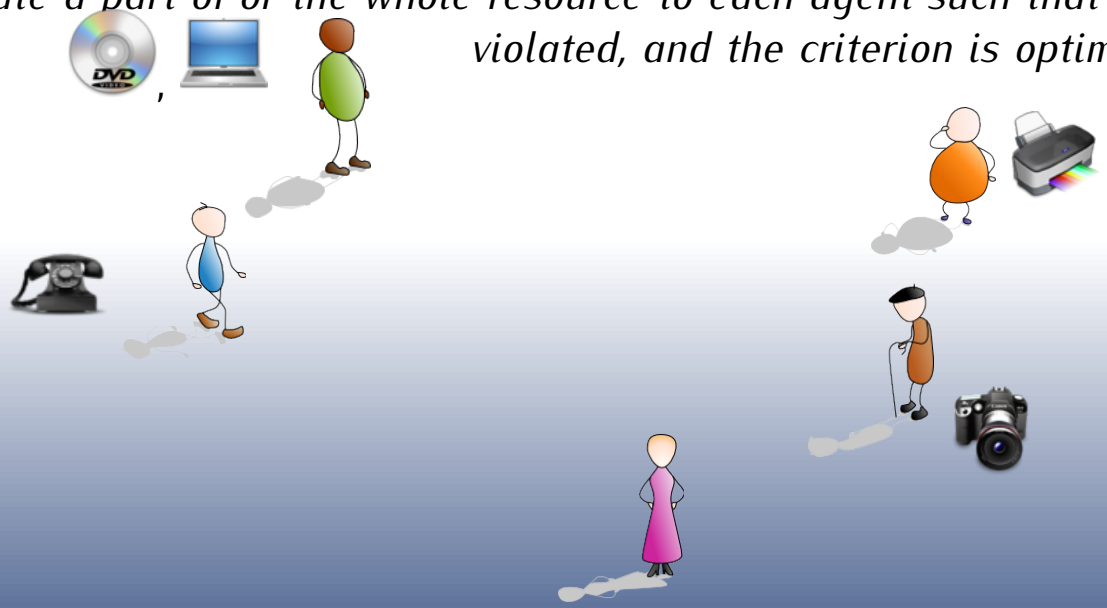


A bundle cannot exceed the transport capacity of an agent.

The resource allocation problem...

- A finite set $\mathcal{N}$ of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.
- A set of **constraints** (physical, legal, moral,...).
- An **optimization** or **decision** criterion.

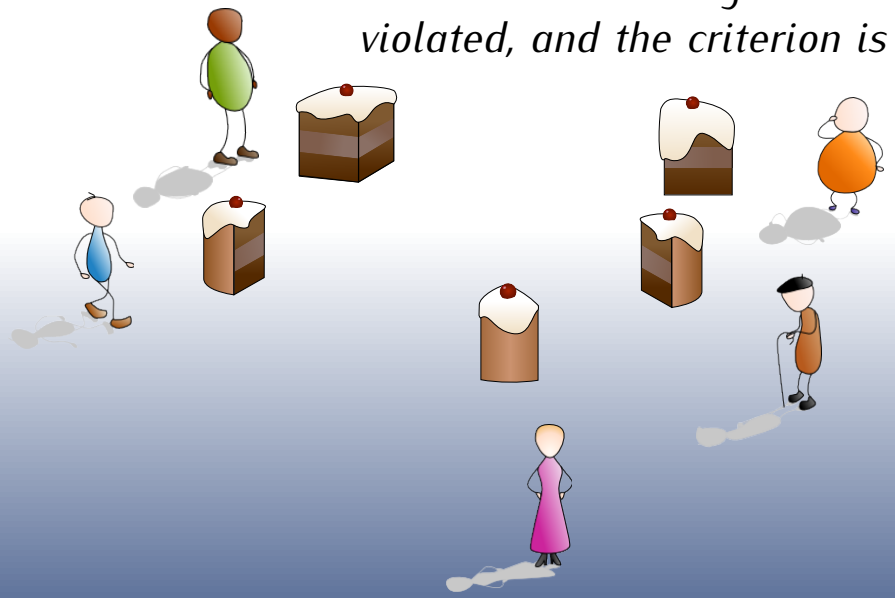
How to allocate a part of or the whole resource to each agent such that no constraint is violated, and the criterion is optimized or verified.



The resource allocation problem...

- A finite set $\mathcal{N}$ of **agents** having some **requests** and **preferences** on the resources.
- A limited common **resource**.
- A set of **constraints** (physical, legal, moral,...).
- An **optimization** or **decision** criterion.

How to allocate a part of or the whole resource to each agent such that no constraint is violated, and the criterion is optimized or verified.



Real-world applications

An ubiquitous problem...

- Fair share of Earth Observation Satellites.
- Tasks or subjects allocation.
- Combinatorial auctions problems [Cramton et al., 2006].
- Allocation of waste water treatement rights [Murillo Espinar, 2010].
- Computer network sharing, rostering problems, allocation of take-off and landing slots in airports [Faltings, 2005],....

Cramton, P., Shoham, Y., and Steinberg, R., editors (2006).
Combinatorial Auctions.
MIT Press.

Faltings, B. (2005).
A budget-balanced, incentive-compatible scheme for social choice.
In Faratin, P. and Rodriguez-Aguilar, J. A., editors, *Agent-Mediated Electronic Commerce VI*, volume 3435 of *LNAI*, pages 30–43. Springer.

Murillo Espinar, J. (2010).
Egalitarian Behaviour in Multiunit Combinatorial Auctions.
PhD thesis, Universitat de Girona.

The basic ingredients

The agents

The agents can be:

- **individuals** from a given (small) society;
- **institutional** or **private entities** (*e.g* companies);
- **computers**, acting on behalf of human agents;
- **machines**, among which one must allocate some tasks.

Agents

In the following, we will write $\mathscr{N} = \langle 1, \ldots, n \rangle$ the vector of agents at stake.

Exogenous rights

The agents can be of **different importance**, for reasons that are completely (or partially) exogenous to the problem at stake.
Examples:

- Seniority
- Different investments in creating the resource (*e.g* Earth Observing Satellites)
- Agents representing different groups of individuals of different sizes (*e.g* council of Europe)

Exogenous rights

Given a vector $\mathscr{N}$ of agents of size n, we will write $\overrightarrow{e} \in \mathbb{N}^n$ the vector of exogenous rights.

Remark: unless explicitly stated, we will suppose in the following that $\overrightarrow{e} = \langle 1, \ldots, 1 \rangle$.

The basic ingredients

Types of Resources

- A central parameter in any resource allocation problem is the nature of the resources themselves.
- There is a whole range of different **types of resources**, and each of them may require different techniques...
- Distinguish properties of the **resources** themselves and characteristics of the chosen **allocation mechanism**.
  Examples:
  - Resource-inherent property: Is the resource perishable?
  - Characteristic of the allocation mechanism: Can the resource be shared amongst several agents?

Continuous vs. Discrete Resources

- Resource may be **continuous** (e.g. energy) or **discrete** (e.g. laptop).
- **Discrete** resources are **indivisible**.
- **Continuous** resources may be treated either as being (infinitely) **divisible** or as being **indivisible** (e.g. only sell orange juice in units of 50 litres $\rightsquigarrow$ discretisation).
- Representation of a single bundle:
  - **Several continuous resources**: vector over $[0; 1]$
    for example: $\langle 10.5\%, 88\%, 32\% \rangle$
  - **Several discrete resources**: vector over non-negative integers
    for example: $\langle 5, 12, 45 \rangle$
  - **Several distinguishable discrete resources**: vector over $\mathbb{B} = \{0, 1\}$
    for example: $\langle 0, 0, 1 \rangle$

Continuous vs. Discrete Resources

- Classical literature in economics mostly concentrates on a **single continuous resource** (*e.g* cake-cutting);
- recent work in AI and Computer Science focuses on discrete resources.

- **Remark**: In addition, **money** can intervene as a special continuous resource to compensate the inequity of a resource allocation.

Sharable or not

- A **sharable** resource can be allocated to a number of different agents at the same time.
  Examples:
  - a photo taken by an earth observation satellite
  - path in a network (network routing)
  - software licenses
- More often though, resources are assumed to be **non-sharable** and can only have a single owner at a time.
  Examples:
  - energy to power a specific device
  - laptop to be used by the agent obtaining it only

Remark: this is roughly the same dichotomy between *rival* (physical) and *non-rival* (virtual, informational) goods.

Static or not

Resources that do not change their properties during a negotiation process are called **static** resources. There are at least two types of resources that are not static:

- **consumable** goods such as fuel
- **perishable** goods such as food

In general, resources cannot be assumed to be static. However, in many cases it is reasonable to assume that they are as far as the negotiation process at hand is concerned.

Handling dynamic resources is crucial *e.g* in **repeated allocation of perishable goods**.

Single-unit vs. Multi-unit

- In **single-unit** settings there is exactly one copy of each type of good; all items are distinguishable (e.g. several houses).
- In **multi-unit** settings there may be several copies of the same type of good (e.g. 10 bottles of wine).
- Note that this distinction is only a matter or **representation**:
  - Every multi-unit problem can be translated into a single-unit problem by introducing new names (inefficient, but possible).
  - Every single-unit problem is in fact also a (degenerate) multi-unit problem.
- Multi-unit problems allow for a more **compact** representation of allocations and preferences, but also require a richer **language** (variables ranging over integers, not just binary values).

Resources vs. Tasks

- **Tasks** may be considered resources with **negative utility** (cost).
- Hence, **task allocation** may be studied as MultiAgent Resource Allocation problem.
- Tasks are often coupled with **constraints** regarding their coherent combination (multiagent scheduling)
  Examples:
  - Crew scheduling
  - Nurse rostering
  - Task allocation on machines

Resource, objects, bundles

More formally...

Resource

- A **continuous resource** is a set isomorphic to $[0, 1]$.
- A **discrete resource** is a finite set $\mathscr{O} = \{o_1, \ldots, o_p\}$.

In the following, we will restrict our setting to a **discrete**, **static**, **single-unit** resource.

Elements of $\mathscr{O}$ are called (indivisible) **items**, **goods** or **objects**.

A **bundle** is an element $\pi \in 2^{\mathscr{O}}$.

Allocation

Allocation

Given a set $\mathscr{N}$ of n agents, and a set $\mathscr{O}$ of objects, an **allocation** is a vector $\overrightarrow{\pi} = \langle \pi_1, \ldots, \pi_n \rangle \in 2^{\mathscr{O}^n}$.

For each $i \in \mathscr{N}$, $\pi_i \in 2^{\mathscr{O}}$ is called the **share** of agent i.

Remark: notice that nothing prevents π_i and π_j from overlapping $\rightsquigarrow$ goods are **sharable**.

Constraints

- **Constraints** are physical, legal or moral elements that restrict the set of possible allocations.
- **Examples**:
  - no agent is allowed to get more than 2 objects.
  - **Earth Observing Satellites**: due to transition times, o_1 and o_5 cannot be allocated both (at least one of these two objects is not allocated).

Constraints

- **Constraints** are physical, legal or moral elements that restrict the set of possible allocations.
- **Examples**:
  - no agent is allowed to get more than 2 objects.
  - **Earth Observing Satellites**: due to transition times, o_1 and o_5 cannot be allocated both (at least one of these two objects is not allocated).

Admissibility constraint

A constraint is a subset $C \subset 2^{\mathscr{O}^n}$.

Admissible allocation

given a set $\mathscr{C}$ of constraints, an admissible allocation $\overrightarrow{\pi}$ is an element of $\bigcap_{C \in \mathscr{C}} C$.

Examples

The preemption constraint: no two shares overlap.

Preemption constraint

$$C_{preempt} = \{\overrightarrow{\pi} \mid \forall i \neq j,\ \pi_i \cap \pi_j = \varnothing\}$$

Examples

The preemption constraint: no two shares overlap.

Preemption constraint

$C_{preempt} = \{\overrightarrow{\pi} \mid \forall i \neq j,\ \pi_i \cap \pi_j = \varnothing\}$

No free disposal: all the objects must be allocated.

No free disposal

$C_{nfd} = \{\overrightarrow{\pi} \mid \bigcup_{i \in \mathcal{N}} \pi_i = \mathcal{O}\}$

Examples

The preemption constraint: no two shares overlap.

Preemption constraint

$C_{preempt} = \{\overrightarrow{\pi} \mid \forall i \neq j,\ \pi_i \cap \pi_j = \varnothing\}$

No free disposal: all the objects must be allocated.

No free disposal

$C_{nfd} = \{\overrightarrow{\pi} \mid \bigcup_{i \in \mathcal{N}} \pi_i = \mathcal{O}\}$

Volume constraint: every object o_k has a volume $\nu(o_k)$, and there is a maximal volume of objects V_{max} that an agent can take.

Volume constraint

$C_{\nu, V_{max}} = \{\overrightarrow{\pi} \mid \forall i,\ \sum_{o_k \in \pi_i} \nu(o_k) \leq V_{max}\}$

The basic ingredients

Preference ?

We now have a formal definition of **agents**, **resource**, **allocation** and **constraints** ...

What about the agents' preferences on the resource they might receive ?

Preference ?

We now have a formal definition of **agents**, **resource**, **allocation** and **constraints** ...

What about the agents' preferences on the resource they might receive ?

- They are not admissibility constraints (they might be satisfied or not).
- Contrary to admissibility constraints, they can be satisfied **at different levels**.
- They can be **antagonistic** (*e.g* two agents willing the same non sharable object).

Preference ?

We now have a formal definition of **agents**, **resource**, **allocation** and **constraints** ...
What about the agents' preferences on the resource they might receive ?

- They are not admissibility constraints (they might be satisfied or not).
- Contrary to admissibility constraints, they can be satisfied **at different levels**.
- They can be **antagonistic** (*e.g* two agents willing the same non sharable object).

More generally, given a set of **options**, *what does it mean to have preferences on this set ?*

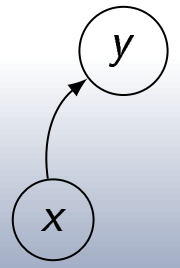
Preference structure

- Given a set of **alternatives** (issues, outcomes, options...) $\mathscr{E}$.
- Having some **preferences** on the alternatives $\mathscr{E}$ informally means "not being indifferent about which one is chosen".

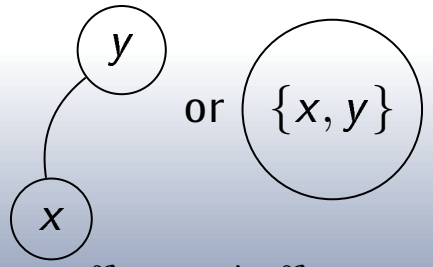
Preference structure

Binary reflexive relation $\Re_S$ on the set of alternatives $\mathscr{E}$.

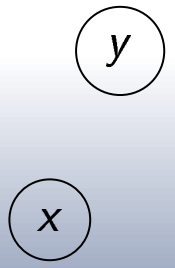
$$x \Re_S y \Leftrightarrow x \text{ is at least as good as } y.$$



$x \Re_S y$ and not $y \Re_S x$.



$x \Re_S y$ and $y \Re_S x$.



not $x \Re_S y$ and not $y \Re_S x$.

Different preference structures

- **Ordinal** preference structure.
  - **Ranking**.
  - **Dichotomous** preference structure.
- **Cardinal** preference structure.
- Semi-orders (threshold models), interval orders (variable threshold models), fuzzy preference structure,...

Quick reminder

Binary relations $\Re$...

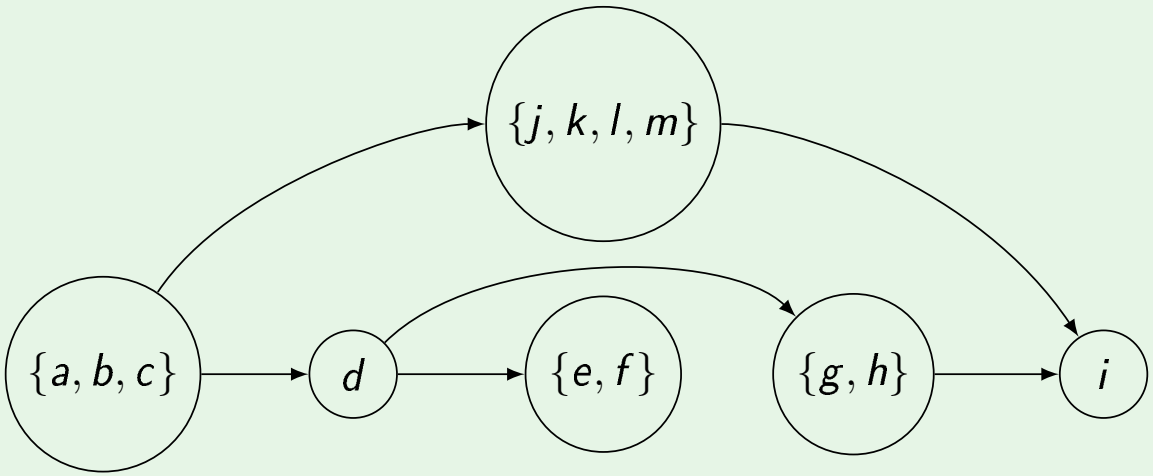
- **reflexive**: for all $x \in X$ it holds that $x\Re x$. For example, "greater than or equal to" is a reflexive relation but "greater than" is not.
- **antisymmetric**: for all x and y in X it holds that if $x\Re y$ and $y\Re x$ then $x = y$. "Greater than or equal to" is an antisymmetric relation, because if $x \geq y$ and $y \geq x$, then $x = y$.
- **asymmetric**: for all x and y in X it holds that if $x\Re y$ then not $y\Re x$. "Greater than" is an asymmetric relation, because if $x > y$ then not $y > x$.
- **transitive**: for all x, y and z in X it holds that if $x\Re y$ and $y\Re z$ then $x\Re z$. "Is an ancestor of" is a transitive relation, because if x is an ancestor of y and y is an ancestor of z, then x is an ancestor of z.
- **total**: for all x and y in X it holds that $x\Re y$ or $y\Re x$ (or both). "Is greater than or equal to" is an example of a total relation (this definition for total is different from the one in the previous section).

Ordinal preferences

Ordinal preference structure

A preorder $\succeq$ on the alternatives ($\Re_S$ + transitivity). (usually required to be complete as well)

Example

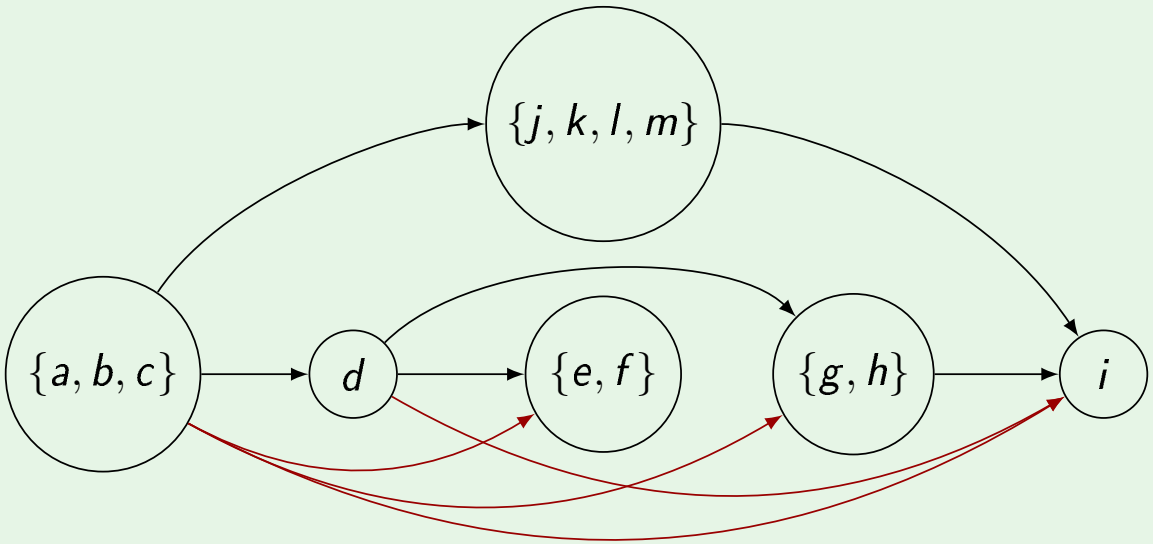


Ordinal preferences

Ordinal preference structure

A preorder $\succeq$ on the alternatives ($\Re_S$ + transitivity). (usually required to be complete as well)

Example



Ranking

- A special case of ordinal preferences: a **linear order**.
- Used mostly in **voting theory**.

Example

$$c \rightarrow b \rightarrow e \rightarrow a \rightarrow f \rightarrow d$$

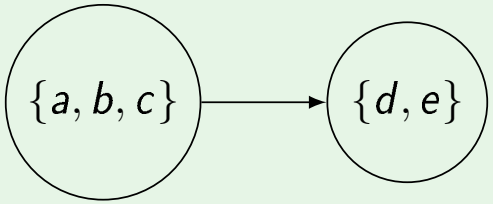
Dichotomous preferences

A special kind of ordinal preferences, with two equivalence classes:

- a set of "good" alternatives $\mathscr{G}$,
- a set of "bad" alternatives $\mathscr{E} \setminus \mathscr{G}$.

$$e \succeq e' \Leftrightarrow (e \in \mathscr{G} \text{ or } e' \in \mathscr{E} \setminus \mathscr{G})$$

Example



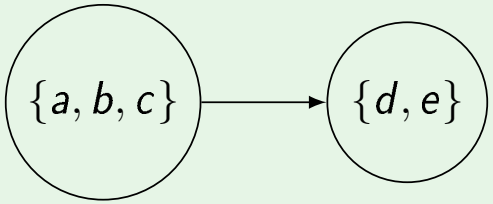
Dichotomous preferences

A special kind of ordinal preferences, with two equivalence classes:

- a set of "good" alternatives $\mathscr{G}$,
- a set of "bad" alternatives $\mathscr{E} \setminus \mathscr{G}$.

$$e \succeq e' \Leftrightarrow (e \in \mathscr{G} \text{ or } e' \in \mathscr{E} \setminus \mathscr{G})$$

Example



Remark: Qualitative preferences are an extension of dichotomous preferences with n ordered equivalence classes.
Example: very good $\geq$ good $\geq$ bad $\geq$ very bad.

Cardinal preferences

From qualitative preferences to **numerical** preferences...

We just add some **algebraic** structure (*e.g* numbers)

Cardinal preferences

From qualitative preferences to **numerical** preferences...

We just add some **algebraic** structure (*e.g* numbers)

Cardinal preference structure

Refinement of the ordinal model by a **utility function** $u : \mathscr{E} \to \mathscr{V}$.
$\mathscr{V}$ is a totally ordered valuation space (*e.g.* $\mathbb{R}$, $\mathbb{N}$).

Example

	a	b	c	d	e
u	5	2	3	8	0

Remark: Every utility function u **induces** a complete preorder $\preceq$ on the set of alternatives $\rightsquigarrow x \preceq y$ iff $u(x) \leq u(y)$.

Ordinal vs Cardinal preferences

- **Intrapersonal comparison**: ordinal and cardinal preferences allow for comparing the satisfaction of an agent for different alternatives.
- **Interpersonal comparison**: ordinal preferences don't allow for interpersonal comparison ("Ann likes x more than Bob likes y").
- **Preference intensity**: ordinal preferences cannot express preference intensity; cardinal preferences can (subject to Val being numerical).
- **Representability**: an ordinal preference relation is representable by a utility function u: $x \preceq y$ iff $u(x) \leq u(y)$.
- **Cognitive relevance**: hard to make general statements, but at least ordinal preferences don't require reasoning with numerical utilities.

Other preference structures

- **Semi-orders** (or quasi-orders) $\rightsquigarrow$ see *e.g* [Vincke, 1978]

$$\forall (x, y) \in \mathscr{E}^2,\ x \Re_S y \Leftrightarrow g(x) \preceq g(y) + q$$

- **Interval orders** $\rightsquigarrow$ see *e.g* [Pirlot and Vincke, 1997]

$$\forall (x, y) \in \mathscr{E}^2,\ x \Re_S y \Leftrightarrow g(x) \preceq g(y) + q(g(y))$$

- **Fuzzy preference structure** $\rightsquigarrow$ see *e.g*
  [Fodor and Roubens, 1994, Perny and Roy, 1992] $\mu : \mathscr{E}^2 \to [0; 1]$
  credibility of the assumption "x preferred to y".

Set of alternatives

Back to multiagent resource allocation...

What is the set of alternatives ?

Set of alternatives

Back to multiagent resource allocation...

What is the set of alternatives ?

- The set of possible **allocations**

Example

Two agents, two objects $\{a, b\}$:

$$\mathscr{E} = \{(\varnothing, \varnothing), (\varnothing, \{a\}), (\varnothing, \{b\}), (\varnothing, \{a, b\}), \ldots, (\{a, b\}, \{a, b\})\}$$

Set of alternatives

Back to multiagent resource allocation...

What is the set of alternatives ?

- The set of possible **shares** (**non exogenous** preferences, **no externalities**)

Each agent can only express preferences on the set of possible allocations (in particular, she cannot take into account what the others receive).

Example

Two agents, two objects $\{a, b\}$:

$$\mathscr{E} = \{\varnothing, \{a\}, \{b\}, \{a, b\}\}$$

Monotony

Remark: When the set of alternatives is $2^{\mathscr{O}}$, one can define another property of preference structures.

Monotonic preferences

A preference relation $\preceq$ is **monotonic** if and only for all bundles π, π',

$$\pi \subseteq \pi' \Rightarrow \pi \preceq \pi'.$$

The basic ingredients

The resource allocation problem

A resource allocation problem

- **Inputs**
  - A set $\mathscr{N}$ of **agents** expressing **preferences** on the resource using preorders $\succeq_i$ or utility functions u_i.
  - **The resource** $\rightsquigarrow$ a finite set $\mathscr{O}$ of indivisible objects.
  - **Some constraints** $\rightsquigarrow$ a finite set $\mathscr{C} \subset 2^{2^{\mathscr{O}^n}}$.
  - **A decision or optimisation criterion**.
- **Output**
  The allocation of a part of or the whole resource to each agent such that no constraint is violated and the criterion optimized or verified.

Part 2

Preference aggregation

What is a Good Allocation?

In this part of the tutorial we are going to give an overview of criteria that have been proposed for deciding what makes a "good" allocation:

- Of course, there are application-specific criteria, e.g.:
  - "the allocation allows the agents to solve the problem"
  - "the auctioneer has generated sufficient revenue"

  Here we are interested in general criteria that can be defined in terms of the individual agent preferences (preference aggregation).
- As we shall see, such criteria can be roughly divided into:
  - **fairness** criteria;
  - **efficiency** criteria (*e.g* the constellation must not be underexploited).

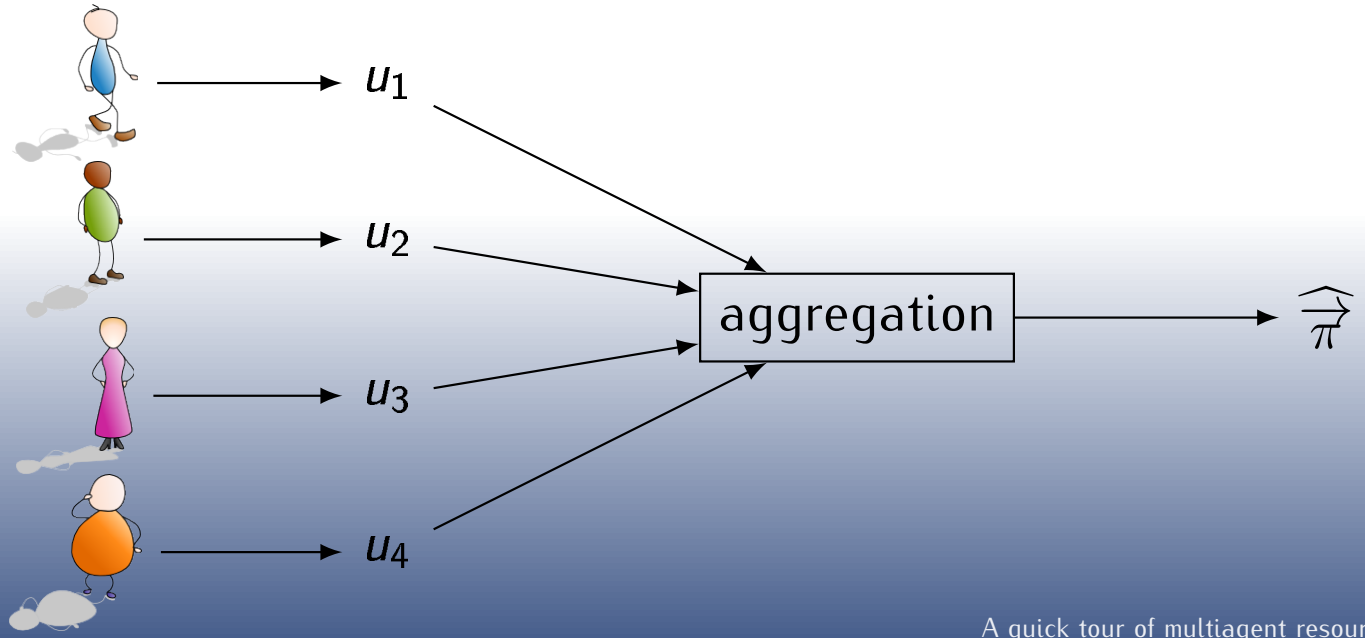
Preference aggregation...

The problem: *How to distribute the resource among the agents, in a way such that it takes into account in an equitable and efficient way their antagonistic preferences ?*

Preference aggregation...

The problem: *How to distribute the resource among the agents, in a way such that it takes into account in an equitable and efficient way their antagonistic preferences ?*

The prominent theory of **cardinal welfarism** handles this collective decision making problem by attaching to each feasible alternative the vector of individual utilities $\langle u_1, \ldots, u_n \rangle$.



Cardinal welfarism

- **Cardinal welfarism** goes back to Jeremy Bentham (1748-1832) and John Stuart Mill (1806-1873) $\rightsquigarrow$ classical utilitarianism:
  - a number representing the sums of pains and joys of a given individual;
  - goal of a benevolent decide: maximizing the total welfare of the society.
- Criticisms against classical utilitarianism: John Rawls $\rightsquigarrow$ egalitarianism.
- The cardinal welfarist point of view is acceptable in a **micro-economical** context.

Notation and Terminology

- Let $\mathscr{N} = \{1, \ldots, n\}$ be a set of agents (or players, or individuals) who need to share several goods (or resources, items, objects).
- An allocation $\overrightarrow{\pi}$ is a vector of **bundles** (shares).
- Most criteria will not be specific to allocation problems, so we also speak of **agreements** (or outcomes, solutions, alternatives, states).
- Each agent $i \in \mathscr{N}$ has a **utility function** u_i, mapping allocations (or agreements) to the reals.
- **Assumption of non exogenous preferences:** $u_i(\overrightarrow{\pi}) = u_i(\pi_i)$, so we can typically assume that each u_i is defined over the set of **bundles** $\rightsquigarrow$ $u_i : 2^{|\mathscr{O}|} \to \mathbb{R}$.
- An allocation $\overrightarrow{\pi}$ gives rise to a **utility profile** $\langle u_1(\pi_1), \ldots, u_n(\pi_n) \rangle$.

The cardinal welfarism

The theory of **cardinal welfarism** handles this collective decision making problem by attaching to each feasible alternative the vector of individual utilities $\langle u_1, \ldots, u_n \rangle$.

Social Welfare Ordering

A social welfare ordering is a preorder $\preceq$ on $\mathscr{V}^n$.

A social welfare ordering reflects the **collective preference ordering** regarding the set of possible allocations.

Collective utility function

A collective utility function is a function from $\mathscr{V}^n$ to $\mathscr{V}$.

A collective utility function represents a particular social welfare ordering.

Preference aggregation

Pareto Efficiency

Pareto-efficiency

Allocation $\overrightarrow{\pi}$ is **Pareto dominated** by allocation $\overrightarrow{\pi}'$ if $u_i(\overrightarrow{\pi}) \leq u_i(\overrightarrow{\pi}')$ for all agents $i \in \mathcal{N}$ and this inequality is strict in at least one case. An allocation $\overrightarrow{\pi}$ is **Pareto efficient** if there is no other feasible agreement $\overrightarrow{\pi}'$ such that $\overrightarrow{\pi}$ is Pareto dominated by $\overrightarrow{\pi}'$.

The idea goes back to Vilfredo Pareto (Italian economist, 1848–1923).
Discussion:

- Pareto efficiency is very often considered a minimum requirement for any agreement/allocation (encodes the notion of efficiency). It is a very weak criterion.
- Only the ordinal content of preferences is needed to check Pareto efficiency (no preference intensity, no interpersonal comparison).

Unanimity

A Social Welfare Ordering $\preceq$ or a collective utility function g satisfies the **unanimity principle** is and only if it is compatible with the Pareto ordering :

$$\overrightarrow{\pi} \text{ Pareto-dominates } \overrightarrow{\pi}' \Rightarrow \overrightarrow{\pi} \succ \overrightarrow{\pi}'.$$

Anonimity

A SWO $\preceq$ satisfies **anonimity** if it all agents are treated equally, no matter what there identities are.

Anonimity

A SWO $\preceq$ satisfies **anonimity** if it all agents are treated equally, no matter what there identities are.

Anonimity

For all permutation σ of $[\![1, n]\!]$, and all $\overrightarrow{\pi}$,

$$\langle \pi_1, \dots \pi_n \rangle \sim \langle \pi_{\sigma(1)}, \dots \pi_{\sigma(n)} \rangle$$

IUA

Independence of Unconcerned Agents

A SWO $\preceq$ satisfies the **Independence of Unconcerned Agents** if and only if for every quadruple $(\overrightarrow{u}, \overrightarrow{v}, \overrightarrow{u}', \overrightarrow{v}')$ such that:

- for a given agent i : $u_i = v_i$ and $u'_i = v'_i$,
- for every agent $k \neq i$: $u_k = u'_k$, $v_k = v'_k$,

we have: $\overrightarrow{u} \preceq \overrightarrow{v} \Leftrightarrow \overrightarrow{u}' \preceq \overrightarrow{v}'$.

It means informally that every agent i that has **no interest** in the choice between two decisions can be safely ignored.

Fairness ?

What is exactly fairness (or equity) ?

Fairness ?

What is exactly fairness (or equity) ?

Fairness [Young, 1994]

By "equitable, I do not necessarily mean ethical or moral, but that which a given society considers to be appropriate to the need, status and contribution of [the society's] various members."

Fairness ?

What is exactly fairness (or equity) ?

Fairness [Young, 1994]

By "equitable, I do not necessarily mean ethical or moral, but that which a given society considers to be appropriate to the need, status and contribution of [the society's] various members."

Aristotle and the theory of distributive justice (*Nicomachean Ethics, Book V*): **equal treatements of equals, unequal treatements of unequals**

Young, H. P. (1994).
Equity in Theory and Practice.
Princeton University Press.

Distributive justice

Story: The flute and the four children
Four principles of distributive justice from Aristotle (*Nicomachean Ethics, Book V*) – see [Moulin, 2003]:

Moulin, H. (2003).
Fair Division and Collective Welfare.
MIT Press.

Distributive justice

Story: The flute and the four children
Four principles of distributive justice from Aristotle (*Nicomachean Ethics, Book V*) – see [Moulin, 2003]:

- compensation $\rightsquigarrow$ ex-post equality

Moulin, H. (2003).
Fair Division and Collective Welfare.
MIT Press.

Distributive justice

Story: The flute and the four children
Four principles of distributive justice from Aristotle (*Nicomachean Ethics, Book V*) – see [Moulin, 2003]:

- compensation $\rightsquigarrow$ ex-post equality
- merits

Moulin, H. (2003).
Fair Division and Collective Welfare.
MIT Press.

Distributive justice

Story: The flute and the four children
Four principles of distributive justice from Aristotle (*Nicomachean Ethics, Book V*) – see [Moulin, 2003]:

- compensation $\rightsquigarrow$ ex-post equality
- merits
- exogenous rights $\rightsquigarrow$ ex-ante (in)equality

Moulin, H. (2003).
Fair Division and Collective Welfare.
MIT Press.

Distributive justice

Story: The flute and the four children
Four principles of distributive justice from Aristotle (*Nicomachean Ethics, Book V*) – see [Moulin, 2003]:

- compensation $\rightsquigarrow$ ex-post equality
- merits
- exogenous rights $\rightsquigarrow$ ex-ante (in)equality
- fitness.

Moulin, H. (2003).
Fair Division and Collective Welfare.
MIT Press.

Distributive justice

Story: The flute and the four children
Four principles of distributive justice from Aristotle (*Nicomachean Ethics, Book V*) – see [Moulin, 2003]:

- compensation $\rightsquigarrow$ ex-post equality
- merits
- exogenous rights $\rightsquigarrow$ ex-ante (in)equality
- fitness.

Remark: Cardinal welfarism is unable to encode justice according to merits.

Moulin, H. (2003).
Fair Division and Collective Welfare.
MIT Press.

Distributive justice

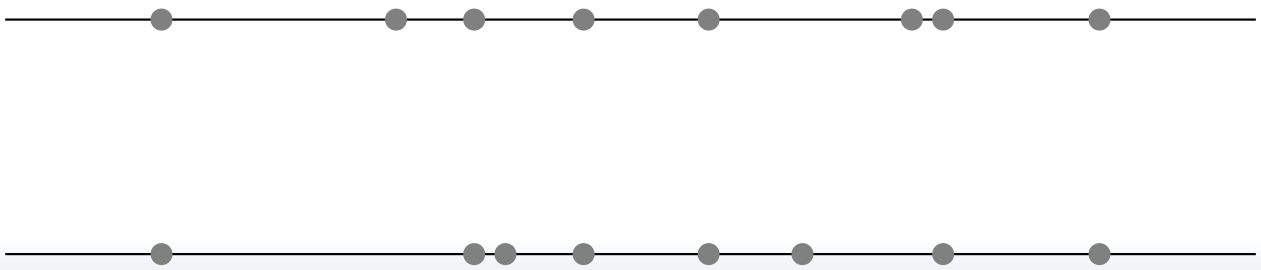
Story: The flute and the four children
Four principles of distributive justice from Aristotle (*Nicomachean Ethics, Book V*) – see [Moulin, 2003]:

- compensation $\rightsquigarrow$ ex-post equality
- merits
- exogenous rights $\rightsquigarrow$ ex-ante (in)equality
- fitness.

Moulin, H. (2003).
Fair Division and Collective Welfare.
MIT Press.

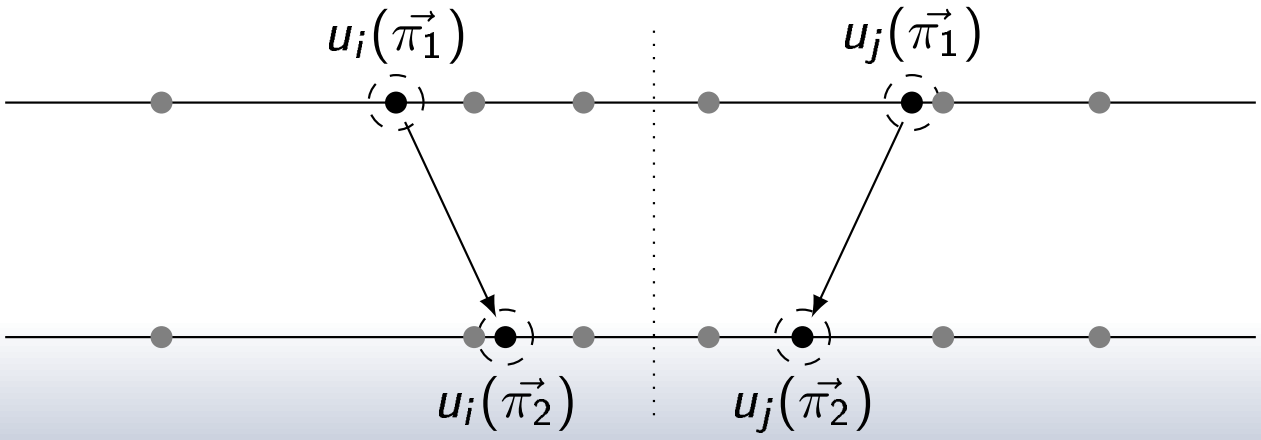
Pigou-Dalton principle

The idea is due to Arthur C. Pigou (British economist, 1877-1959) and Hugh Dalton (British economist and politician, 1887-1962).



Pigou-Dalton principle

The idea is due to Arthur C. Pigou (British economist, 1877-1959) and Hugh Dalton (British economist and politician, 1887-1962).



Pigou-Dalton principle

More formally, $\overrightarrow{u} \rightsquigarrow \overrightarrow{u}'$ **reduces inequalities** iff $\exists i \neq j$ such that:

- $u_i + u_j = u'_i + u'_j$ (sum conservation)
- $u_i < \{u'_i, u'_j\} < u_j$ (reduction of inequalities)
- $\forall j \in \mathcal{N} \setminus \{i, j\}$, $u_k = u'_k$.

Pigou-Dalton principle

More formally, $\overrightarrow{u} \rightsquigarrow \overrightarrow{u}'$ **reduces inequalities** iff $\exists i \neq j$ such that:

- $u_i + u_j = u'_i + u'_j$ (sum conservation)
- $u_i < \{u'_i, u'_j\} < u_j$ (reduction of inequalities)
- $\forall j \in \mathcal{N} \setminus \{i, j\}$, $u_k = u'_k$.

A SWO satisfies the Pigou-Dalton principle iff:

$$\overrightarrow{u} \rightsquigarrow \overrightarrow{u}' \text{ reduces inequalities } \Rightarrow \overrightarrow{u} \prec \overrightarrow{u}'.$$

Lorenz-curve

A first way to "measure" the inequalities is to use the **Lorenz curve**

Ordered utility vector

For any $\overrightarrow{u} \in \mathbb{R}^n$, the ordered utility vector $\overrightarrow{u}^{\uparrow}$ is defined as the vector we obtain when we rearrange the elements of u in increasing order.

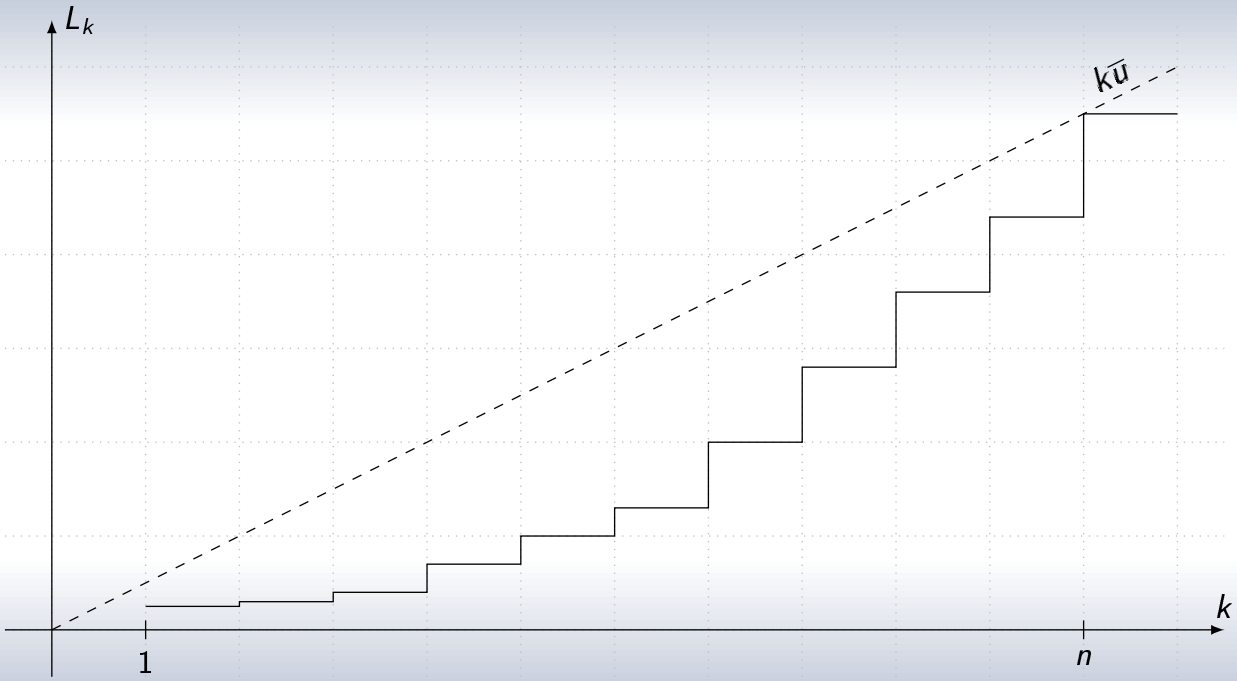
Example: $\overrightarrow{u} = \langle 5, 20, 0 \rangle \rightsquigarrow \overrightarrow{u}^{\uparrow} = \langle 0, 5, 20 \rangle$

Lorenz curve

The **Lorenz curve** of a utility profile $\overrightarrow{u}$ is the following vector:

$$\overrightarrow{L(u)} = \langle u_1^{\uparrow}, \ldots, \sum_{k=1}^{i} u_k^{\uparrow}, \ldots, \sum_{k=1}^{n} u_k^{\uparrow} \rangle.$$

Lorenz-curve



Lorenz, Pareto and inequalities

A “Lorenz-improvement” is a Pareto-improvement over the Lorenz curve.

- Every Pareto improvement is also a Lorenz improvement.
- Every inequality reduction is also a Lorenz improvement.

Lorenz, Pareto and inequalities

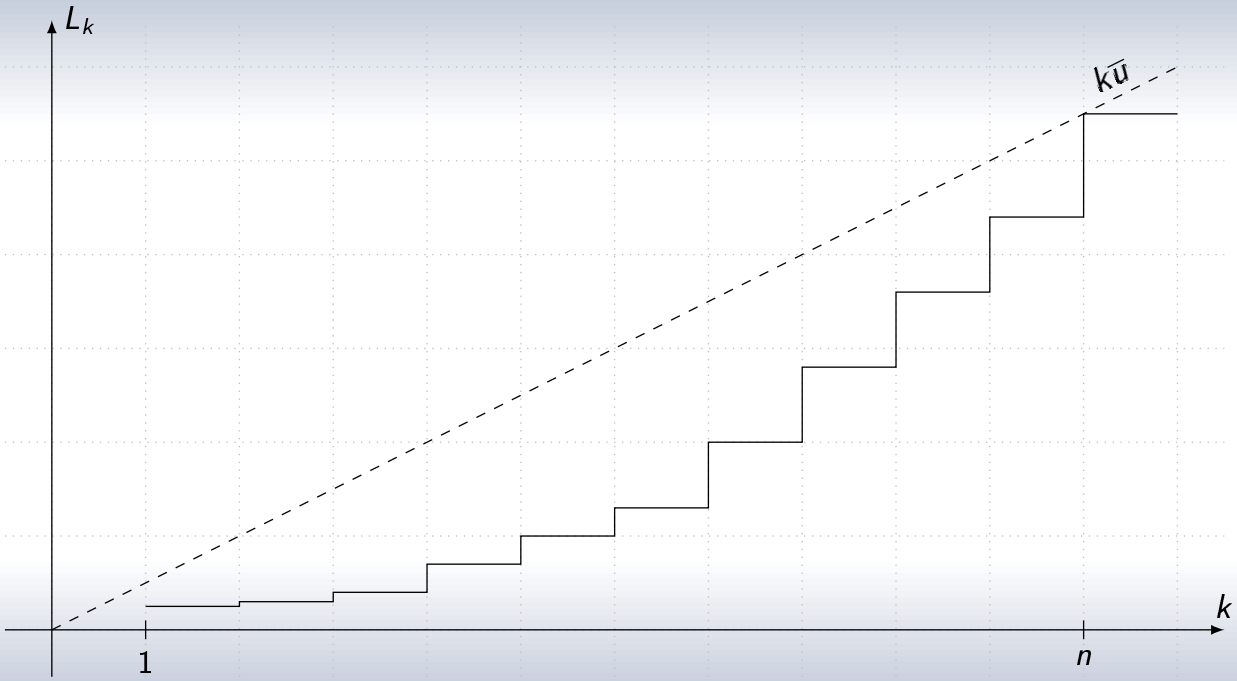
A "Lorenz-improvement" is a Pareto-improvement over the Lorenz curve.

- Every Pareto improvement is also a Lorenz improvement.
- Every inequality reduction is also a Lorenz improvement.

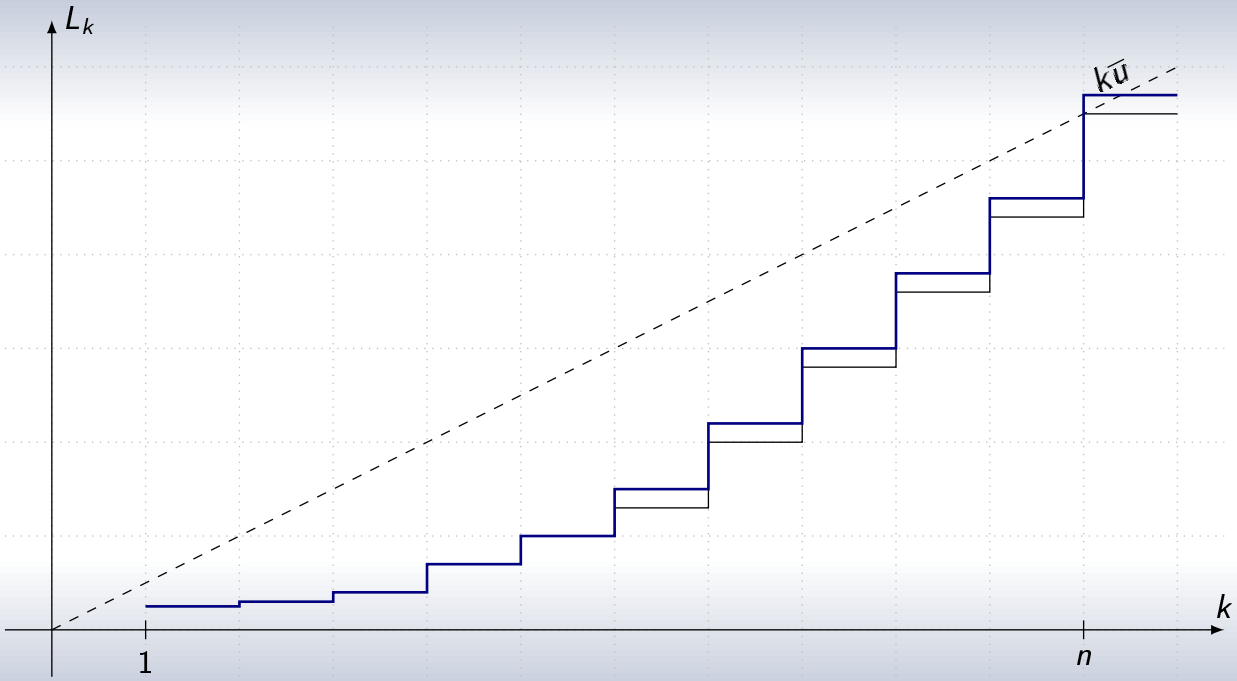
Result

Let $\overrightarrow{u}$ and $\overrightarrow{v}$ be two profiles. Then $\overrightarrow{L(u)}$ Pareto-dominates $\overrightarrow{L(v)} \Leftrightarrow \overrightarrow{v} \rightsquigarrow \cdots \rightsquigarrow \overrightarrow{u}$ is a sequence of Pareto improvements or inequality reducing moves.

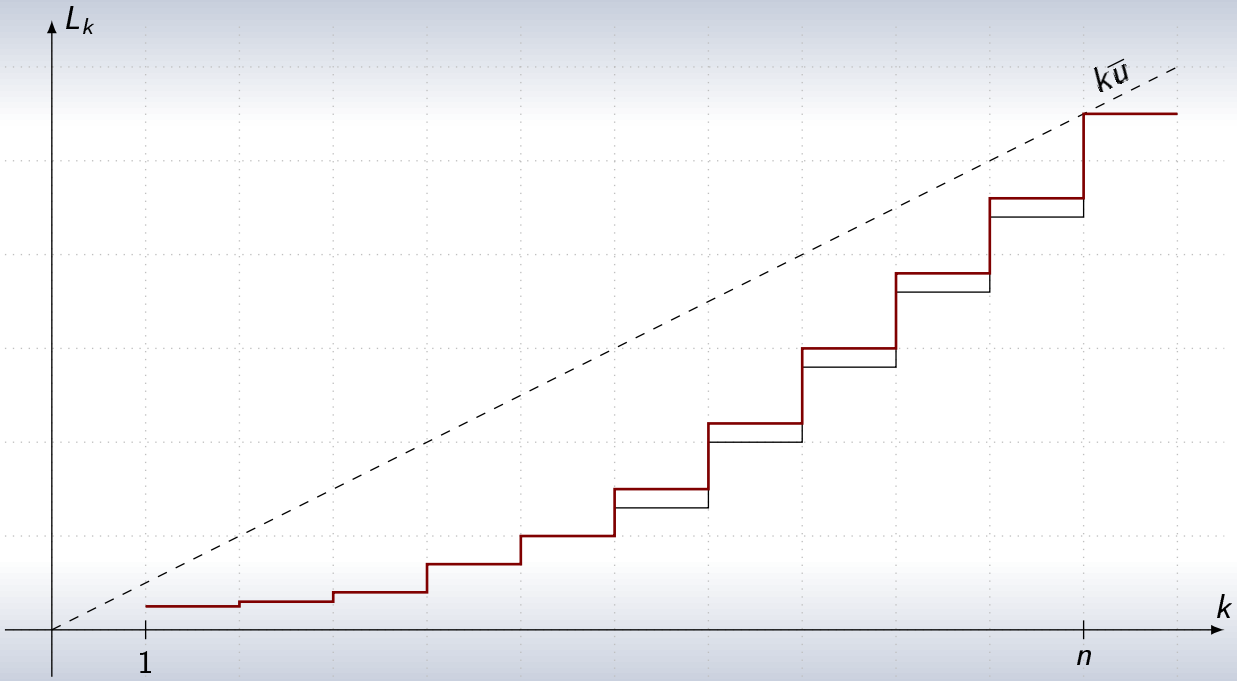
Lorenz-curve



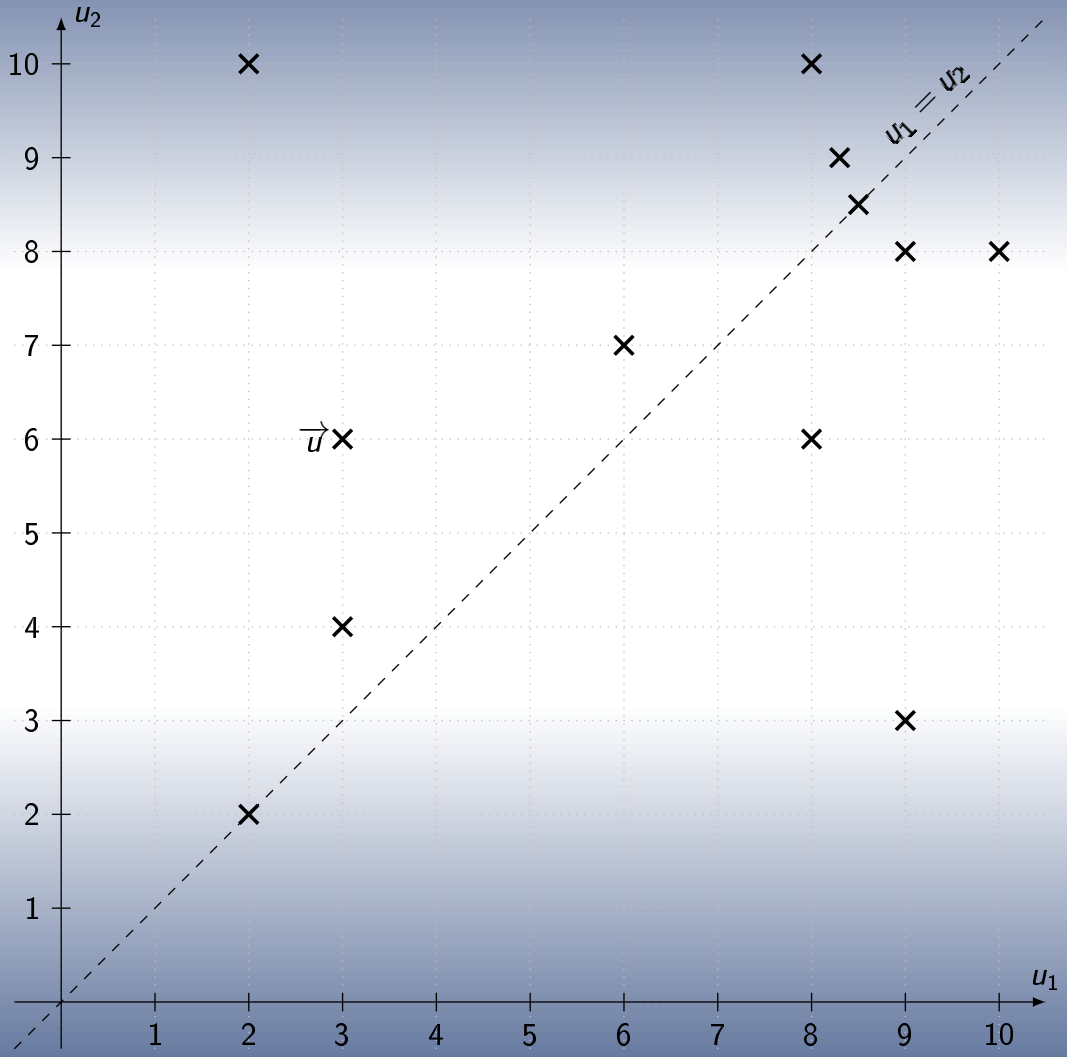
Lorenz-curve



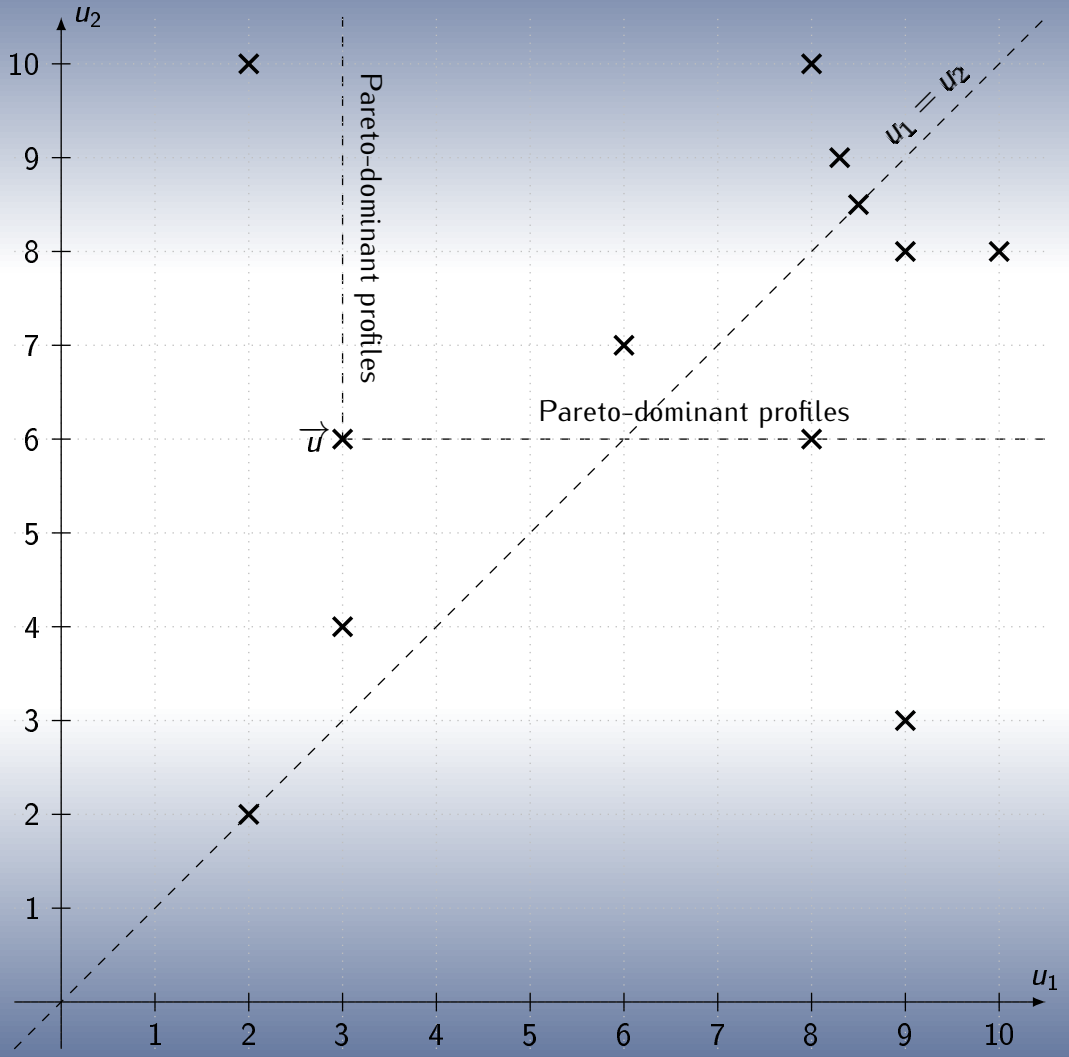
Lorenz-curve



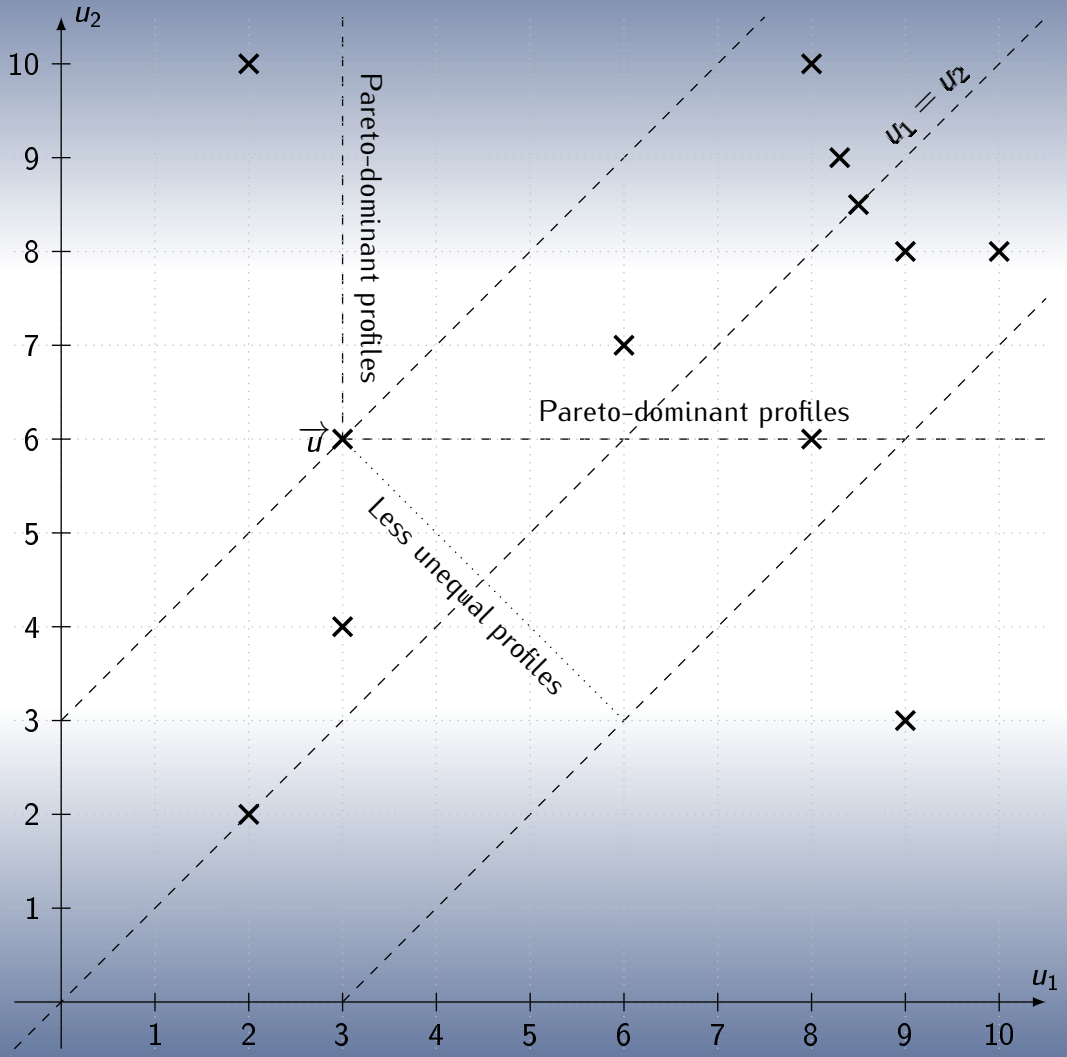
Lorenz domination



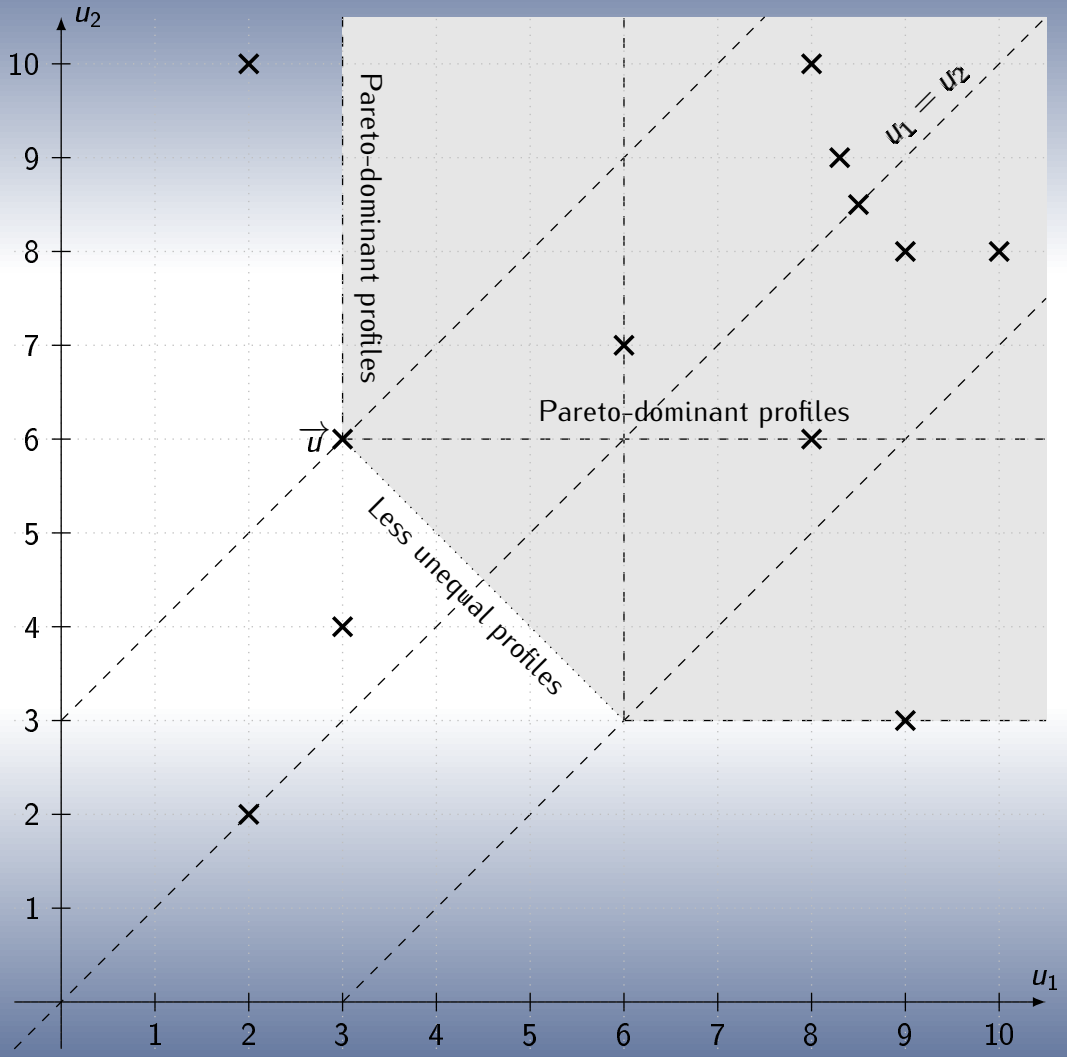
Lorenz domination



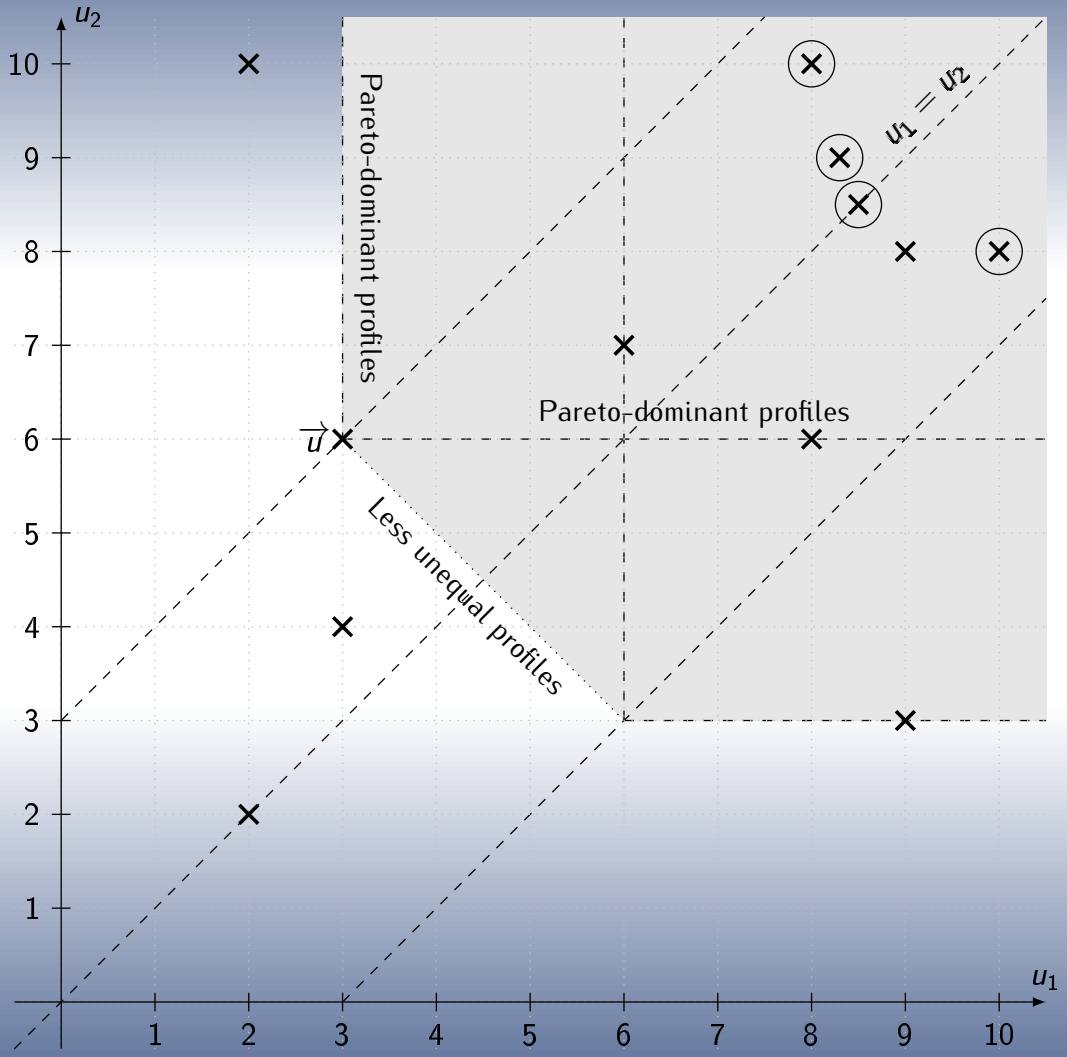
Lorenz domination



Lorenz domination



Lorenz domination



Inequality indexes

A more systematic way to **measure inequalities** ...

For a given SWO $\preceq$, and a profile $\overrightarrow{u}$, we will write $\varepsilon(\overrightarrow{u})$ the value such that $\langle \varepsilon(\overrightarrow{u}), \ldots, \varepsilon(\overrightarrow{u}) \rangle \sim \overrightarrow{u}$, and $\overline{u}$ the mean of $\overrightarrow{u}$.

Inequality index

The inequality index associated to $\preceq$ is:

$$J_{\preceq}(\overrightarrow{u}) = 1 - \frac{\varepsilon(\overrightarrow{u})}{\overline{u}}.$$

Inequality indexes

A more systematic way to **measure inequalities** ...

For a given SWO $\preceq$, and a profile $\overrightarrow{u}$, we will write $\varepsilon(\overrightarrow{u})$ the value such that $\langle \varepsilon(\overrightarrow{u}), \ldots, \varepsilon(\overrightarrow{u}) \rangle \sim \overrightarrow{u}$, and $\overline{u}$ the mean of $\overrightarrow{u}$.

Inequality index

The inequality index associated to $\preceq$ is:

$$J_{\preceq}(\overrightarrow{u}) = 1 - \frac{\varepsilon(\overrightarrow{u})}{\overline{u}}.$$

Remarks:

- $J(\overrightarrow{u}) \leq 1$.
- $J(\overrightarrow{u}) \geq 0$ if $\preceq$ reduces inequalities ($J(\overrightarrow{u}) = 0$ if and only if $u_i = u_j$, $\forall i, j$).

Inequality indexes: examples

Atkinson inequality indexes

$$\begin{cases} J_q(\overrightarrow{u}) = 1 - \left(\frac{1}{n} \sum_{i=1}^{n} \left(\frac{u_i}{\overline{u}} \right)^q \right)^{\frac{1}{q}}, & 0 < q < 1 \text{ or } q < 0 \\ J_0(\overrightarrow{u}) = 1 - \left(\prod_{i=1}^{n} \frac{u_i}{\overline{u}} \right)^{\frac{1}{n}}. \end{cases}$$

(Inequality index based on generalized averages – see later)

Inequality indexes: examples

Gini index

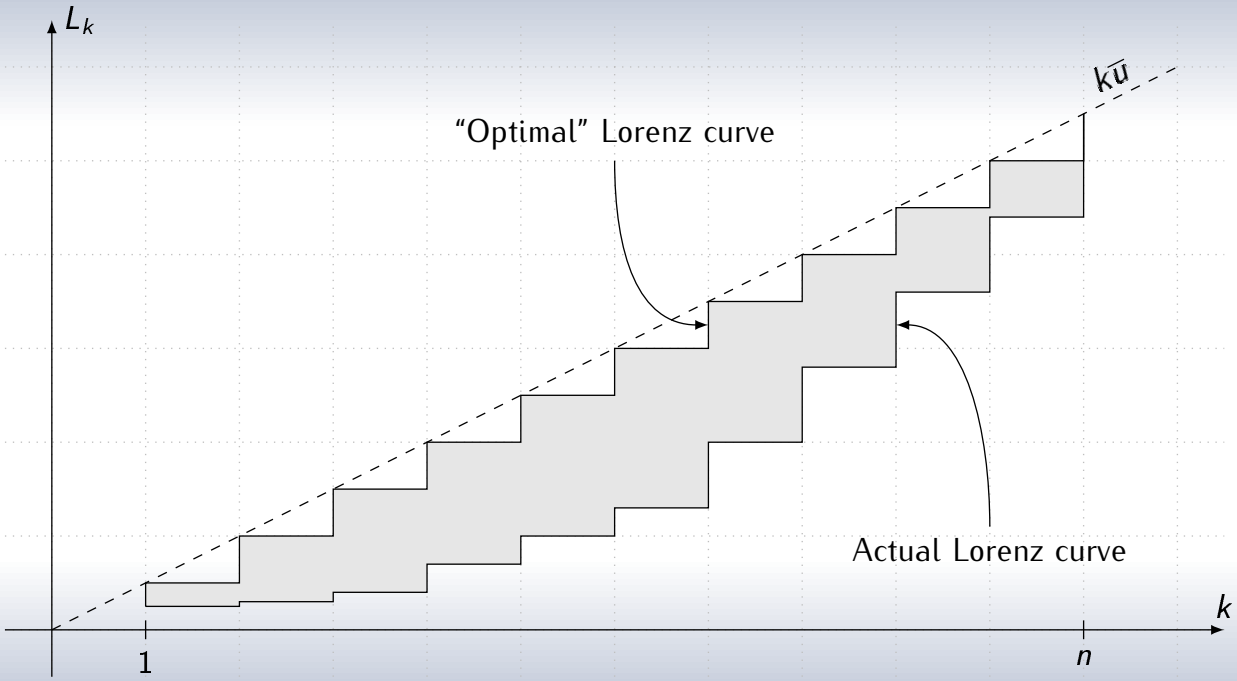
$$G(\overrightarrow{u}) = \frac{\sum_{k=1}^{n}(k\overline{u} - L(\overrightarrow{u})_k)}{\frac{n}{2}\sum_{i=1}^{n} u_i} = 1 - \frac{1}{n^2\overline{u}}\left(\sum_{k=1}^{n}(2(n-k)+1)u_k^{\uparrow}\right)$$

$$= \frac{1}{2n^2\overline{u}} \sum_{1\leq i,j\leq n} |u_i - u_j|.$$

Three interpretations:

- Area between two Lorenz curves (see next slide)
- Collective Utility Function based definition (OWA – see later)
- Differential utilities

Gini index: interpretation



Fair-share test

The most prominent formalization of fairness is **ex-post equality**.

The first property encoding ex-post equality is the **fair-share test**

Fair share test

$\overrightarrow{\pi}$ satisfies the fair-share test iff for all i, $u_i(\pi_i) \geq \hat{u}_i/n$, with $\hat{u}_i$ the maximal utility obtained by i if she were alone.

Fair-share test

The most prominent formalization of fairness is **ex-post equality**.

The first property encoding ex-post equality is the **fair-share test**

Fair share test

$\overrightarrow{\pi}$ satisfies the fair-share test iff for all i, $u_i(\pi_i) \geq \hat{u}_i/n$, with $\hat{u}_i$ the maximal utility obtained by i if she were alone.

A SWO **guarantees fair-share** if and only if there is at least one undominated allocation that satisfies the fair share test.

Envy-freeness

- First defined in [Tinbergen, 1953] $\rightsquigarrow$ no one in a society wishes to change her place with somebody else's place (strong vision of envy-freeness)
- Weaker (allocation-based) vision by [Foley, 1967]

Envy-free allocation

An allocation $\vec{\pi}$ is envy-free if and only if $\forall i \neq j$, $u_i(\pi_i) \geq u_i(\pi_j)$.

Foley, D. K. (1967).
Resource allocation and the public sector.
Yale Economic Essays, 7(1):45–98.

Tinbergen, J. (1953).
Redeljke Inkomensverdeling.
N. V. DeGulden Pers., Haarlem.

Envy-freeness

Remarks:

- Specific to allocation problems.
- Does not require interpersonal comparison of utilities.
- Encodes more a notion of "social peace" than ex-post fairness.
- Encodes a kind of **exogeneity**, based on endogenous preferences (no externalities).
- Envy-freeness alone is not sufficient $\rightsquigarrow$ needs an efficiency criterion as well.

Envy-freeness

Remarks:

- Specific to allocation problems.
- Does not require interpersonal comparison of utilities.
- Encodes more a notion of "social peace" than ex-post fairness.
- Encodes a kind of **exogeneity**, based on endogenous preferences (no externalities).
- Envy-freeness alone is not sufficient $\rightsquigarrow$ needs an efficiency criterion as well.
  - Complete allocation
  - Pareto-efficiency
  - maximizing a given SWO

Preference aggregation

Social Welfare Orderings

Now that we have some **principles** and **formal properties** ...

...How can we design social welfare orderings and collective utility functions that satisfy (most of) these properties ?

Classical utilitarian SWO

One approach to social welfare is to try to **maximise overall profit**.

This is known as **classical utilitarianism** (advocated, amongst others, by Jeremy Bentham (1748–1832), British philosopher, and John Harsanyi (1920-2000), Hungarian-Australian-American economist).

Classical utilitarian SWO

One approach to social welfare is to try to **maximise overall profit**.

This is known as **classical utilitarianism** (advocated, amongst others, by Jeremy Bentham (1748–1832), British philosopher, and John Harsanyi (1920-2000), Hungarian-Australian-American economist).

Classical utilitarianism

The classical utilitarian CUF is defined as $g^\star : \overrightarrow{u} \mapsto \sum_{i=1}^{n} u_i$.

- Conveys the sum-fitness principle (resource goes to who makes the best use of it).
- Indifferent to inequalities $\rightsquigarrow$ can lead to huge inequalities between the agents.

Classical utilitarian SWO

One approach to social welfare is to try to **maximise overall profit**.

This is known as **classical utilitarianism** (advocated, amongst others, by Jeremy Bentham (1748–1832), British philosopher, and John Harsanyi (1920-2000), Hungarian-Australian-American economist).

Classical utilitarianism

The classical utilitarian CUF is defined as $g^\star : \overrightarrow{u} \mapsto \sum_{i=1}^{n} u_i$.

- Conveys the sum-fitness principle (resource goes to who makes the best use of it).
- Indifferent to inequalities $\rightsquigarrow$ can lead to huge inequalities between the agents.

Example: $\langle 10, 10, 10, 10 \rangle \preceq \langle 41, 0, 0, 0 \rangle$

Egalitarian Social Welfare

Egalitarianism [Rawls]

The egalitarian CUF is defined as $g^e : \overrightarrow{u} \mapsto \min_{i=1}^{n} u_i$.

Egalitarian Social Welfare

Egalitarianism [Rawls]

The egalitarian CUF is defined as $g^e : \overrightarrow{u} \mapsto \min_{i=1}^{n} u_i$.

- Conveys the compensation principle: the least well-off must be made as well-off as possible (justice according to needs) $\rightsquigarrow$ tends to equalize the utility profile.
- The egalitarian variant of welfare economics is inspired by the work of John Rawls (American philosopher, 1921–2002) [Rawls, 1971] and has been formally developed, amongst others, by Amartya Sen since the 1970s (Nobel Prize in Economic Sciences in 1998).

Example: $\langle 10, 10, 10, 10 \rangle \succeq \langle 41, 0, 0, 0 \rangle$

Utilitarianism versus Egalitarianism

- In the MAS literature the utilitarian viewpoint (that is, social welfare = sum of individual utilities) is often taken for granted.
- In philosophy, economics, political science not.
- John Rawls' **"veil of ignorance"** (A Theory of Justice, 1971): Without knowing what your position in society (class, race, sex, ...) will be, what kind of society would you choose to live in?
- Reformulating the veil of ignorance for multiagent systems: If you were to send a software agent into an artificial society to negotiate on your behalf, what would you consider acceptable principles for that society to operate by?
- **Conclusion**: worthwhile to investigate egalitarian (and other) social principles also in the context of multiagent systems.

Egalitarianism and drowning effect

Egalitarianism can lead to non Pareto-efficient decisions (drowning effect).

Egalitarianism and drowning effect

Egalitarianism can lead to non Pareto-efficient decisions (drowning effect).

Egalitarian SWO and Pareto-efficiency

$\langle 1, 1, 1, 1 \rangle \sim \langle 1000, 1, 1000, 1000 \rangle$

The Leximin Ordering (1)

We now introduce an SWO that may be regarded as a refinement of the SWO induced by the egalitarian CUF.

Leximin egalitarianism [Sen, 1970; Kolm, 1972]

$\overrightarrow{u} \succ_{leximin} \overrightarrow{v} \Leftrightarrow \exists k$ such that $\forall i \leq k,\ u_i^{\uparrow} = v_i^{\uparrow}$ and $u_{k+1}^{\uparrow} > v_{k+1}^{\uparrow}$.
This is a lexicographical comparison over sorted vectors.

Remark: This is a **social welfare ordering**, but as we shall see later, under reasonable conditions, it can be represented by a collective utility function.

The Leximin Ordering (2)

Perform a leximin comparison...

Two vectors to compare: $\overrightarrow{u} = \langle 4, 10, 3, 5 \rangle$ and $\overrightarrow{v} = \langle 4, 3, 6, 6 \rangle$.

- We sort the two vectors: $\begin{cases} \overrightarrow{u}^{\uparrow} = \langle 3, 4, 5, 10 \rangle \\ \overrightarrow{v}^{\uparrow} = \langle 3, 4, 6, 6 \rangle \end{cases}$
- We lexicographically sort the ordered vectors : $\overrightarrow{u}^{\uparrow} \prec_{lexico} \overrightarrow{v}^{\uparrow}$

Features:

- both refines the egalitarian SWO and the Pareto relation $\rightsquigarrow$ inherits of the fairness features of egalitarianism, while overcoming drowning effect.
- the only one SWO that both respects **reduction of inequalities**, and **independence of the common utility pace**.

Leximin SWO leximin and Pareto-efficiency

$\langle 1, 1, 1, 1 \rangle \prec \langle 1000, 1, 1000, 1000 \rangle$ (the second value of the two vectors is discriminating).

Nash Product

Egalitarianism [Rawls]

The egalitarian CUF is defined as $g^N : \overrightarrow{u} \mapsto \prod_{i=1}^{n} u_i$.

This is a useful measure of social welfare as long as all utility functions can be assumed to be positive. Named after John F. Nash (Nobel Prize in Economic Sciences in 1994; Academy Award in 2001).

Features:

- compromise between utilitarian and egalitarian CUF: favours increases in overall utility, while reducing inequalities.
- Independent of the individual utility scales.

Generalized averages

Generalized averages

$$\forall p \in \mathbb{R}\ g^{(p)}(\overrightarrow{u}) = \begin{cases} sign(p) \cdot \left(\sum_{i=1}^{n} u_i^p \right)^{\frac{1}{p}} & \text{for } p \neq 0 \\ \sum_{i=1}^{n} \log u_i & \text{for } p = 0 \end{cases}$$

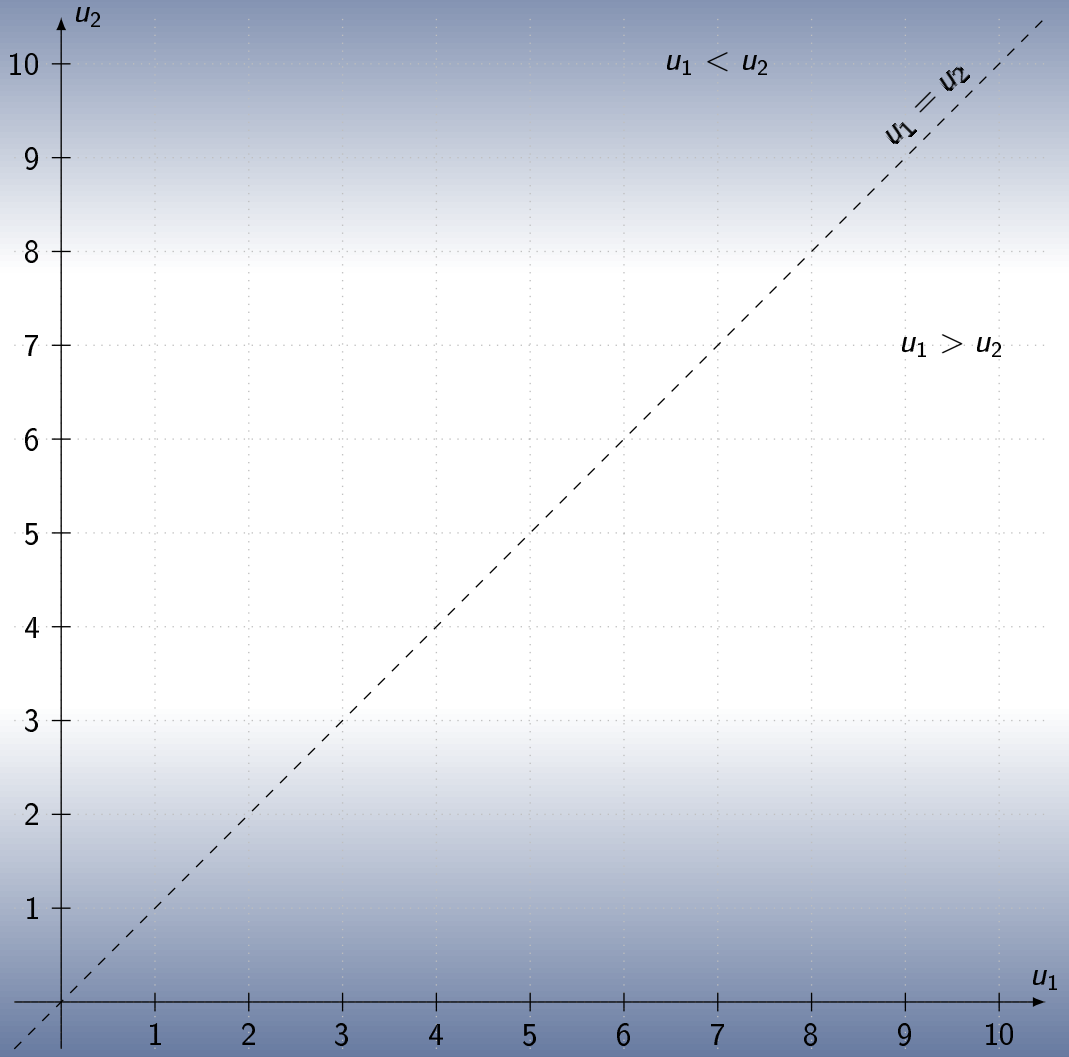
- $p > 1 \Leftrightarrow g^{(p)}$ strictly increases inequalities.
- $g^{(1)} = g^{\star}$ the utilitarian CUF (indifferent to inequalities).
- $g^{(0)} = g^{N}$ the Nash CUF.
- $p \to -\infty$ the leximin CUF.

Generalized averages

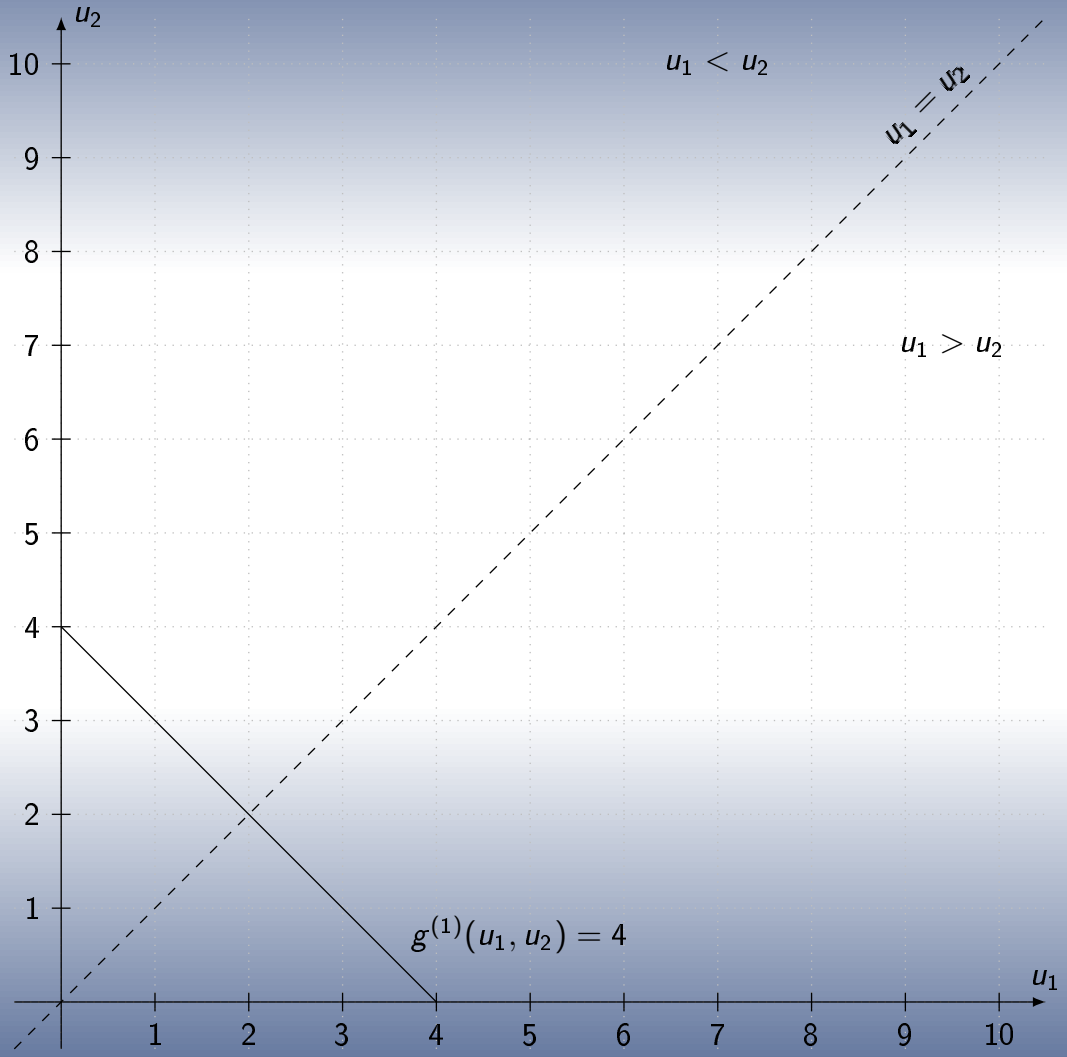
Some nice features...

- If the set of possible utility profiles is finite, then there is a p such that for all $p' < p$, $g^{(p')}$ represents the leximin SWO.
- [Roberts, 1980]: every **anonymous**, **continuous**, **separable** (*i.e* satisfying Independence of Unconcerned Agents) social welfare ordering can be represented by a generalized average.

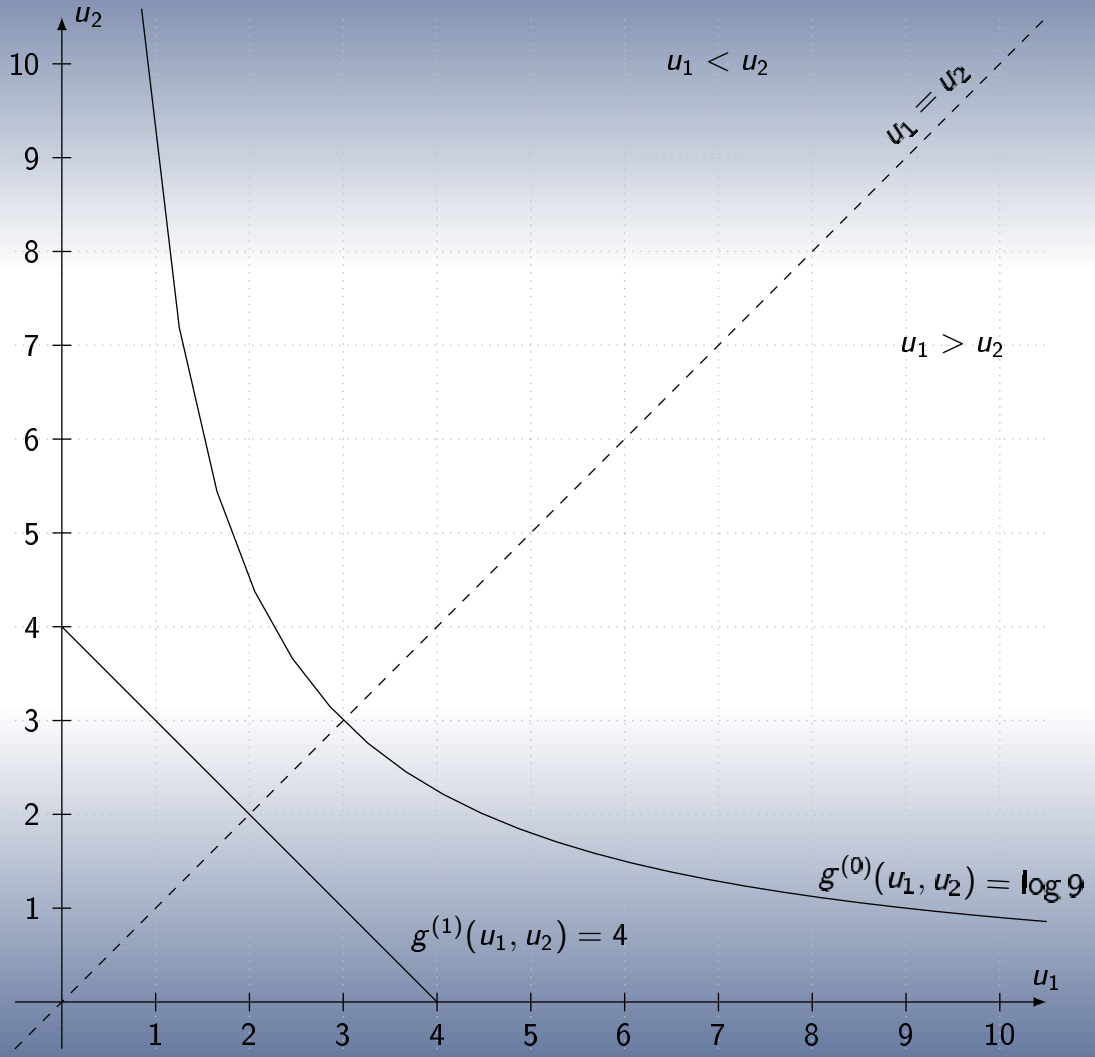
Generalized Averages



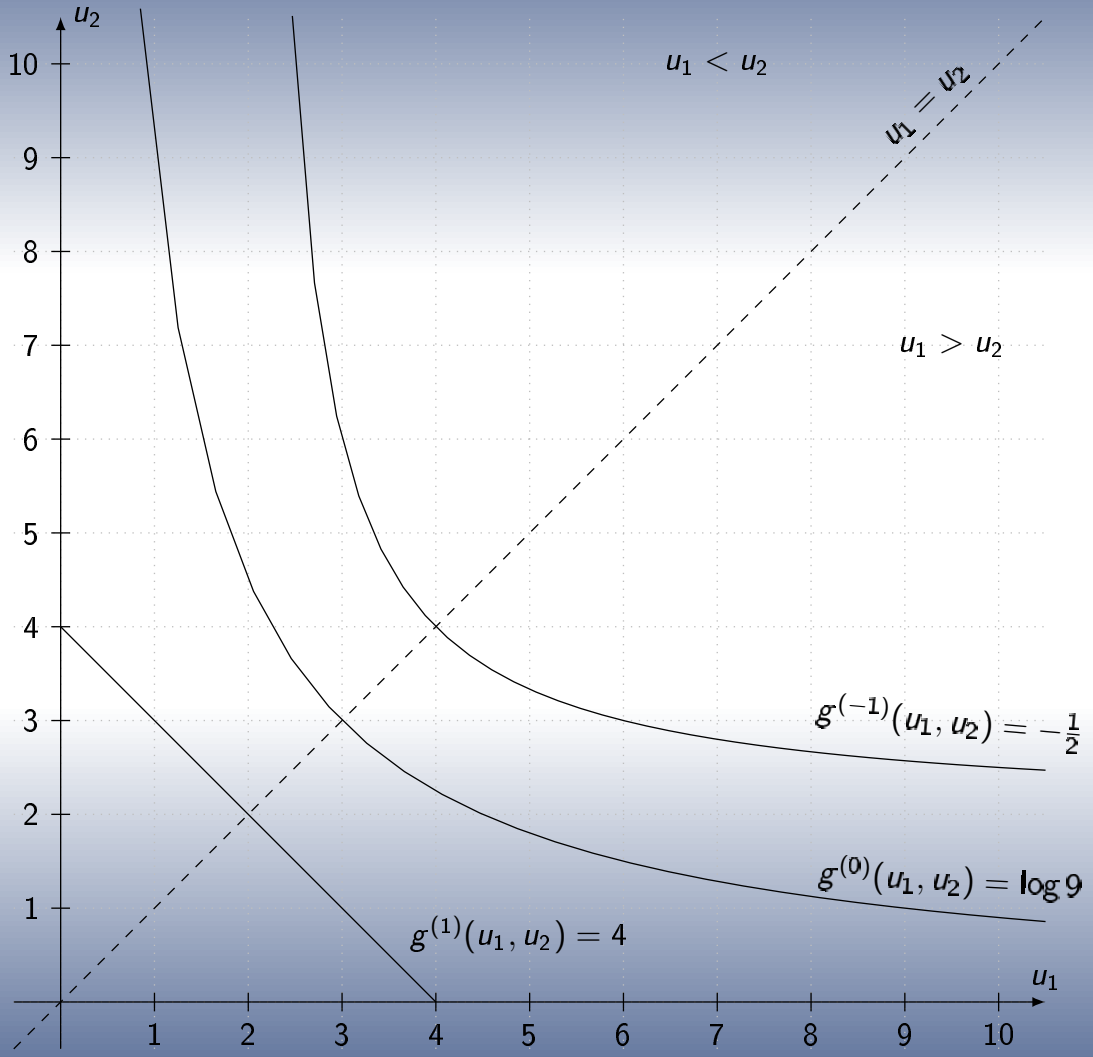
Generalized Averages



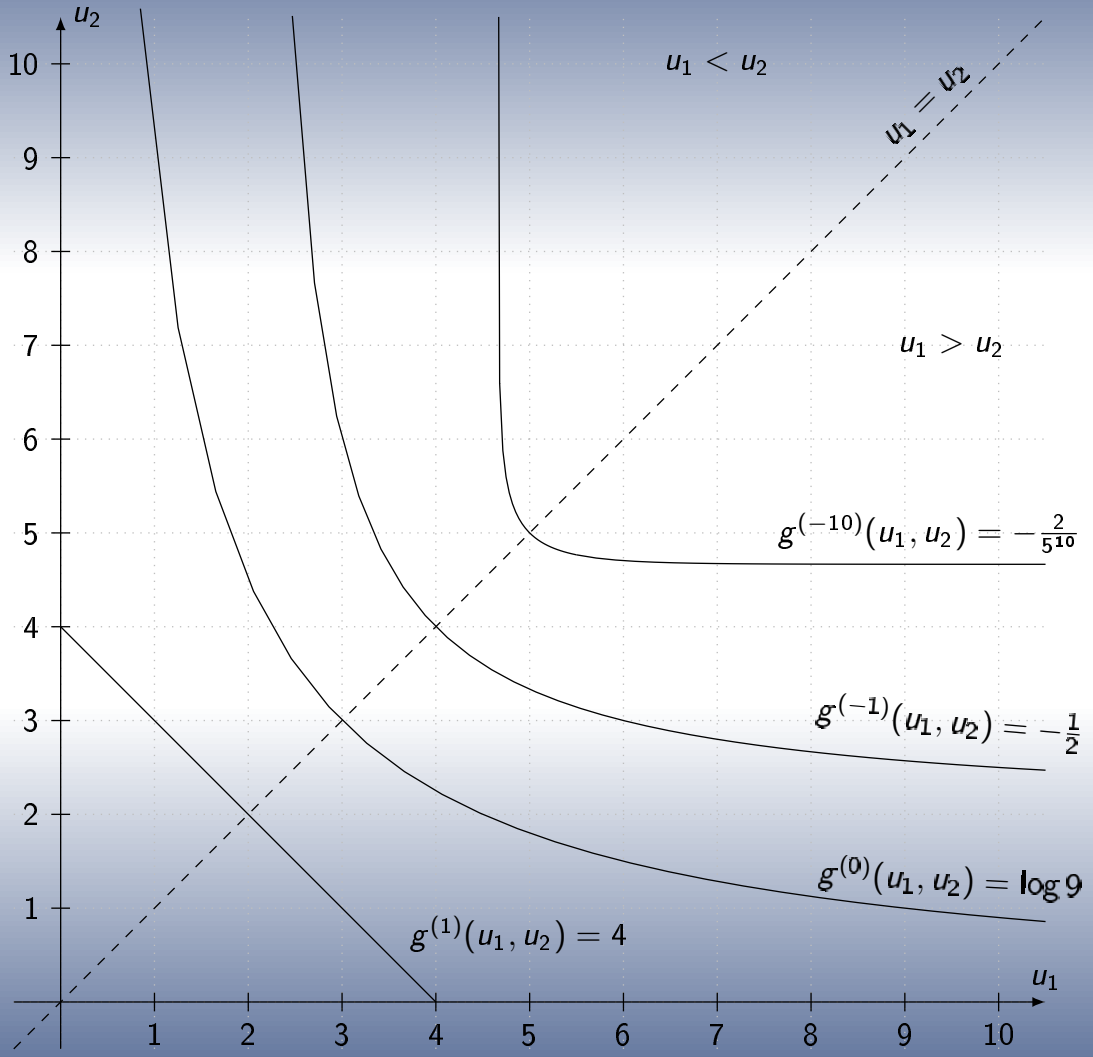
Generalized Averages



Generalized Averages



Generalized Averages



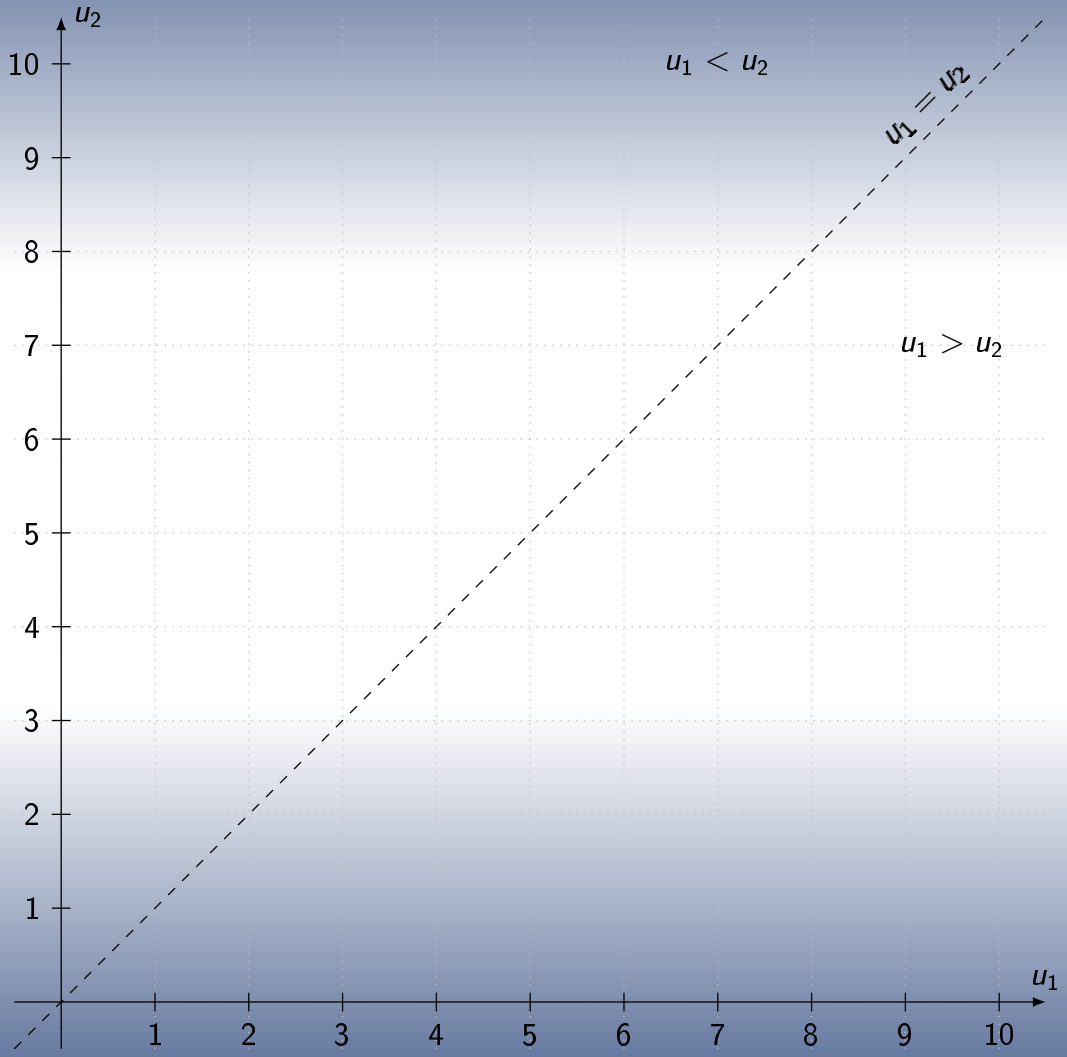
Ordered Weighted Averages

OWA family

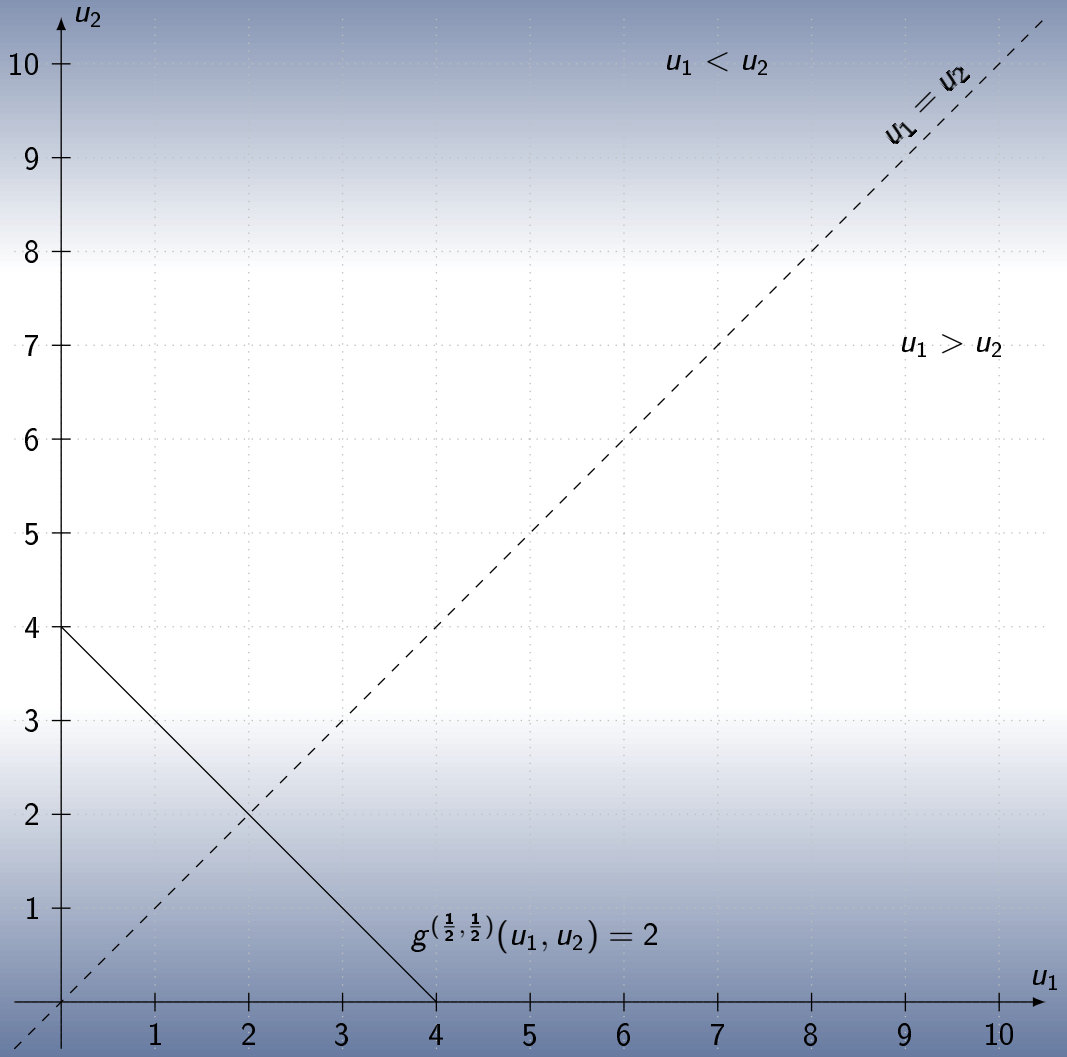
$$g^{\overrightarrow{w}} = \sum_{i=1}^{n} w_i \times u_i^{\uparrow}, \text{ with } \overrightarrow{w} \in [0,1]^n \text{ and } \sum_{i=1}^{n} w_i = 1.$$

- $g^{\langle 1/n,\ldots,1/n\rangle}$ classical utilitarian CUF.
- $g^{\langle 1,0,\ldots,0\rangle}$ egalitarian CUF.
- $g^{\overrightarrow{w}}$ satisfies inequality reduction $\Leftrightarrow w_1 > w_2 > \cdots > w_n$.
- If the set of utility profiles is finite, there is an OWA that represents the leximin SWO ($w_1 \gg w_2 \gg \cdots \gg w_n$).

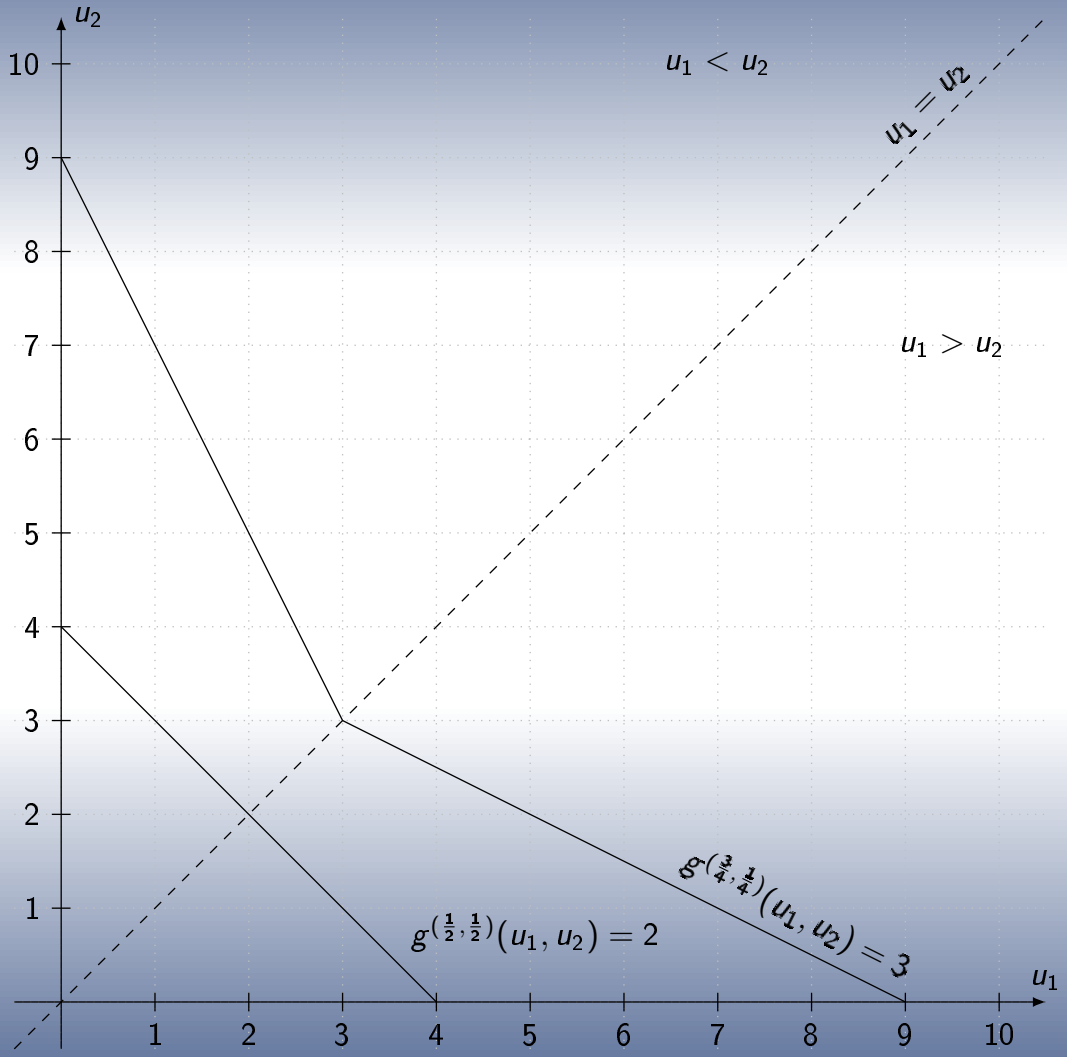
Ordered weighted averages



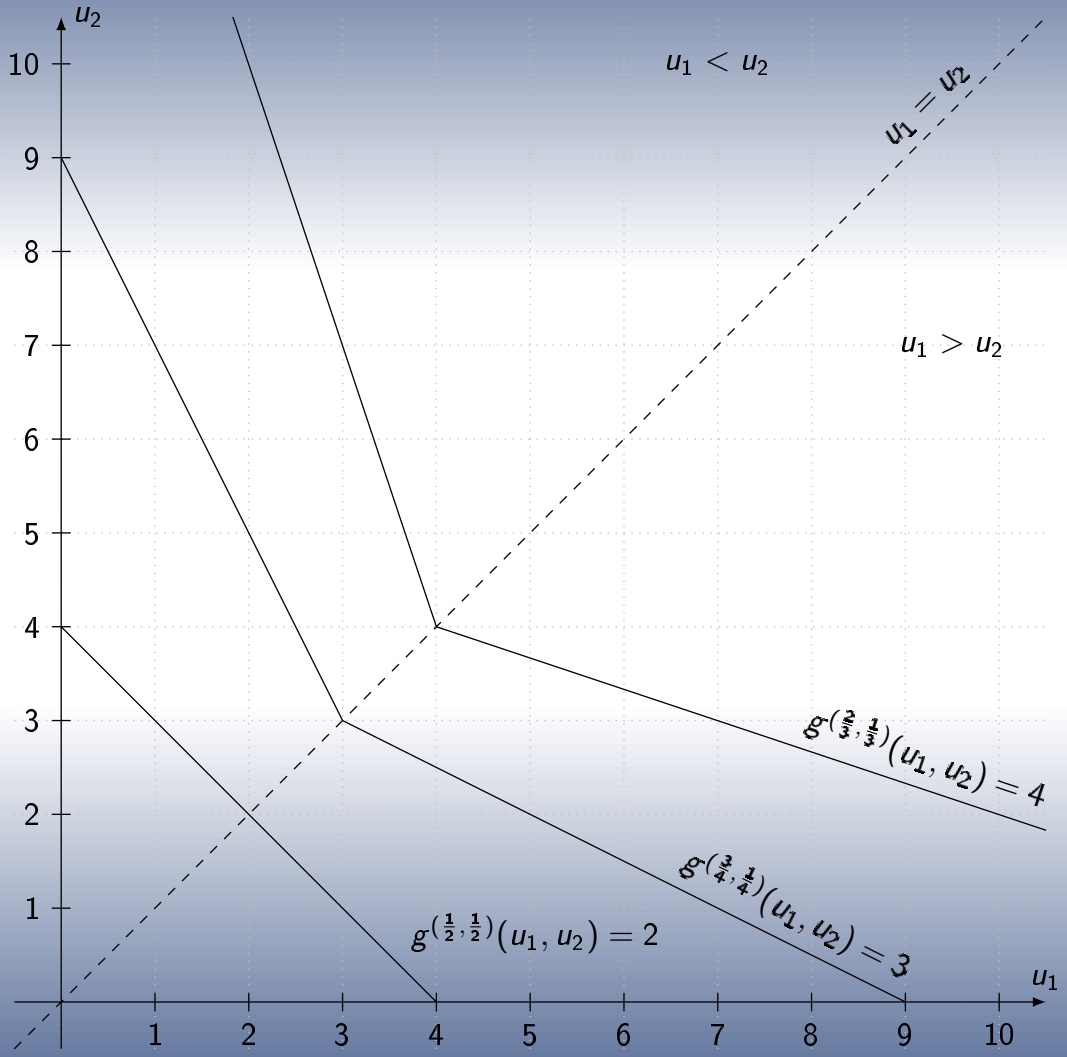
Ordered weighted averages



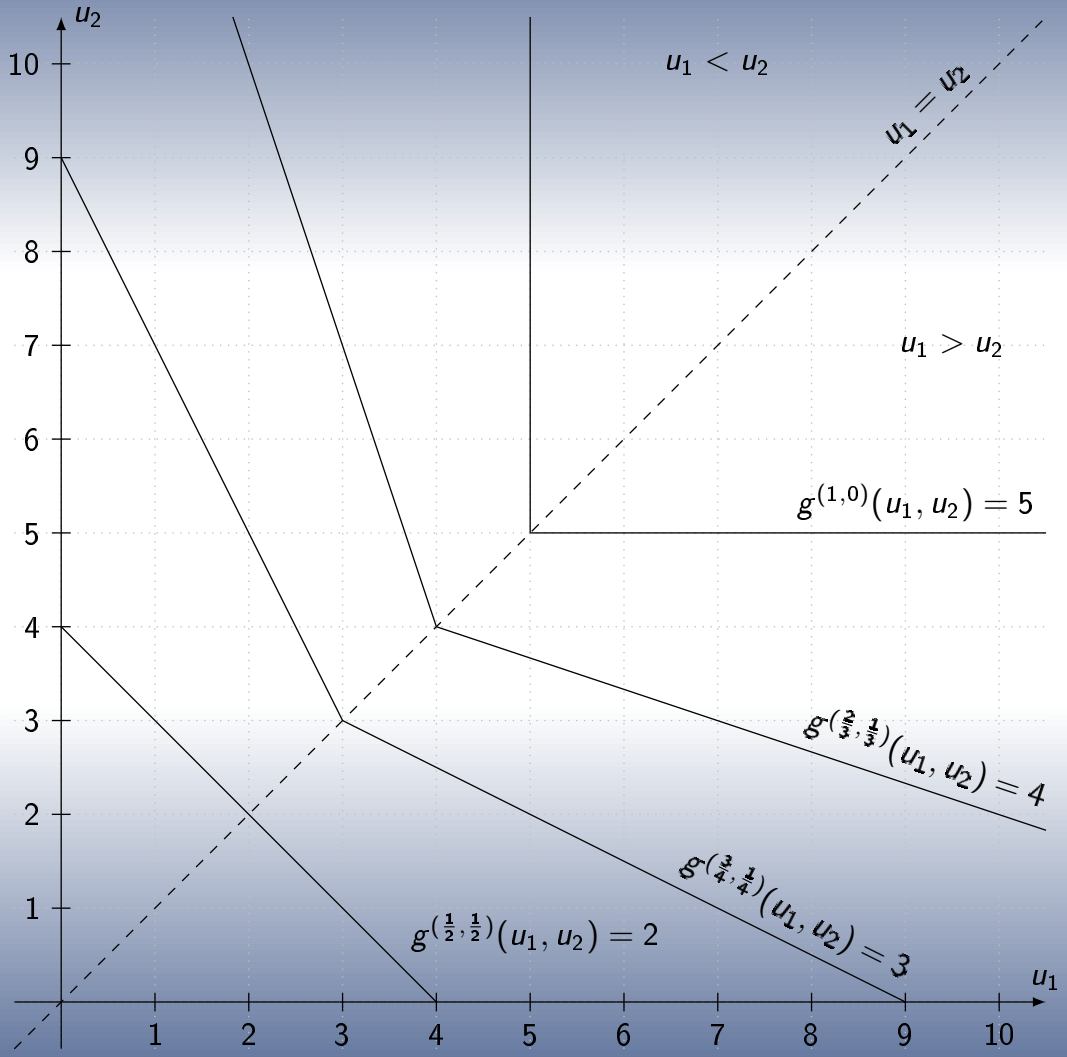
Ordered weighted averages



Ordered weighted averages



Ordered weighted averages



Rank Dictators

A special case of OWA.
The **k-rank dictator** CUF for $k \in \mathcal{N}$ is mapping utility vectors to the utility enjoyed by the k-poorest agent:

$$g^{d-k}(\overrightarrow{u}) = u_k^{\uparrow}$$

Interesting special cases:

- For $k = 1$ we obtain the **egalitarian** CUF.
- For $k = n$ we obtain an **elitist** CUF measuring social welfare in terms of the happiest agent.
- For $k = \lfloor \frac{n+1}{2} \rfloor$ we obtain the **median-rank-dictator** CUF.

Normalization issues

Most SWO require **interpersonal comparison of utilities** ...
...But in general, how can we ensure that utility "10" for agent 1 means the same as utility "10" for agent 2 ?

Normalization issues

Most SWO require **interpersonal comparison of utilities** ...
...But in general, how can we ensure that utility "10" for agent 1 means the same as utility "10" for agent 2 ?

Solution 1: Impose a **common utility scale** (*e.g* money, fixed number of points, etc.)

Normalization issues

Most SWO require **interpersonal comparison of utilities** ...
...But in general, how can we ensure that utility "10" for agent 1 means the same as utility "10" for agent 2 ?

Solution 1: Impose a **common utility scale** (*e.g* money, fixed number of points, etc.)

Solution 2: **Normalize** utilities. Example: Kalai-Smorodinsky solution (divide individual utilities by "maximal utility if alone").

Preference aggregation

Conclusion on preference aggregation

- **Efficiency**: we require at least Pareto-efficient allocations.
- **Fairness**: two main (different) points of view
  - Reduction of inequalities (ex-post fairness) $\rightsquigarrow$ Pigou-Dalton, Lorenz, inequality indexes.
  - Envy-freeness (needs efficiency as well).
- **Classical SWO**: classical utilitarianism, and egalitarianism (leximin).
- **Compromises**: Nash, generalized averages, OWA.

Part 3

Preference representation

Preference representation

Back to individual preferences...

A resource allocation problem

- **Inputs**
  - A set $\mathscr{N}$ of **agents** expressing **preferences** on the resource using preorders $\succeq_i$ or utility functions u_i.
  - **The resource** $\rightsquigarrow$ a finite set $\mathscr{O}$ of indivisible objects.
  - **Some constraints** $\rightsquigarrow$ a finite set $\mathscr{C} \subset 2^{2^{\mathscr{O}^n}}$.
  - A **criterion** $\rightsquigarrow$ maximization of a SWO or of a CUF, or efficiency and envy-freeness.
- **Output**
  The allocation of a part of or the whole resource to each agent such that no constraint is violated and the criterion optimized or verified.

Back to individual preferences...

A resource allocation problem

- **Inputs**
  - A set $\mathscr{N}$ of **agents** expressing **preferences** on the resource using preorders $\succeq_i$ or utility functions u_i.
  - **The resource** $\rightsquigarrow$ a finite set $\mathscr{O}$ of indivisible objects.
  - **Some constraints** $\rightsquigarrow$ a finite set $\mathscr{C} \subset 2^{2^{\mathscr{O}^n}}$.
  - A **criterion** $\rightsquigarrow$ maximization of a SWO or of a CUF, or efficiency and envy-freeness.
- **Output**
  The allocation of a part of or the whole resource to each agent such that no constraint is violated and the criterion optimized or verified.

- No idea on how the instances are **formally represented**, and how they should be implemented.
- These precisions are crucial, particularly for the representation of **preferences** (and **constraints** as well).

Complex preferences

Preferences over objects $\mathscr{O} = \{\text{laptop}, \text{computer tower}, \text{screen}\}$ ...

Complex preferences

Preferences over objects $\mathscr{O} = \{\text{laptop}, \text{computer tower}, \text{screen}\}$ ...

"I want a complete computer (only one), no matter if it is a laptop or not."

Complex preferences

Preferences over objects $\mathscr{O} = \{\text{laptop}, \text{computer tower}, \text{screen}\}$ ...

"I want a complete computer (only one), no matter if it is a laptop or not."

Two types of dependencies appear:

- A **complementarity** between the computer tower and the screen: having one of them makes me completely unhappy (utility 0), but having both makes me completely happy (say utility 10). $\rightsquigarrow u(\{ct, s\}) > u(\{ct\}) + u(\{s\})$.
- A **substitutability** between the laptop and the pair computer tower + screen: having one of them makes me completely happy (say utility 10), but having both does not bring me more utility. $\rightsquigarrow$ $u(\{l, ct, s\}) < u(\{ct, s\}) + u(\{l\})$.

Complex preferences

Preferences over objects $\mathscr{O} = \{\text{laptop}, \text{computer tower}, \text{screen}\}$ ...

"I want a complete computer (only one), no matter if it is a laptop or not."

Two types of dependencies appear:

- A **complementarity** between the computer tower and the screen: having one of them makes me completely unhappy (utility 0), but having both makes me completely happy (say utility 10). $\rightsquigarrow u(\{ct, s\}) > u(\{ct\}) + u(\{s\})$.
- A **substitutability** between the laptop and the pair computer tower + screen: having one of them makes me completely happy (say utility 10), but having both does not bring me more utility. $\rightsquigarrow$ $u(\{l, ct, s\}) < u(\{ct, s\}) + u(\{l\})$.

Such an utility function **cannot** be represented by an additive utility function (*i.e* utilities over objects only)

Complex preferences and EOS

- **Stereographic** images (examples of complementarity).
- **Large areas**: images whose width is larger than the satellite's swath (examples of complementarity).
- **Several acquisition opportunities** (examples of substitutability).

Explicit representation

Example

Resource allocation problem with 2 objects o_1 and o_2.
Expression of the utility function:
$u(\emptyset) = 0$, $u(o_1) = 5$, $u(o_2) = 7$, $u(\{o_1, o_2\}) = 3$.

Explicit representation

Example

Resource allocation problem with 4 objects o_1, o_2, o_3 and o_4.
Expression of the utility function:
$u(\emptyset) = 0$, $u(o_1) = 5$, $u(o_2) = 7$, $u(o_3) = 2$, $u(o_4) = 8$, $u(\{o_1, o_2\}) = 3$, $u(\{o_1, o_3\}) = 5$, $u(\{o_1, o_4\}) = 3$, $u(\{o_2, o_3\}) = 0$, $u(\{o_2, o_4\}) = 6$, $u(\{o_3, o_4\}) = 2$, $u(\{o_1, o_2, o_3\}) = 8$, $u(\{o_1, o_2, o_4\}) = 9$, $u(\{o_1, o_3, o_4\}) = 10$, $u(\{o_2, o_3, o_4\}) = 3$, $u(\{o_1, o_2, o_3, o_4\}) = 10$.

Explicit representation

Example

Resource allocation problem with 20 objects $o_1, \ldots, o_{20}$
Expression of the utility function:

$u(\emptyset) = 0$, $u(o_1) = 5$, $u(o_2) = 7$, $u(o_3) = 2$, $u(o_4) = 8$ $u(o_5) = 5$, $u(o_6) = 0$, $u(o_7) = 1$, $u(o_8) = 15$ $u(o_9) = 4$, $u(o_{10}) = 6$, $u(o_{11}) = 6$, $u(o_{12}) = 8$, $u(o_{13}) = 5$, $u(o_{14}) = 7$, $u(o_{15}) = 2$, $u(o_{16}) = 8$, $u(o_{17}) = 7$, $u(o_{18}) = 2$, $u(o_{19}) = 8$, $u(o_{20}) = 7$, $u(\{o_1, o_2\}) = 15$, $u(\{o_1, o_3\}) = 12$, $u(\{o_1, o_4\}) = 5$, $u(\{o_1, o_5\}) = 1$, $u(\{o_1, o_6\}) = 4$, $u(\{o_1, o_7\}) = 2$, $u(\{o_1, o_8\}) = 8$, $u(\{o_1, o_9\}) = 10$, $u(\{o_1, o_{10}\}) = 3$, $u(\{o_1, o_{11}\}) = 11$, $u(\{o_1, o_{12}\}) = 12$, $u(\{o_1, o_{13}\}) = 5$, $u(\{o_1, o_{14}\}) = 13$, $u(\{o_1, o_{15}\}) = 3$, $u(\{o_1, o_{16}\}) = 15$, $u(\{o_1, o_{17}\}) = 1$, $u(\{o_1, o_{18}\}) = 3$, $u(\{o_1, o_{19}\}) = 11$, $u(\{o_2, o_3\}) = 12$, $u(\{o_2, o_4\}) = 5$, $u(\{o_2, o_5\}) = 1$, $u(\{o_2, o_6\}) = 4$, $u(\{o_2, o_7\}) = 2$, $u(\{o_2, o_8\}) = 8$, $u(\{o_2, o_9\}) = 10$, $u(\{o_2, o_{10}\}) = 3$, $u(\{o_2, o_{11}\}) = 11$, $u(\{o_2, o_{12}\}) = 12$, $u(\{o_2, o_{13}\}) = 5$, $u(\{o_2, o_{14}\}) = 13$, $u(\{o_2, o_{15}\}) = 3$, $u(\{o_2, o_{16}\}) = 15$, $u(\{o_2, o_{17}\}) = 1$, $u(\{o_2, o_{18}\}) = 3$, $u(\{o_2, o_{19}\}) = 11$, $u(\{o_3, o_4\}) = 5$, $u(\{o_3, o_5\}) = 1$, $u(\{o_3, o_6\}) = 4$, $u(\{o_3, o_7\}) = 2$, $u(\{o_3, o_8\}) = 8$, $u(\{o_3, o_9\}) = 10$, $u(\{o_3, o_{10}\}) = 3$, $u(\{o_3, o_{11}\}) = 11$, $u(\{o_3, o_{12}\}) = 12$, $u(\{o_3, o_{13}\}) = 5$, $u(\{o_3, o_{14}\}) = 13$, $u(\{o_3, o_{15}\}) = 3$, $u(\{o_3, o_{16}\}) = 15$, $u(\{o_3, o_{17}\}) = 1$, $u(\{o_3, o_{18}\}) = 3$,

Explicit representation

Example

Resource allocation problem with 20 objects $o_1, \ldots, o_{20}$
Expression of the utility function:

$u(\emptyset) = 0$, $u(o_1) = 5$, $u(o_2) = 7$, $u(o_3) = 2$, $u(o_4) = 8$ $u(o_5) = 5$, $u(o_6) = 0$, $u(o_7) = 1$, $u(o_8) = 15$ $u(o_9) = 4$, $u(o_{10}) = 6$, $u(o_{11}) = 6$, $u(o_{12}) = 8$, $u(o_{13}) = 5$, $u(o_{14}) = 7$, $u(o_{15}) = 2$, $u(o_{16}) = 8$, $u(o_{17}) = 7$, $u(o_{18}) = 2$, $u(o_{19}) = 8$, $u(o_{20}) = 7$, $u(\{o_1, o_2\}) = 15$, $u(\{o_1, o_3\}) = 12$, $u(\{o_1, o_4\}) = 5$, $u(\{o_1, o_5\}) = 1$, $u(\{o_1, o_6\}) = 4$, $u(\{o_1, o_7\}) = 2$, $u(\{o_1, o_8\}) = 8$, $u(\{o_1, o_9\}) = 10$, $u(\{o_1, o_{10}\}) = 3$, $u(\{o_1, o_{11}\}) = 11$, $u(\{o_1, o_{12}\}) = 12$, $u(\{o_1, o_{13}\}) = 5$, $u(\{o_1, o_{14}\}) = 13$, $u(\{o_1, o_{15}\}) = 3$, $u(\{o_1, o_{16}\}) = 15$, $u(\{o_1, o_{17}\}) = 1$, $u(\{o_1, o_{18}\}) = 3$, $u(\{o_1, o_{19}\}) = 11$, $u(\{o_2, o_3\}) = 12$, $u(\{o_2, o_4\}) = 5$, $u(\{o_2, o_5\}) = 1$, $u(\{o_2, o_6\}) = 4$, $u(\{o_2, o_7\}) = 2$, $u(\{o_2, o_8\}) = 8$, $u(\{o_2, o_9\}) = 10$, $u(\{o_2, o_{10}\}) = 3$, $u(\{o_2, o_{11}\}) = 11$, $u(\{o_2, o_{12}\}) = 12$, $u(\{o_2, o_{13}\}) = 5$, $u(\{o_2, o_{14}\}) = 13$, $u(\{o_2, o_{15}\}) = 3$, $u(\{o_2, o_{16}\}) = 15$, $u(\{o_2, o_{17}\}) = 1$, $u(\{o_2, o_{18}\}) = 3$, $u(\{o_2, o_{19}\}) = 11$, $u(\{o_3, o_4\}) = 5$, $u(\{o_3, o_5\}) = 1$, $u(\{o_3, o_6\}) = 4$, $u(\{o_3, o_7\}) = 2$, $u(\{o_3, o_8\}) = 8$, $u(\{o_3, o_9\}) = 10$, $u(\{o_3, o_{10}\}) = 3$, $u(\{o_3, o_{11}\}) = 11$, $u(\{o_3, o_{12}\}) = 12$, $u(\{o_3, o_{13}\}) = 5$, $u(\{o_3, o_{14}\}) = 13$, $u(\{o_3, o_{15}\}) = 3$, $u(\{o_3, o_{16}\}) = 15$, $u(\{o_3, o_{17}\}) = 1$, $u(\{o_3, o_{18}\}) = 3$,

1048576 values $\leadsto$ the expression needs more than 12 days (supposing the agent expresses 1 value per second).

Compact preference representation

Three possible answers to combinatorial explosion:

1. Ignore it and suppose that the number of objects is low [Herreiner and Puppe, 2002].

Brams, S. J., Edelman, P. H., and Fishburn, P. C. (2004).
Fair division of indivisible items.
Theory and Decision, 5(2):147–180.

Demko, S. and Hill, T. P. (1998).
Equitable distribution of indivisible items.
Mathematical Social Sciences, 16:145–158.

Herreiner, D. K. and Puppe, C. (2002).
A simple procedure for finding equitable allocations of indivisible goods.
Social Choice and Welfare, 19:415–430.

Compact preference representation

Three possible answers to combinatorial explosion:

1. Ignore it and suppose that the number of objects is low [Herreiner and Puppe, 2002].
2. Add some restrictive assumptions on the preferences (for example : additivity) that make the expression possible [Brams et al., 2004] and [Demko and Hill, 1998].

Brams, S. J., Edelman, P. H., and Fishburn, P. C. (2004).
Fair division of indivisible items.
Theory and Decision, 5(2):147–180.

Demko, S. and Hill, T. P. (1998).
Equitable distribution of indivisible items.
Mathematical Social Sciences, 16:145–158.

Herreiner, D. K. and Puppe, C. (2002).
A simple procedure for finding equitable allocations of indivisible goods.
Social Choice and Welfare, 19:415–430.

Compact preference representation

Three possible answers to combinatorial explosion:

1. Ignore it and suppose that the number of objects is low [Herreiner and Puppe, 2002].
2. Add some restrictive assumptions on the preferences (for example : additivity) that make the expression possible [Brams et al., 2004] and [Demko and Hill, 1998].
3. Use a **compact representation language**.

Brams, S. J., Edelman, P. H., and Fishburn, P. C. (2004).
Fair division of indivisible items.
Theory and Decision, 5(2):147–180.

Demko, S. and Hill, T. P. (1998).
Equitable distribution of indivisible items.
Mathematical Social Sciences, 16:145–158.

Herreiner, D. K. and Puppe, C. (2002).
A simple procedure for finding equitable allocations of indivisible goods.
Social Choice and Welfare, 19:415–430.

Compact representation ?

A compact representation language $\mathscr{R}$ is made of:

- **Syntax**: a set of **well-formed formulas** $\mathcal{L}_{\mathscr{R}}$;
- **Semantics**: a function from $\mathcal{L}_{\mathscr{R}}$ to $\mathscr{E} \times \mathscr{E}$ (ordinal preferences) or to $F(\mathscr{E}, \mathbb{R})$ (cardinal preferences).

Preferences, languages

Model ← Requests

$\succeq$ preorder over $\mathscr{E}$ (partial/total)
or $u : \mathscr{E} \to \mathbb{R}$

find an optimal option

find an optimal admissible option

compare two options

rank a set of options

Preferences, languages

Model — **Language** ← Requests

$\succeq$ preorder over $\mathscr{E}$ (partial/total)
or $u : \mathscr{E} \to \mathbb{R}$

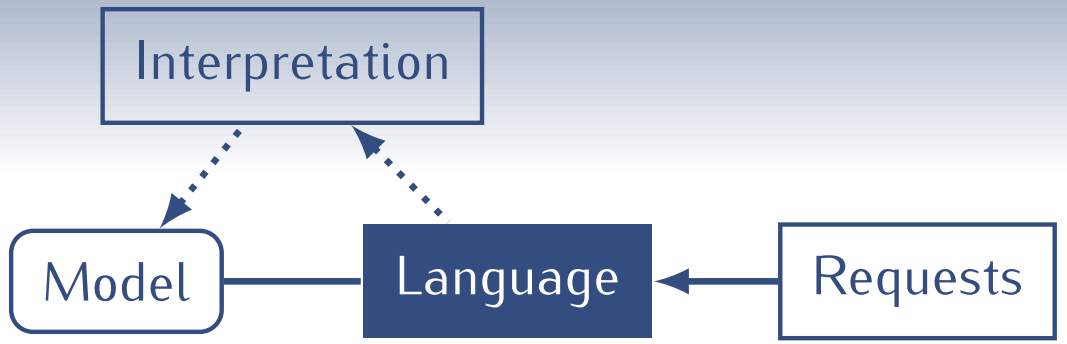
find an optimal option

find an optimal admissible option

compare two options

rank a set of options

Preferences, languages



$\succeq$ preorder over $\mathscr{E}$ (partial/total)
or $u : \mathscr{E} \to \mathbb{R}$

find an optimal option

find an optimal admissible option

compare two options

rank a set of options

Compact ?

- Number of different formulas of size less than t bits: $< 2^t$.
- 1 preorder = at least 1 formula of $\mathcal{L}_{\mathscr{R}}$ (if the language is fully expressive).

Compact ?

- Number of different formulas of size less than t bits: $< 2^t$.
- 1 preorder = at least 1 formula of $\mathcal{L}_{\mathscr{R}}$ (if the language is fully expressive).

$\rightsquigarrow$ At least $2^{|\mathscr{E}|\times|\mathscr{E}|}$ formulas (ordinal preferences). For a resource allocation problem, $|\mathscr{E}| = 2^{|\mathscr{O}|}$. Thus, $\mathcal{L}_{\mathscr{R}}$ needs at least $2^{2^{2^{|\mathscr{O}|}}}$ formulas

Compact ?

- Number of different formulas of size less than t bits: $< 2^t$.
- 1 preorder = at least 1 formula of $\mathcal{L}_{\mathscr{R}}$ (if the language is fully expressive).

$\rightsquigarrow$ At least $2^{|\mathscr{E}| \times |\mathscr{E}|}$ formulas (ordinal preferences). For a resource allocation problem, $|\mathscr{E}| = 2^{|\mathscr{O}|}$. Thus, $\mathcal{L}_{\mathscr{R}}$ needs at least $2^{2^{2|\mathscr{O}|}}$ formulas

$\rightsquigarrow$ **some formulas cannot be expressed in less than $2^{|\mathscr{O}|}$ bits.**

Compact ?

- Number of different formulas of size less than t bits: $< 2^t$.
- 1 preorder = at least 1 formula of $\mathcal{L}_{\mathscr{R}}$ (if the language is fully expressive).

$\rightsquigarrow$ At least $2^{|\mathscr{E}| \times |\mathscr{E}|}$ formulas (ordinal preferences). For a resource allocation problem, $|\mathscr{E}| = 2^{|\mathscr{O}|}$. Thus, $\mathcal{L}_{\mathscr{R}}$ needs at least $2^{2^{2|\mathscr{O}|}}$ formulas

$\rightsquigarrow$ **some formulas cannot be expressed in less than $2^{|\mathscr{O}|}$ bits.**

A compact representation language is not about representing **all** preference relations compactly, but just **the interesting ones**.

How to evaluate a language ?

- **Elicitation / cognitive relevance** [Chen and Pu, 2004]
- **Expressive power**: can a language express all possible preorders ? [Coste-Marquis et al., 2004, Chevaleyre et al., 2006]
- **Relative succinctness**: is there a polynomial size translation from $\mathscr{R}_1$ to $\mathscr{R}_2$? [Coste-Marquis et al., 2004, Chevaleyre et al., 2006]
- **Computational complexity**: how complex are the basic computational tasks (dominance, optimization,...) ? [Lang, 2004]

Chen, L. and Pu, P. (2004).
Survey of preference elicitation methods.
Technical report, École Polytechnique Fédérale de Lausanne.

Chevaleyre, Y., Endriss, U., and Lang, J. (2006).
Expressive power of weighted propositional formulas for cardinal preference modelling.
In *Proc. of KR-06*.

Coste-Marquis, S., Lang, J., Liberatore, P., and Marquis, P. (2004).
Expressive power and succinctness of propositional languages for preference representation.
In *Proc. of KR-04*.

Lang, J. (2004).
Logical preference representation and combinatorial vote.
Annals of Mathematics and Artificial Intelligence, 42(1):37–71.

About computational complexity (1)

A quick reminder about complexity theory...

Complexity theory usually deals with **decision problems**. A decision problem is made of:

- **Inputs**
- **A Yes / No question**.

Example: [SAT]

- **Input:** A formula φ in propositional logic.
- **Question:** Is φ satisfiable ?

About computational complexity (2)

- Given a class of problems parametrised by their "size", how hard it is to solve a problem of size n?
- Distinguish: **time**/space **worst-case**/average-case complexity
- Problems that can be solved in **polynomial** time (P) are considered tractable, problems requiring **exponential** time (EXPTIME) not.
- Think of a problem that requires searching through a tree. If you are lucky and go down the right branch at every branching point, you may need only polynomial time, otherwise exponential time. A **nondeterministic** algorithm is a (hypothetical) algorithm with an "oracle" that tells us which branch to explore next.
- NP is the class of decision problems that can be solved by such **nondeterministic** algorithms in **polynomial** time.

About computational complexity (3)

- Equivalent definition: NP is the class of problems for which a candidate solution can be **verified** in (determ.) polynomial time.
- A decision problem is **NP-hard** iff it is at least as hard as any of the problems in NP.
- A decision problem is **NP-complete** iff it is NP-hard and in NP.
- We do not know whether P = NP, but strongly suspect P $\neq$ NP.
- NP-complete problems are generally considered intractable. Unless P = NP, there can be no general algorithm solving NP-complete problems efficiently.
- As a rule of thumb, NP-completeness means that a naive approach won't work, but a sophisticated algorithm may well give good results in practice.
- To prove that P_1 is at least as hard as P_2, one has to find a polynomial time reduction from P_2 to P_1.

About computational complexity (4)

Example: [SAT]

- **Input:** A formula φ in propositional logic.
- **Question:** Is φ satisfiable ?

A deterministic algorithm roughly needs to search through every possible instantiations.

A non-deterministic one chooses the correct one immediately.

Some decision problems (1)

We will be mainly interested in...

[Dominance]

- **Inputs:** A preorder $\preceq$ expressed compactly. Two bundles π_1 and π_2.
- **Question:** $\pi_1 \preceq \pi_2$?

[Optimization]

- **Inputs:** A utility function u expressed compactly. An integer K.
- **Question:** Is there a bundle π such that $u(\pi) \geq K$?

Some decision problems (2)

We will be mainly interested in...

[Max CUF]

- **Inputs:** A set of utility functions $u_1, \ldots, u_n$ expressed compactly. A CUF g. An integer K.
- **Question:** Is there an allocation $\vec{\pi}$ such that $g(u_1(\pi_1), \ldots, u_n(\pi_n)) \geq K$?

[EEF Existence]

- **Inputs:** A set of preorders $\preceq_1, \ldots, \preceq_n$ expressed compactly.
- **Question:** Does there exist an efficient and envy-free allocation ?

Preference representation

Extend additivity...

Additive utility functions are probably the simplest and the most intuitive ones. If $\mathscr{O}$ is of size p, an additive utility function is only defined by p weights.

Example: $u = 3o_1 + 7o_2 - 2o_3$ is an additive function. $u(\{o_1, o_2\}) = 10$.

Extend additivity...

Additive utility functions are probably the simplest and the most intuitive ones. If $\mathscr{O}$ is of size p, an additive utility function is only defined by p weights.

Example: $u = 3o_1 + 7o_2 - 2o_3$ is an additive function. $u(\{o_1, o_2\}) = 10$.

But...Additive utility functions are unable to represent dependencies (synergies) between object.

Is there a way to represent these dependencies while keeping the representation not too big ?

The k-additive form

- A utility function is called **k-additive** iff the utility assigned to a bundle π can be represented as the sum of basic utilities assigned to subsets of π with cardinality $\leq k$ (limited synergies).
- The **k-additive form** of representing utility functions:

$$u(\pi) = \sum_{\pi' \subseteq \pi} \alpha^{\pi'} \text{ with } \alpha^{\pi'} = 0 \text{ whenever } |\pi'| > k$$

  Example: $u = 3o_1 + 7o_2 - 2o_2o_3$ is a 2-additive function
- That is, specifying a utility function in this language means specifying the **coefficients** α^{π} for bundles $\pi \subseteq \mathscr{O}$.
- The value α^{π} can be seen as the additional benefit incurred from owning the items in π **together**, i.e. beyond the benefit of owning all proper subsets.

The k-additive form – expressivity

The k-additive form is **fully expressive**, if we choose k large enough:

Proposition

Any utility function is representable in the k-additive form for some $k \leq |\mathscr{O}|$.

Proof: For any utility function u, we can define coefficients α^π:

- $\alpha^\varnothing = u(\varnothing)$;
- $\alpha^\pi = u(\pi) - \sum_{\pi' \subsetneq \pi} \alpha^{\pi'}$ for all $\pi \subseteq \mathscr{O}$.

The function $u \mapsto (\pi \mapsto \alpha^\pi)$ is called the **Möbius transform** of u [Grabisch, 1997].

Grabisch, M. (1997).
k-order additive discrete fuzzy measure and their representation.
Fuzzy Sets and Systems, 92:167–189.

Example

$\mathscr{O} = \{o_1, o_2, o_3, o_4\}$

$u = 2o_1 + 3o_2 + 2o_2o_3 + 2o_3o_4$

	$\varnothing$	o_1	o_2	o_3	o_4	o_1o_2	o_1o_3	o_1o_4	o_2o_3	o_2o_4
u	0	2	3	0	0	5	2	2	5	3

	o_3o_4	$o_1o_2o_3$	$o_1o_2o_4$	$o_2o_3o_4$	$o_1o_2o_3o_4$
u	2	7	5	7	9

Explicit vs. k-additive Form

Among the k-additive form and the explicit one, which one is the **most succinct** ?

Explicit vs. k-additive Form

Among the k-additive form and the explicit one, which one is the **most succinct** ?

Proposition

The explicit and the k-additive form of representing utility functions are incomparable with respect to succinctness.

Proof sketch: The following two functions can be used to prove the mutual lack of a polysize reduction:

- $u_1(\pi) = |\pi|$: representing u_1 requires $|\mathscr{O}|$ non-zero coefficients in the k-additive form (linear); but $2^{|\mathscr{O}|} - 1$ non-zero values in the explicit form (exponential).
- $u_2(\pi) = 1$ for $|\pi| = 1$ and $u_2(\pi) = 0$ otherwise: requires $|\mathscr{O}|$ non-zero values in the explicit form (linear); but $2^{|\mathscr{O}|} - 1$ non-zero coefficients in the k-additive form (exponential), namely $\alpha^\pi = 1$ for $|\pi| = 1$, $\alpha^\pi = -2$ for $|\pi| = 2$, $\alpha^\pi = 3$ for $|\pi| = 3$, ...

k-additive form: complexity

- [Dominance] is in P.
- [Optimization] is NP-complete, but in P if all the weights are positive.
- [EEF existence] with preemption constraints only is Σ_2^P-complete ($\Sigma_2^p = \mathbf{NP}^{\mathbf{NP}}$) even for additive preferences.
- [Max-CUF] $\rightsquigarrow$ see later.

Generalized Additive Independence

$\mathcal{X}$ set of variables (binary or not), v instantiation.

Definition

Let $\mathcal{X}_1, \ldots, \mathcal{X}_k$ be a family of subsets of $\mathcal{X}$ such that $\bigcup_i \mathcal{X}_i = \mathcal{X}$.
u is GAI-decomposable with respect to $\mathcal{X}_1, \ldots, \mathcal{X}_k$ iff $\exists u_1 \ldots u_k$ utility functions $\mathcal{X}_i \to \mathbb{R}$ such that $u(v) = \sum_{i=1}^{k} u_i(v_{\downarrow \mathcal{X}_i})$

Remarks:

- For binary variables, additivity = GAI-decomposability, with $|\mathcal{X}_i| = 1$, $\forall i$.
- For binary variables, k-Additivity = GAI-decomposability, with $|\mathcal{X}_i| \leq k$, $\forall i$.

GAI-nets: example

{house, tractor, lawn mower, fields}

f	0	1
u_1	0	10

h	0	1
u_2	0	15

tr	0	1
u_3	0	2

lm	0	1
u_4	0	1

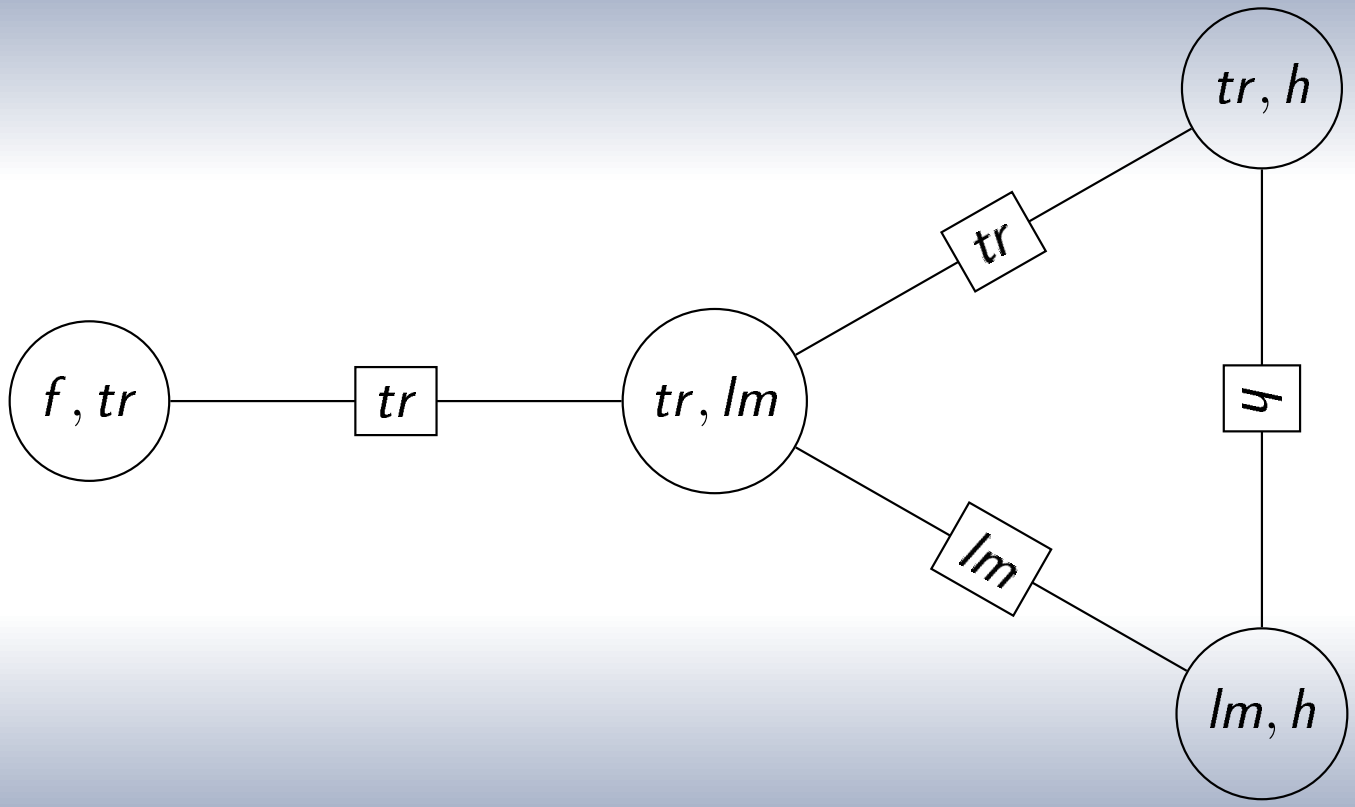
f	0	0	1	1
tr	0	1	0	1
u_5	0	0	0	8

tr	0	0	1	1
lm	0	1	0	1
u_6	0	0	0	-8

h	0	0	1	1
tr	0	1	0	1
u_7	0	0	0	7

h	0	0	1	1
lm	0	1	0	1
u_8	0	0	0	10

GAI-nets



Preference representation

Of logic and goals

Logic-based languages suit well when we have to deal with **binary variables** (*e.g.* resource allocation problems).

- A propositional language $L_{\mathscr{O}}$...
  - set of propositional symbols $\mathscr{O}$,
  - usual connectives $\neg, \wedge, \vee$

Of logic and goals

Logic-based languages suit well when we have to deal with **binary variables** (*e.g.* resource allocation problems).

- A propositional language $L_{\mathscr{O}}$...
  - set of propositional symbols $\mathscr{O}$,
  - usual connectives $\neg, \wedge, \vee$

Example

- $\mathscr{O} = \{$  $\}$.
- Set of requests for one agent:
  -  $\wedge \Big(($$) \vee$ $\Big)$,
  -  $\wedge$.

Dichotomous preferences...

What to do with all these goals ?

Dichotomous preferences...

What to do with all these goals ?

A first (simplistic) example

Example

Variables $\mathscr{O} = \{o_1, o_2, o_3\}$

- $o_2 \wedge \neg o_1$
  $\{o_2\}, \{o_2, o_3\} \succ \varnothing, \{o_1\}, \{o_3\}, \{o_1, o_2\}, \{o_1, o_3\}, \{o_1, o_2, o_3\}$
- $(o_1 \wedge o_2 \wedge \neg o_3) \vee (\neg o_1 \wedge o_2 \wedge o_3)$
  $\{o_1, o_2\}, \{o_2, o_3\} \succ \varnothing, \{o_1\}, \{o_2\}, \{o_3\}, \{o_1, o_3\}, \{o_1, o_2, o_3\}$

A bit more interesting...

- Goal bases (*e.g* $\{\varphi_1, \ldots, \varphi_m\}$):
  - cardinality
  - inclusion
  - distances
- **Prioritized** goal bases (*e.g* $\{\langle\varphi_1, k_1\rangle, \ldots, \langle\varphi_m, k_m\rangle\}$):
  - discrimin
  - leximin
  - best-out
- **Weighted** goal bases

A language based on weighted logic

Preference representation:

- A propositional language $L_{\mathscr{O}}$...
  - a set of propositional symbols $\mathscr{O}$,
  - the usual connectives $\neg, \wedge, \vee$
- ...and some weights $w \in \mathscr{V}$.

A language based on weighted logic

Preference representation:

- A propositional language $L_{\mathscr{O}}$...
  - a set of propositional symbols $\mathscr{O}$,
  - the usual connectives $\neg, \wedge, \vee$
- ...and some weights $w \in \mathscr{V}$.

Example

- $\mathscr{O} = \{$  $\}$.
- Agent 1's requests:
  - $\Big\langle$  $\Big, 110\Big\rangle$,
  - $\Big\langle$  $, -10\Big\rangle$,
  - $\Big\langle$  $, 50\Big\rangle$.

Individual utility

Expresses the satisfaction of an agent regarding an allocation. Depends on:

- her share (assumption of non exogenous preferences),
- her weighted requests,

and is obtained by **aggregating** the weights of the satisfied formulas, using an operator $\oplus$.

Individual utility

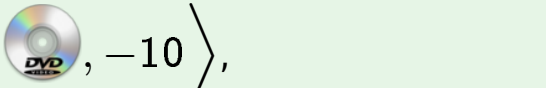
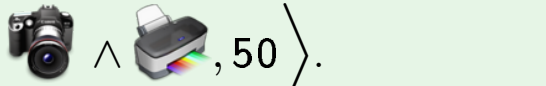
Given an agent i, her requests Δ_i, an allocation $\overrightarrow{\pi}$, her individual utility is:

$$u_i(\pi_i) = \bigoplus \{w \mid \langle \varphi, w \rangle \in \Delta_i \text{ and } x_i \vDash \varphi\}.$$

Two reasonable choices for $\oplus$: $+$ or max.

Individual utility

Example

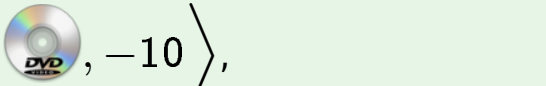
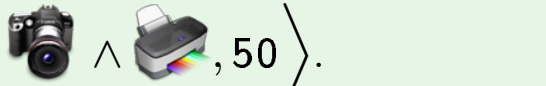
- $\mathscr{O} = \{$  $\}$.
- Agent 1's requests:
  - $\Big\langle$  $\Big\rangle$,
  - $\Big\langle$  $, -10\Big\rangle$,
  - $\Big\langle$  $, 50\Big\rangle$.

Computation of individual utility ($\oplus = +$) :

$\pi_1 = \{$  $\}$

Individual utility

Example

- $\mathscr{O} = \{$  $\}$.
- Agent 1's requests:
  - $\Big\langle$  $\Big\rangle$,
  - $\Big\langle$  $, -10\Big\rangle$,
  - $\Big\langle$  $, 50\Big\rangle$.

Computation of individual utility ($\oplus = +$) :

$\pi_1 = \{$  $\} \Rightarrow u_1(\pi_1) =$  110

Individual utility

Example

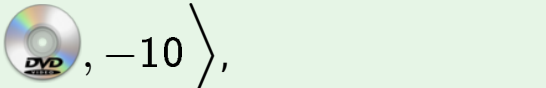
- $\mathscr{O} = \{$  $\}$.
- Agent 1's requests:
  - $\Big\langle$  $\Big, 110\Big\rangle$,
  - $\Big\langle$  $, -10\Big\rangle$,
  - $\Big\langle$  $, 50\Big\rangle$.

Computation of individual utility ($\oplus = +$) :

$\pi_1 = \{$  $\} \Rightarrow u_1(\pi_1) = 110$  -10

Individual utility

Example

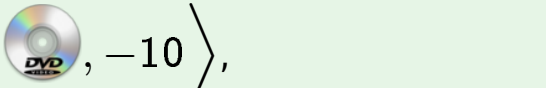
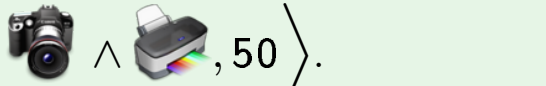
- $\mathscr{O} = \{$  $\}$.
- Agent 1's requests:
  - $\Big\langle$  $, 110\Big\rangle$,
  - $\Big\langle$  $, -10\Big\rangle$,
  - $\Big\langle$  $, 50\Big\rangle$.

Computation of individual utility ($\oplus = +$) :

$\pi_1 = \{$  $\} \Rightarrow u_1(\pi_1) = 110 - 10 +$  0

Individual utility

Example

- $\mathscr{O} = \{$  $\}$.
- Agent 1's requests:
  - $\Big\langle$  $\Big\rangle$,
  - $\Big\langle$  $, -10\Big\rangle$,
  - $\Big\langle$  $, 50\Big\rangle$.

Computation of individual utility ($\oplus = +$) :

$\pi_1 = \{$  $\} \Rightarrow u_1(\pi_1) = 110 - 10 + 0 = 100$

Weighted logic: expressive power

- **Unrestricted language**: fully expressive.
- **Positive cubes only**: fully expressive (why ?).
- **Positive cubes and positive weights only**: only monotonic functions.
- **Literals only**: additive functions

See [Uckelman, 2008] for more details

Uckelman, J. (2008).
More Than the Sum of Its Parts. Compact Preference Representation of Combinatorial Domains.
PhD thesis, Universiteit van Amsterdam.

Weighted logic: succinctness

- **Unrestricted language**: fully expressive.
- **Positive cubes only**: fully expressive.

Weighted logic: succinctness

- **Unrestricted language**: fully expressive.
- **Positive cubes only**: fully expressive.

But... Unrestricted language strictly more succinct than the restriction to positive cubes only.

Example

u defined as $\langle o_1 \vee o_2 \vee \cdots \vee o_n, 1 \rangle$.

Dominance and optimization

- [Dominance] can be solved in **polynomial time**.
- [Optimization] is **NP-complete** in the general case.
- [Optimization] is **polynomial** for the monotonic fragment of weighted logic (no negation, positive weights).
- What about [Max CUF] ?
- What about [EEF-Existence] ?

Max CUF ?

What is the complexity of the problem of maximizing collective utility ?

Problem [Max CUF]

Given an instance of the resource allocation problem with **monotonic** formulas in weighted logic and a set of **admissibility constraints**, and an integer K ($\mathscr{V} = \mathbb{N}$), does an admissible allocation $\overrightarrow{\pi}$ exists, such that $uc(\overrightarrow{\pi}) \geq K$?

This problem is **NP**-complete.

Max CUF ?

What is the complexity of the problem of maximizing collective utility ?

Problem [Max CUF]

Given an instance of the resource allocation problem with **monotonic** formulas in weighted logic and a set of **admissibility constraints**, and an integer K ($\mathscr{V} = \mathbb{N}$), does an admissible allocation $\overrightarrow{\pi}$ exists, such that $uc(\overrightarrow{\pi}) \geq K$?

This problem is **NP**-complete.

Does it remain **NP**-complete in the following cases:

- restrictions on the operators ($\oplus \in \{+, \max\}$, $g \in \{+, \min, \text{leximin}\}$),
- restrictions on the constraints (preemption, volume, exclusion),
- restriction on the preferences (atomic) ?

The complexity results

[MAX-CUF]

Any kind of constraints: **NPC**

No constraint: **P**

Exclusion constraints only

$\oplus$ \ g	+	(lexi)min
+	**NPC**	**NPC**
max	**NPC**	**NPC**

Volume constraints only

$\oplus$ \ g	+	(lexi)min
+	**NPC**	**NPC**
max	**NPC**	**NPC**

Preemption constraints

Atomic requests (**additive** prefs.)

$\oplus$ \ g	+	min	leximin
+	**P**	**NPC**, **P** if eq. wgts	**NPC**
max	**P**	**P**	?

Any kind of requests

$\oplus$ \ g	+	(lexi)min
+	**NPC**	**NPC**
max	**NPC**	**NPC**

EEF Existence ?

Proposition

The [EEF EXISTENCE] problem for agents having (monotonic or not) **dichotomous** preferences under logical form is Σ_2^p-complete ($\Sigma_2^p = \mathbf{NP}^{\mathbf{NP}}$).

Proposition

The [EEF EXISTENCE] problem for agents having **additive** Σ_2^p-complete ($\Sigma_2^p = \mathbf{NP}^{\mathbf{NP}}$).

EEF Existence ?

Proposition

The [EEF EXISTENCE] problem for agents having (monotonic or not) **dichotomous** preferences under logical form is Σ_2^p-complete ($\Sigma_2^p = \mathbf{NP}^{\mathbf{NP}}$).

Proposition

The [EEF EXISTENCE] problem for agents having **additive** Σ_2^p-complete ($\Sigma_2^p = \mathbf{NP}^{\mathbf{NP}}$).

- **Alternative efficiency criterion:**
  - completeness (NP-complete),
  - maximal number of satisfied agents (θ_2^p-complete),
  - maximal utilitarian or egalitarian utility (Δ_2^p-complete).

Preference representation

Auctions

- Numerical preferences: **money**.
- **Usual criterion**: maximize the revenue of the auctioneer.

Classical auctions:

- Additive preferences.
- Each object is sold to the bidder proposing the highest price.

From classical to combinatorial auctions

Example from [Uckelman, 2008].

An auction with two shoes: {left, right}

- The first bidder has the following utility function: $u(lr) = €40$, $u(\pi) = 0$ otherwise.
- The second bidder has an additive utility function : $u = 10l + 10r$.

How should the bidders bid in a traditional (sequential) auction ?

Uckelman, J. (2008).
More Than the Sum of Its Parts. Compact Preference Representation of Combinatorial Domains.
PhD thesis, Universiteit van Amsterdam.

Bidding languages

Bidding languages are preference representation languages developed specifically for combinatorial auctions, to allow bidders to transmit their valuations to the auctioneer.

- Bids are combinations of atomic bids $b = \langle \pi, w \rangle$: bundle/price. The valuation of a bundle π' is w is $\pi \subseteq \pi'$, and 0 otherwise.
- In the XOR-language, atomic bids by the same bidder are assumed to be mutually exclusive:

$$(b_1 \text{ XOR } b_2)(\pi) = \max\{b_1(\pi), b_2(\pi)\}$$

- In the OR-language, the valuation is taken to be the maximal value that can be obtained by accepting disjoint bids

$$(b_1 \text{ OR } b_2)(\pi) = \max_{\pi' \subseteq \pi}\{b_1(\pi), b_2(\pi \setminus \pi')\}$$

Examples

XOR language: $(\pi_1, w_1)\ XOR\ \ldots\ XOR\ (\pi_n, w_n)$

$(\{$  ,  ,  $\}, 400€)\ XOR\ (\{$  ,  $\}, 700€)\ XOR\ (\{$  $\}, 100€)$

$\{$  ,  ,  ,  ,  $\} \Rightarrow u =$

Examples

XOR language: $(\pi_1, w_1)\ XOR\ \ldots\ XOR\ (\pi_n, w_n)$

$(\{$  ,  ,  $\}, 400€)\ XOR\ (\{$  ,  $\}, 700€)\ XOR\ (\{$  $\}, 100€)$

$\{$  ,  ,  ,  ,  $\} \Rightarrow u = 700€$

Examples

OR language: $(\pi_1, w_1)\ OR\ \ldots\ OR\ (\pi_n, w_n)$

$(\{$  ,  ,  $\}, 400€)\ OR\ (\{$  ,  $\}, 700€)\ OR\ (\{$$\}, 100€)$

$\{$  ,  ,  ,  , $\} \Rightarrow u =$

Examples

OR language: (π_1, w_1) *OR* ... *OR* (π_n, w_n)

({  ,  ,  }, 400€) *OR* ({  ,  }, 700€) *OR* ({}, 100€)

{  ,  ,  ,  , } $\Rightarrow u = 800€$

Examples

Other languages

- Combinations of OR and XOR (OR-of-XOR / XOR-of-OR...)
- The **OR*-language** is like the OR-language, but dummy items can be used to express exclusiveness constraints. Example:

$$\langle\{a, dummy\}, 3\rangle \text{ OR } \langle\{b, dummy\}, 3\rangle \text{ OR } \langle\{a, b, dummy\}, 5\rangle.$$

- Weighted logic.

The WDP

Winner Determination Problem

- **Input :** A set of bids (XOR, OR, or other...)
- **Solution :** An allocation that maximizes the sum of utilities (*i.e* the auctioneer's payment).

Example

- **Agent 1 :** ({  ,  ,  }, 400€)
- **Agent 2 :** ({  ,  }, 700€) *OR* ({  }, 100€)

The WDP

Winner Determination Problem

- **Input :** A set of bids (XOR, OR, or other...)
- **Solution :** An allocation that maximizes the sum of utilities (*i.e* the auctioneer's payment).

Example

- **Agent 1 :** ({  ,  ,  }, 400€)
- **Agent 2 :** ({  ,  }, 700€) *OR* ({  }, 100€)

Solution :

- **Agent 1 :** Nothing
- **Agent 2 :** {  ,  ,  } $\rightarrow u = 800$ €.

Bidding language: features

- **Expressive power:**
  - XOR can represent all (monotonic) valuations.
  - OR can only represent superadditive valuations (why ?).
- **Succinctness:**
  - OR is strictly more succinct than XOR.

Bidding language: features

- **Expressive power:**
  - XOR can represent all (monotonic) valuations.
  - OR can only represent superadditive valuations (why ?).
- **Succinctness:**
  - OR is strictly more succinct than XOR.

Proof sketch:

- Every superadditive function has the same representation in XOR and OR.
- Function $u(\pi) = |\pi|$ needs an exponential XOR formula (whereas it only needs a linear size OR formula).

Bidding language: Complexity

- Compute the utility of a given XOR bundle: **polynomial**.
- Compute the utility of a given OR bundle: **NP-complete** !
- The WDP is also NP-complete for most languages.

Bidding language: Complexity

- Compute the utility of a given XOR bundle: **polynomial**.
- Compute the utility of a given OR bundle: **NP-complete** !
- The WDP is also NP-complete for most languages.

Proof sketch

- Membership to NP is easy.
- Hardness by reduction from [Set Packing]

[Set Packing]

- **Input**: Collection $\mathcal{C}$ of finite sets and $K \in \mathbb{N}$.
- **Question**: Is there a collection of disjoint sets $\mathcal{C}' \subseteq \mathcal{C}$ s.t. $|\mathcal{C}'| > K$?

Preference representation

Compact representation of ordinal preferences

Most works on fair division assume cardinal preferences. But...

- ordinal preferences are easier to elicit
- ordinality does not require preferences to be interpersonally comparable
- several important criteria need only ordinal preferences

Compact representation of ordinal preferences

Most works on fair division assume cardinal preferences. But...

- ordinal preferences are easier to elicit
- ordinality does not require preferences to be interpersonally comparable
- several important criteria need only ordinal preferences

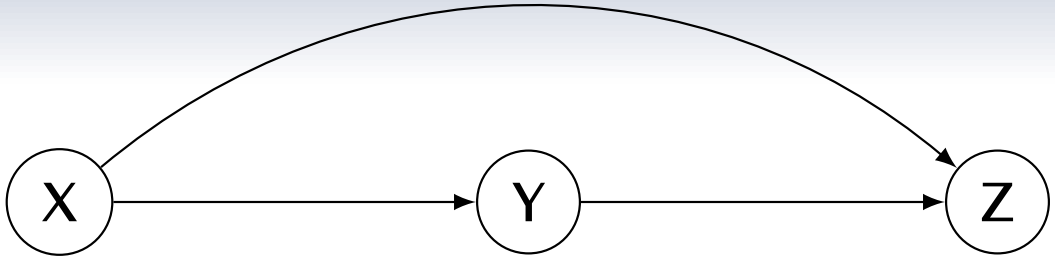
Question: *How to represent ordinal preferences compactly ?*

Idea: Use conditional preference independence
[Keeney and Raiffa, 1976].

Keeney, R. L. and Raiffa, H. (1976).
Decisions with Multiple Objectives: Preferences and Value Tradeoffs.
John Wiley and Sons.

CP-nets [Boutilier et al., 2004]

A graphical language dedicated to the specification of preferences over combinatorial domains, based on conditional preference independence.

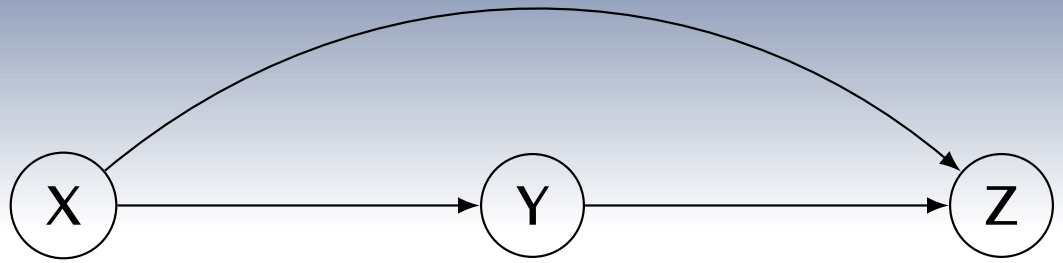


$x \succ \bar{x}$	$x : y \succ \bar{y}$ $\bar{x} : \bar{y} \succ y$	$x \vee y : z \succ \bar{z}$ $\neg(x \vee y) : \bar{z} \succ z$

$x : y \succ \bar{y}$ | if $X = x$
then $Y = y$ preferred to $Y = \bar{y}$
all other things (z) being equal (*ceteris paribus*)

$$xyz \succ x\bar{y}z; \quad xy\bar{z} \succ x\bar{y}\bar{z};$$
$$\bar{x}\bar{y}z \succ \bar{x}yz; \quad \bar{x}\bar{y}\bar{z} \succ \bar{x}y\bar{z}$$

CP-nets



$x \succ \bar{x}$	$x : y \succ \bar{y}$ $\bar{x} : \bar{y} \succ y$	$x \vee y : z \succ \bar{z}$ $\neg(x \vee y) : \bar{z} \succ z$

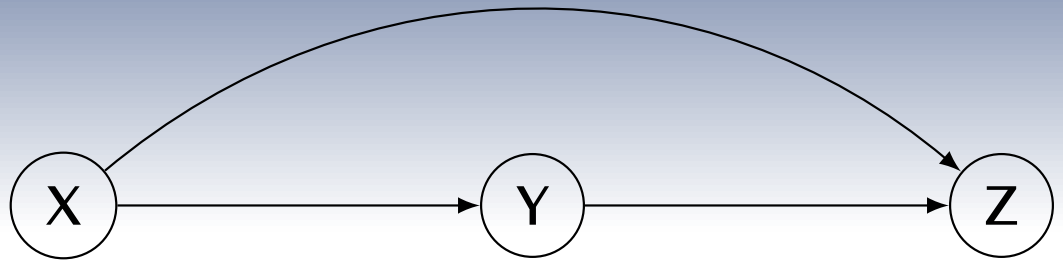
$\succ^X$: $xyz \succ \bar{x}yz$, $xy\bar{z} \succ \bar{x}y\bar{z}$, $x\bar{y}z \succ \bar{x}\bar{y}z$, $x\bar{y}\bar{z} \succ \bar{x}\bar{y}\bar{z}$

$\succ^Y$: $xyz \succ x\bar{y}z$, $xy\bar{z} \succ x\bar{y}\bar{z}$, $\bar{x}\bar{y}z \succ \bar{x}yz$, $\bar{x}\bar{y}\bar{z} \succ \bar{x}y\bar{z}$

$\succ^Z$: $xyz \succ xy\bar{z}$, $x\bar{y}z \succ x\bar{y}\bar{z}$, $\bar{x}yz \succ \bar{x}\bar{y}z$, $\bar{x}\bar{y}\bar{z} \succ \bar{x}\bar{y}z$

$\succ_{\mathcal{C}}$ = transitive closure of $\succ^X \cup \succ^Y \cup \succ^Z$

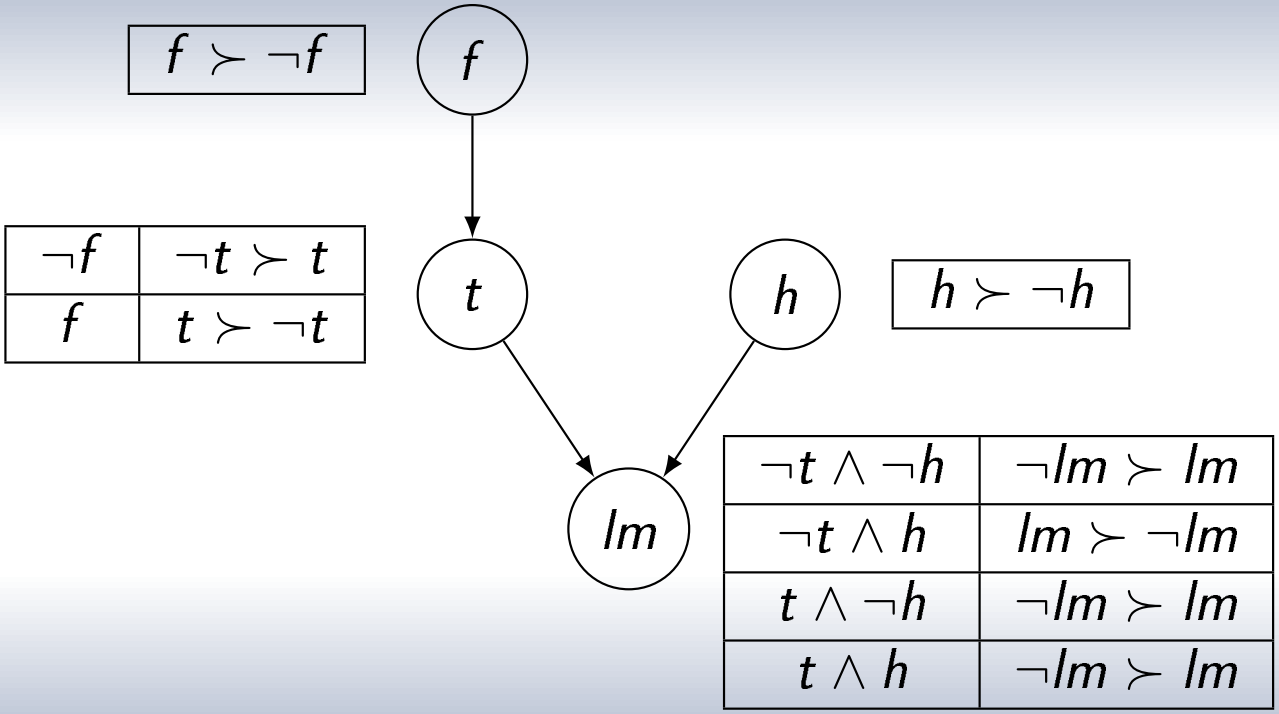
CP-nets



$x \succ \bar{x}$	$x : y \succ \bar{y}$ $\bar{x} : \bar{y} \succ y$	$x \vee y : z \succ \bar{z}$ $\neg(x \vee y) : \bar{z} \succ z$

$$\succ: \quad xyz \begin{array}{c} \nearrow \\ \searrow \end{array} \begin{array}{c} x\bar{y}z \\ xy\bar{z} \end{array} \begin{array}{c} \searrow \\ \nearrow \end{array} x\bar{y}\bar{z} \rightarrow \bar{x}\bar{y}\bar{z} \rightarrow \bar{x}\bar{y}z \rightarrow \bar{x}yz \rightarrow \bar{x}y\bar{z}$$

CP-nets

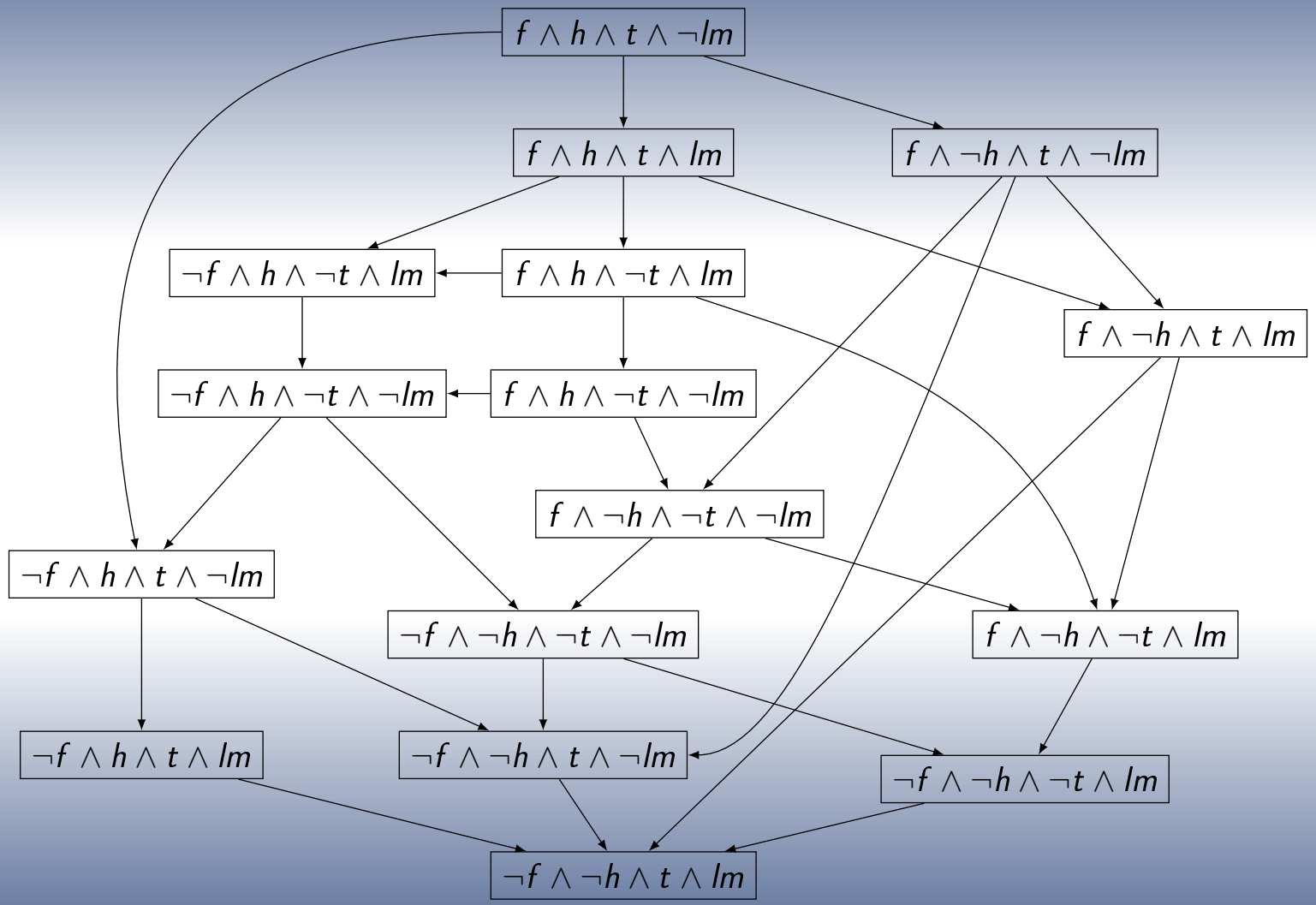


$\neg f$	$\neg t \succ t$
f	$t \succ \neg t$

$\neg t \wedge \neg h$	$\neg lm \succ lm$
$\neg t \wedge h$	$lm \succ \neg lm$
$t \wedge \neg h$	$\neg lm \succ lm$
$t \wedge h$	$\neg lm \succ lm$

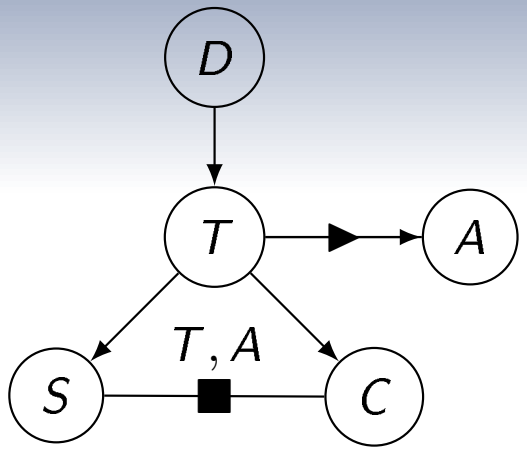
(f: fields, t: tractor, lm: lawn mower, h: house)

Transcription of the CP-net



Extensions of CP-nets

- TCP-nets [Brafman and Domshlak, 2002].



- CP theories [Wilson, 2004].

Brafman, R. I. and Domshlak, C. (2002).
Introducing variable importance tradeoffs into CP-nets.
In Darwiche, A. and Friedman, N., editors, *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI-02)*, pages 69–76, Edmonton, Alberta, Canada. Morgan Kaufmann.

Wilson, N. (2004).
Extending CP-nets with stronger conditional preference statements.
In *Proceedings of AAAI'04*.

CP-nets [Boutilier et al., 2004]

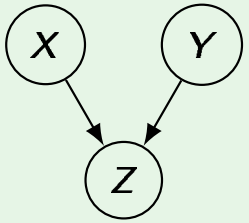
Back to resource allocation...
...Are CP-nets really well-suited represent preferences in resource allocation problems ?

CP-nets [Boutilier et al., 2004]

Back to resource allocation...
...Are CP-nets really well-suited represent preferences in resource allocation problems ?

Example

$x, y : z_1 \succ z_2 \rightarrow$ "All other things being equal, if $X = x$ and $Y = y$, then I prefer having $Z = z_1$ than $Z = z_2$".



- **CP-nets**: $a : b \triangleright \overline{b}$;
- whereas we want: $a : b \triangleright c$.

Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H., and Poole, D. (2004).
CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements.
Journal of Artificial Intelligence Research, 21:135–191.

TCP-nets [Brafman et al., 2006]

- CP-nets enriched with (conditional) importance statements.
- **TCP-nets**: $a : b \triangleright c$

Brafman, R. I., Domshlak, C., and Shimony, S. E. (2006).
On graphical modeling of preference and importance.
J. Artif. Intell. Res. (JAIR), 25:389–424.

TCP-nets [Brafman et al., 2006]

- CP-nets enriched with (conditional) importance statements.
- **TCP-nets**: $a : b \triangleright c$
- ...but we also want: $a : bc \triangleright de$.

Brafman, R. I., Domshlak, C., and Shimony, S. E. (2006).
On graphical modeling of preference and importance.
J. Artif. Intell. Res. (JAIR), 25:389–424.

CI-nets: the language

Conditional importance statement

Conditional importance statement: $\mathcal{S}^+, \mathcal{S}^- : \mathcal{S}_1 \triangleright \mathcal{S}_2$ (with $\mathcal{S}^+$, $\mathcal{S}^-$, $\mathcal{S}_1$ and $\mathcal{S}_2$ pairwise-disjoint).

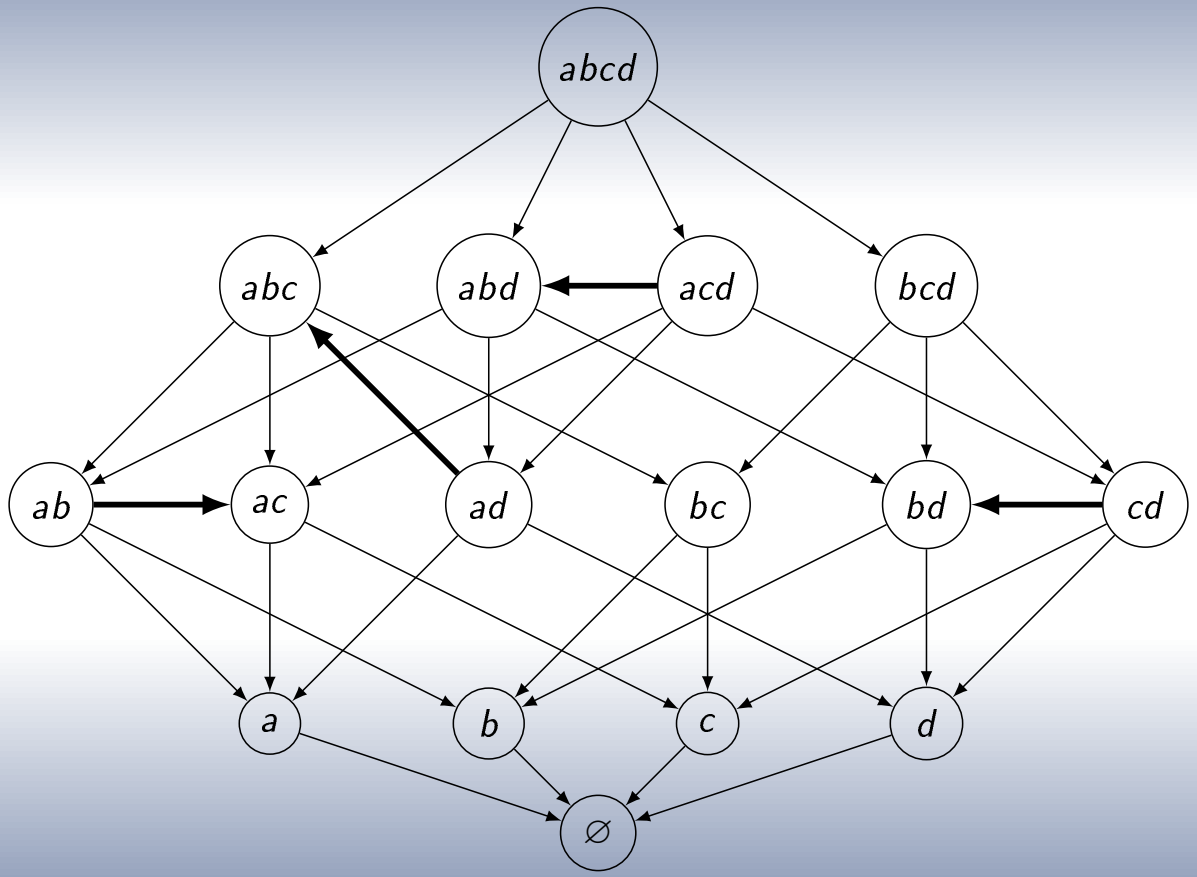
Example: $a\overline{d} : b \triangleright ce$ implies for example $ab \succ ace$, $abfg \succ acefg$, ...

CI-net

A CI-net on $\mathcal{V}$ is a set $\mathcal{N}$ of conditional importance statements on $\mathcal{V}$.

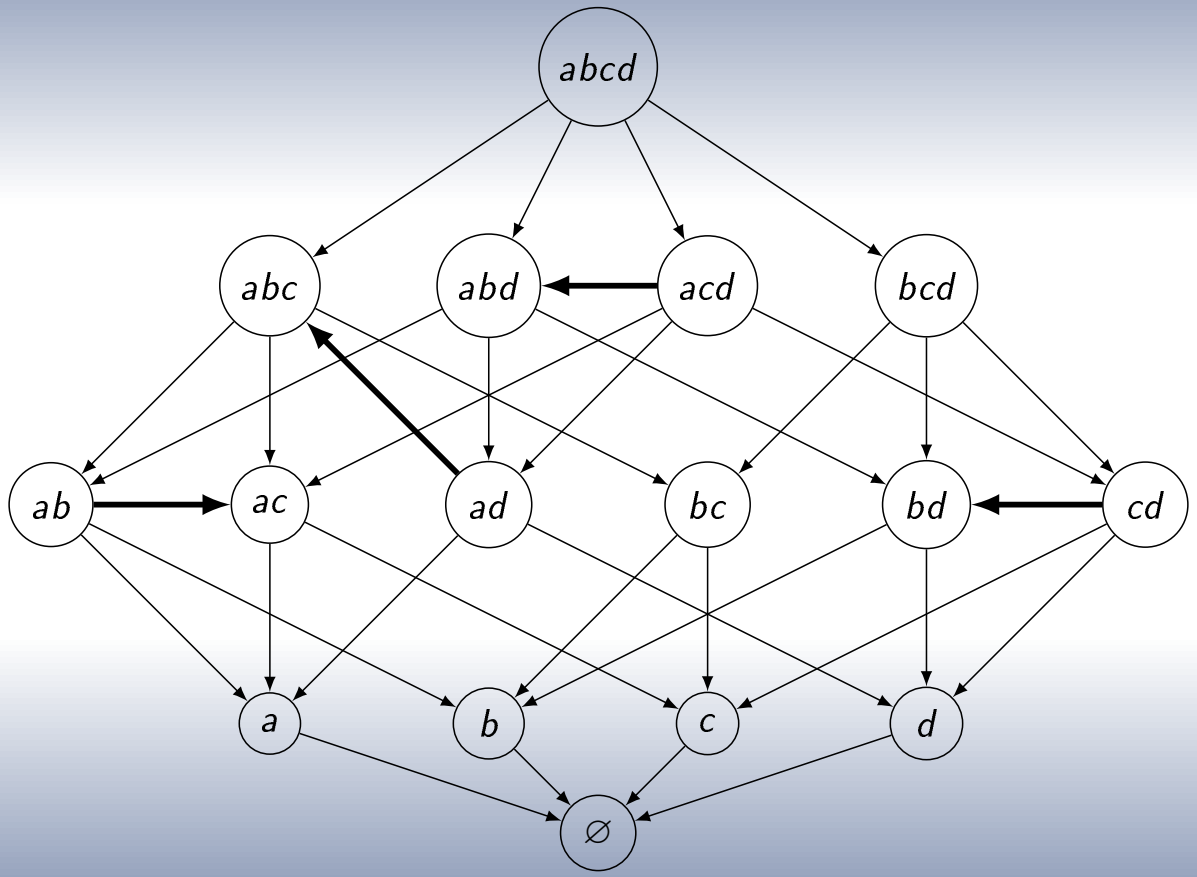
Semantics

A CI-net of 4 objects $\{a, b, c, d\}$: $\{a : d \triangleright bc, a\overline{d} : b \triangleright c, d : c \triangleright b\}$



Semantics

A CI-net of 4 objects $\{a, b, c, d\}$: $\{a : d \triangleright bc, a\overline{d} : b \triangleright c, d : c \triangleright b\}$



Induced preference relation $\succ_{\mathcal{N}}$: the smallest preference **monotonic** relation compatible with **all CI-statements**.

Expressivity

Proposition

CI-nets can express all strict monotonic preference relations on $2^{\mathcal{V}}$.

Proof sketch: for every (X, Y) such that $X \succ Y$ and $X \not\supset Y$, add the CI-statement $(X \cap Y), (\overline{X \cup Y}) : X \setminus Y \rhd Y \setminus X$.

Proposition

Full expressivity is lost as soon as:

(i) we do not allow positive preconditions;

(ii) we do not allow negative preconditions;

(iii) the cardinality of compared sets is bounded by a fixed integer.

Optimization

[Optimization]

Input: A satisfiable CI-net $\mathcal{N}$, a bundle π.
Question: Is π non dominated for $\succ_{\mathcal{N}}$?

Optimization

[Optimization]

Input: A satisfiable CI-net $\mathcal{N}$, a bundle π.
Question: Is π non dominated for $\succ_{\mathcal{N}}$?

Irrelevant: the entire set is the only one non dominated set!

Optimization

[Optimization]

Input: A satisfiable CI-net $\mathcal{N}$, a bundle π.
Question: Is π non dominated for $\succ_{\mathcal{N}}$?

Irrelevant: the entire set is the only one non dominated set!

More interesting:

- **Constrained optimization**
- **Resource allocation**

Dominance

[Dominance]

Input: A (satisfiable) CI-net $\mathcal{N}$, two bundles π and π'.
Question: $\pi \succ_{\mathcal{N}} \pi'$?

Dominance

[Dominance]

Input: A (satisfiable) CI-net $\mathcal{N}$, two bundles π and π'.
Question: $\pi \succ_{\mathcal{N}} \pi'$?

- [Dominance] is in general PSPACE-complete (*i.e* very hard !);
- But polynomial for precondition-free, singleton-comparing CI-statements.
  Example: $\{a \triangleright c, b \triangleright c, e \triangleright d\}$.

Separable ordinal preferences

- CI-nets are probably too general $\rightsquigarrow$ too complex.
- Let's choose a simpler language.
- **Restriction**: each agent specifies a linear order $\rhd$ on O (**single** objects)

$$\mathcal{N} : a \rhd b \rhd c \rhd d$$

Separable ordinal preferences

- CI-nets are probably too general $\rightsquigarrow$ too complex.
- Let's choose a simpler language.
- **Restriction**: each agent specifies a linear order $\rhd$ on O (**single** objects)

$$\mathcal{N} : a \rhd b \rhd c \rhd d$$

Problem: How to compare **subsets** of objects ?

$$\rightsquigarrow e.g\ abc \overset{?}{\prec\succ} ab;\ ab \overset{?}{\prec\succ} ac\ ?$$

Separable ordinal preferences

- CI-nets are probably too general $\rightsquigarrow$ too complex.
- Let's choose a simpler language.
- **Restriction**: each agent specifies a linear order $\rhd$ on O (**single** objects)

$$\mathcal{N} : a \rhd b \rhd c \rhd d$$

Problem: How to compare **subsets** of objects ?

$$\rightsquigarrow e.g\ abc \overset{?}{\prec\succ} ab;\ ab \overset{?}{\prec\succ} ac\ ?$$

1. Assume monotonicity $\rightsquigarrow e.g\ abc \succ ab$.

Separable ordinal preferences

- CI-nets are probably too general $\rightsquigarrow$ too complex.
- Let's choose a simpler language.
- **Restriction**: each agent specifies a linear order $\rhd$ on O (**single** objects)

$$\mathcal{N} : a \rhd b \rhd c \rhd d$$

Problem: How to compare **subsets** of objects ?

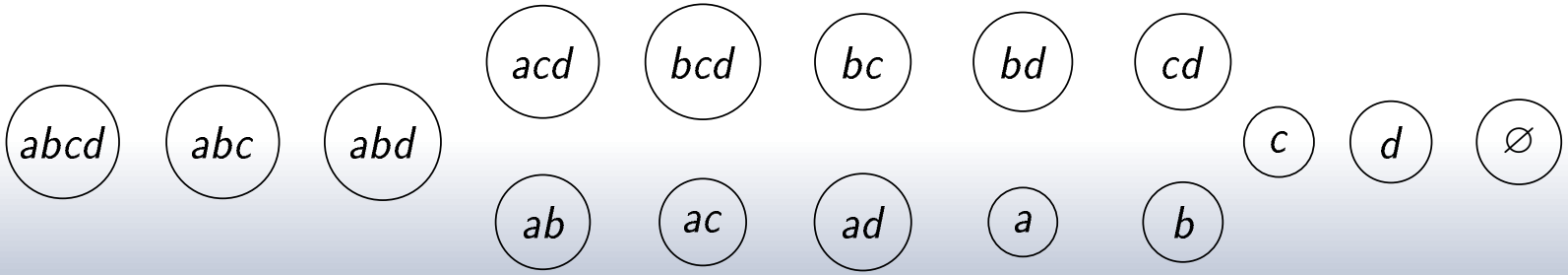
$$\rightsquigarrow e.g\ abc \overset{?}{\prec\succ} ab;\ ab \overset{?}{\prec\succ} ac\ ?$$

1. Assume monotonicity $\rightsquigarrow e.g\ abc \succ ab$.
2. Assume separability: if $(X \cup Y) \cap Z = \varnothing$ then $X \succ Y$ iff $X \cup Z \succ Y \cup Z$.
   $\rightsquigarrow e.g\ ab \succ ac$.

...Just like in SCI-nets!

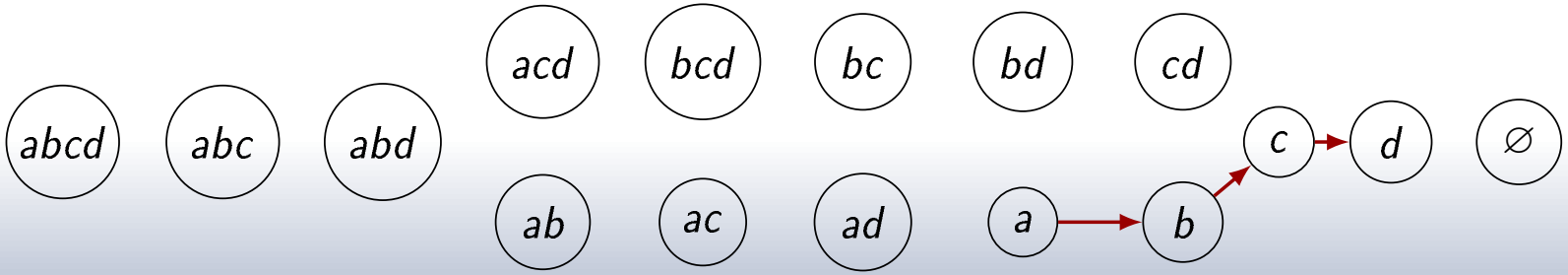
Example

- $\mathcal{N} : a \rhd b \rhd c \rhd d$
- Separability
- Monotonicity



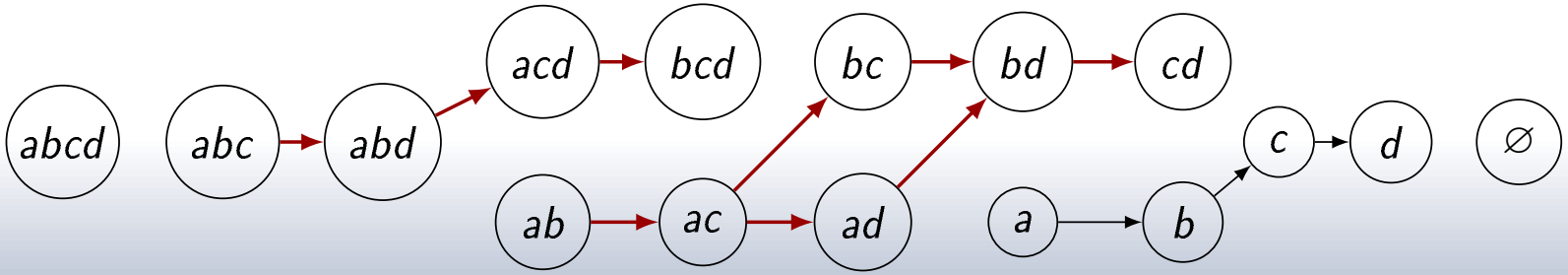
Example

- $\mathcal{N} : a \rhd b \rhd c \rhd d$
- Separability
- Monotonicity



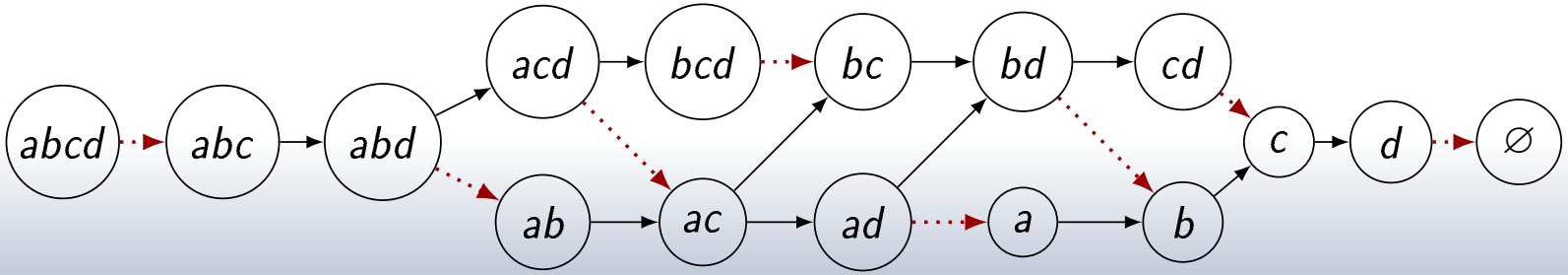
Example

- $\mathcal{N} : a \rhd b \rhd c \rhd d$
- Separability
- Monotonicity



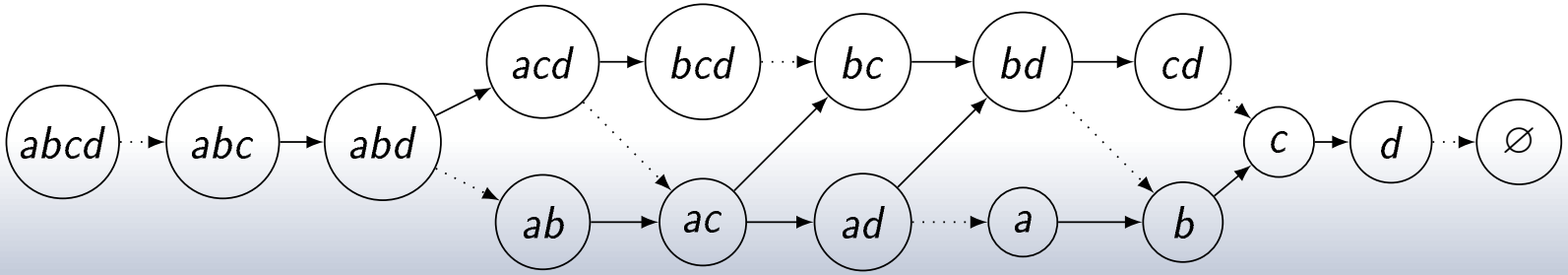
Example

- $\mathcal{N} : a \rhd b \rhd c \rhd d$
- Separability
- Monotonicity



Example

- $\mathcal{N} : a \rhd b \rhd c \rhd d$
- Separability
- Monotonicity



Dominance

Proposition

$X \succ_{\mathcal{N}} Y \Leftrightarrow \exists$ an **injective mapping of improvements** $Y \mapsto X$.

Dominance

Proposition

$X \succ_{\mathcal{N}} Y \Leftrightarrow \exists$ an **injective mapping of improvements** $Y \mapsto X$.

Example: $\mathcal{N} = a \rhd b \rhd c \rhd d \rhd e \rhd f$

- $\{ a , c , d \} \succ_{\mathcal{N}} \{ b , c , e \}$
- $\{ a , d , e \}$ and $\{ b , c , f \}$ are incomparable.
- $\{a, c, d\}$ and $\{b, c, e, f\}$ are incomparable.

Dominance

Proposition

$X \succ_{\mathcal{N}} Y \Leftrightarrow \exists$ an **injective mapping of improvements** $Y \mapsto X$.

Example: $\mathcal{N} = a \rhd b \rhd c \rhd d \rhd e \rhd f$

- $\{ a , c , d \} \succ_{\mathcal{N}} \{ b , c , e \}$
- $\{ a , d , e \}$ and $\{ b , c , f \}$ are incomparable.
- $\{a, c, d\}$ and $\{b, c, e, f\}$ are incomparable.

Dominance

Proposition

$X \succ_{\mathcal{N}} Y \Leftrightarrow \exists$ an **injective mapping of improvements** $Y \mapsto X$.

Example: $\mathcal{N} = a \rhd b \rhd c \rhd d \rhd e \rhd f$

- $\{ a , c , d \} \succ_{\mathcal{N}} \{ b , c , e \}$
- $\{ a , d , e \}$ and $\{ b , \overset{?}{c} , f \}$ are incomparable.
- $\{a, c, d\}$ and $\{b, c, e, f\}$ are incomparable.

Dominance

Proposition

$X \succ_{\mathcal{N}} Y \Leftrightarrow \exists$ an **injective mapping of improvements** $Y \mapsto X$.

Example: $\mathcal{N} = a \rhd b \rhd c \rhd d \rhd e \rhd f$

- $\{ a , c , d \} \succ_{\mathcal{N}} \{ b , c , e \}$
- $\{ a , d , e \}$ and $\{ b , c , f \}$ are incomparable.
- $\{a, c, d\}$ and $\{b, c, e, f\}$ are incomparable.

Brams, S. J., Edelman, P. H., and Fishburn, P. C. (2004).
Fair division of indivisible items.
Theory and Decision, 5(2):147–180.

Brams, S. J. and King, D. (2005).
Efficient fair division—help the worst off or avoid envy?
Rationality and Society, 17(4):387–421.

Envy-freeness

Fairness...

Envy-freeness

Fairness...

Envy-freeness: $\langle \succ_1, \ldots, \succ_n \rangle$ **total** strict orders, allocation π.

$$\pi \text{ envy-free } \Leftrightarrow \forall i, j, \pi(i) \succ_i \pi(j)$$

Envy-freeness

Fairness...

Envy-freeness: $\langle \succ_1, \ldots, \succ_n \rangle$ **total** strict orders, allocation π.

$$\pi \text{ envy-free } \Leftrightarrow \forall i, j, \pi(i) \succ_i \pi(j)$$

When $\langle \succ_1, \ldots, \succ_n \rangle$ are partial orders ?

Envy-freeness

Fairness...

Envy-freeness: $\langle \succ_1, \ldots, \succ_n \rangle$ **total** strict orders, allocation π.

$$\pi \text{ envy-free } \Leftrightarrow \forall i, j, \pi(i) \succ_i \pi(j)$$

When $\langle \succ_1, \ldots, \succ_n \rangle$ are partial orders ?

$\rightsquigarrow$ Envy-freeness becomes a **modal** notion

Possible and necessary Envy-freeness

- π is Possibly Envy-Free *iff* for all i, j, we have $\pi(j) \not\succ_i \pi(i)$;
- π is Necessary Envy-Free *iff* for all i, j, we have $\pi(i) \succ_i \pi(j)$.

Pareto-efficiency

Efficiency...

Pareto-efficiency

Efficiency...

- Complete allocation.
- Pareto-efficiency

Pareto-efficiency

Efficiency...

Classical Pareto dominance

π' **dominates** π if for all i, $\pi'(i) \succeq_i \pi(i)$ and for some j, $\pi'(j) \succ_j \pi(j)$

Extended to possible and necessary Pareto dominance.

- π is *possibly Pareto-efficient* (PPE) if there exists no allocation π' such that π' necessarily dominates π.
- π' is *necessarily Pareto-efficient* (NPE) if there exists no allocation π' such that π' possibly dominates π.

Envy-freeness and efficiency

complete PPE NPE ← Efficiency

Envy-freeness and efficiency

complete PPE NPE ← Efficiency

PEF
NEF
↑
Fairness

Envy-freeness and efficiency

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

Efficiency

Fairness

Envy-freeness and efficiency

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

Efficiency

Fairness

Envy-freeness and efficiency cannot always be satisfied simultaneously

Envy-freeness and efficiency

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

Efficiency

Fairness

Envy-freeness and efficiency cannot always be satisfied simultaneously

Questions:

- under which conditions is it guaranteed that there exists a allocation that satisfies Fairness and Efficiency ?
- how hard it is to determine whether such an allocation exists?

Complete possibly envy-free allocations

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

Complete possibly envy-free allocations

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

Result

n agents, m objects, k distinct goods are top-ranked by some agent.

$$\exists \text{ complete PEF allocation } \Leftrightarrow m \geq 2n - k.$$

Constructive proof (algorithm/protocol)

Example

$\mathcal{N}_1$: $a \rhd b \rhd c \rhd d \rhd e \rhd f$ $\quad$ $\mathcal{N}_2$: $a \rhd d \rhd b \rhd c \rhd e \rhd f$
$\mathcal{N}_3$: $b \rhd a \rhd c \rhd d \rhd f \rhd e$ $\quad$ $\mathcal{N}_4$: $b \rhd a \rhd d \rhd e \rhd f \rhd c$

$$(k = 2;\ m = 6 \geq 2n - k)$$

Consider the agents in order $1 > 2 > 3 > 4$:

- *first step*: give a to 1; give b to 3; 1 and 3 leave the room;
- *second step*: give d to 2; give d to 4;
- *third step*: give e to 4; give c to 2.

PPE-PEF allocations

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

PPE-PEF allocations

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

Result

∃ PPE-PEF allocation ⇔ ∃ complete, PEF allocation.

Based on the characterization of efficient allocations in [Brams and King, 2005].

Brams, S. J. and King, D. (2005).
Efficient fair division—help the worst off or avoid envy?
Rationality and Society, 17(4):387–421.

NPE-PEF allocations

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

NPE-PEF allocations

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

Complexity of the existence of NPE-PEF allocations: *open*.

Complete NEF allocations

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

- Two easy necessary conditions:
  - distinct top ranked objects;
  - m is a multiple of n.

Complete allocation

- deciding whether there exists a complete NEF allocation is NP-complete (even if $m = 2n$).
- the problem falls down in P for two agents

(hardness by reduction from [X3C])

Pareto-efficient-NEF allocations

	complete	PPE	NPE
PEF	X	X	X
NEF	X	X	X

Possible and necessary Pareto-efficiency

- existence of a PPE-NEF allocation: NP-complete
- existence of a NPE-NEF allocation: NP-hard but probably not in NP (Σ_2^p-completeness conjectured).

Preference representation

Conclusion on preference representation

- We need to express **dependencies** between objects.
- Thus we need compact representation language.

Conclusion on preference representation

- We need to express **dependencies** between objects.
- Thus we need compact representation language.

Designing such a language is not an easy task...

- Cognitive relevance (intuition)
- Relevance to the problem at stake (resource allocation)
- Computational complexity
- Succinctness

Conclusion on preference representation

- We need to express **dependencies** between objects.
- Thus we need compact representation language.

Designing such a language is not an easy task...

- Cognitive relevance (intuition)
- Relevance to the problem at stake (resource allocation)
- Computational complexity
- Succinctness

Main families of languages:

- **Generalized additive**
- **Logic based**
- **Bidding languages**
- **Conditional preferences**

Part 4

Solving multiagent resource allocation problems

Solving multiagent resource allocation problems

Back to individual preferences...

A resource allocation problem

- **Inputs**
  - A set $\mathscr{N}$ of **agents** expressing preferences on the resource in a compact way using preorders $\succeq_i$ or utility functions u_i.
  - **The resource** $\rightsquigarrow$ a finite set $\mathscr{O}$ of indivisible objects.
  - **Some constraints** $\rightsquigarrow$ a finite set $\mathscr{C} \subset 2^{2^{\mathscr{O}^n}}$.
  - A **criterion** $\rightsquigarrow$ maximization of a SWO or of a CUF, or efficiency and envy-freeness.
- **Output**
  The allocation of a part of or the whole resource to each agent such that no constraint is violated and the criterion optimized or verified.

Back to individual preferences...

A resource allocation problem

- **Inputs**
  - A set $\mathscr{N}$ of **agents** expressing preferences on the resource in a compact way using preorders $\succeq_i$ or utility functions u_i.
  - **The resource** $\rightsquigarrow$ a finite set $\mathscr{O}$ of indivisible objects.
  - **Some constraints** $\rightsquigarrow$ a finite set $\mathscr{C} \subset 2^{2^{\mathscr{O}^n}}$.
  - A **criterion** $\rightsquigarrow$ maximization of a SWO or of a CUF, or efficiency and envy-freeness.
- **Output**
  The allocation of a part of or the whole resource to each agent such that no constraint is violated and the criterion optimized or verified.

- We know that we have to find a *good* allocation (and we have formal definitions of what "good" is), but we do not know (yet) **how** to find it.

Centralised vs. Distributed Negotiation

An allocation procedure to determine a suitable allocation of resources may be either centralised or distributed:

- In the centralised case, a single entity decides on the final allocation, possibly after having elicited the preferences of the other agents. Example: combinatorial auctions
- In the distributed case, allocations emerge as the result of a sequence of local negotiation steps. Such local steps may or may not be subject to structural restrictions (say, bilateral deals).

Which approach is appropriate under what circumstances?

Advantages of the Centralised Approach

Much recent work in the MAS community on negotiation and resource allocation has concentrated on centralised approaches, in particular on combinatorial auctions.
There are several reasons for this:

- The communication protocols required are relatively simple.
- Many results from economics and game theory , in particular on mechanism design, can be exploited.
- There has been a recent push in the design of powerful algorithms for winner determination in combinatorial auctions.

Drawbacks of the Centralised Approach

But there are also some disadvantages of the centralised approach:

- Can we trust the centre (the auctioneer)?
- Does the centre have the computational resources required?
- Less natural to take an initial allocation into account (in an auction, typically the auctioneer owns everything to begin with).
- Less natural to model step-wise improvements over the status quo.
- Arguably, only the distributed approach is a serious implementation of the MAS paradigm.

Solving multiagent resource allocation problems

Solving the WDP

- The large majority of works in solving multiagent resource allocation problems concern the Winner Determination Problem
- In spite of the intractability of the WDP, sophisticated algorithms often manage to solve even large CA instances in practice.
- Two types of approaches to optimal winner determination in the general case:
  - Use powerful general-purpose mathematical programming (integer programming) software (next slide)
  - Develop search algorithms specifically for winner determination, combining general AI search techniques and domain-specific heuristics (rest of this lecture)

Integer Linear Programming

An integer linear program (ILP for short) is made of:

- A set of **integer variables** $\{\mathbf{x_1}, \ldots, \mathbf{x_n}\}$.
- A **linear criterion** to maximize or minimize : $\sum_{i=1}^{n} c_i \cdot \mathbf{x_i}$.
- A set of **linear constraints** : $\sum_{j=1}^{n} a_{i,j} \cdot \mathbf{x_j} \leq b_i$.

Highly optimised software packages for mathematical programming (such as CPLEX/ILOG) can often solve such problems efficiently.

The linear model (OR bids)

- **Variables**: $\mathbf{x_i}$, for all bid (π_i, w_i), with $d(x_i) = \{0, 1\}$.
- **Criterion**: **maximize** $\sum_i w_i \times \mathbf{x_i}$
- **Constraints**: For each object o, $\sum_{\pi_i | o \in \pi_i} \mathbf{x_i} \leq 1$.

Search for an Optimal Solution

Next we are going to see how to customise well-known search techniques developed in AI so as to solve the WDP. This part of the lecture will largely follow the survey article by [Sandholm, 2006]

Sandholm, T. W. (2006).
Optimal winner determination algorithms.
In Cramton, P., Shoham, Y., and Steinberg, R., editors, *Combinatorial auctions*. MIT Press.

Search Techniques in AI

A generic approach to search uses the **state-space representation**:

- Represent the problem as a set of **states** and define **moves** between states. Given an initial state, this defines a search tree.
- The goal states are states that correspond to valid solutions.
- Each move is associated with a cost (or a payoff).
- A solution is a sequence of moves from the initial state to a goal state with minimum cost (maximum payoff).
- **Examples**: route finding (states are cities and moves are directly connecting roads), but it also applies to CAs...

A search algorithm defines the order in which to traverse the tree:

- Uninformed search: breadth-first, depth-first, iterat. deepening
- Heuristic-guided search: branch-and-bound, A*

State Space and Moves

There are (at least) two natural ways of representing the state space and moves between states.

State Space and Moves

There are (at least) two natural ways of representing the state space and moves between states.

- Either: Define a state as a set of goods for which an allocation decision has already been made. Then making a move in the state space amounts to making a decision for a further good.
- Or: Define a state as a set of atomic bids for which an acceptance decision has already been made. In this case, a move amounts to making a decision for a further bid.

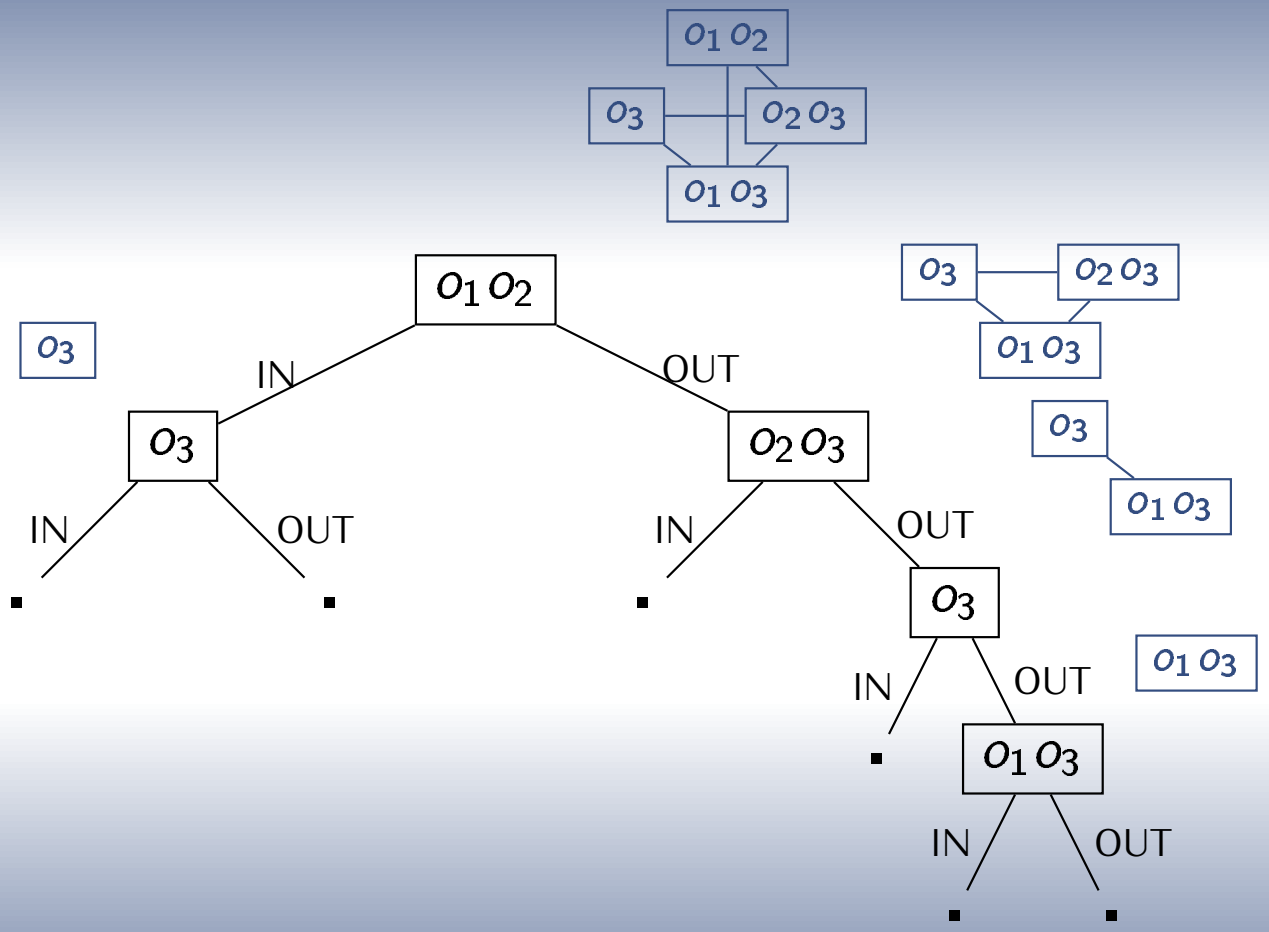
What is the initial state? What are the goal states?

According to Sandholm (2006), the bid-oriented approach tends to give better performance in practice.

Moves in Bid-oriented Search

- We represent bids as triples (i, π_j, w_j): agent i is offering to buy the bundle π_j for a price of w_j .
- The initial state is when no decisions on bids have been made.
- A move amounts to making a decision (accept/reject) for a new bid.
- The bidding language specifies which bids (if any) must be accepted/rejected given earlier decisions. **Examples:**
  - For both the OR and the XOR language: only accept bids with empty intersection of bundles.
  - For the XOR language: accept at most one bid per agent.
- We are in a goal state once a decision for every bid has been made (some of which will be consequences of the explicit choices).
- Observe that that the search tree will be binary (accept or reject?).

Example



Source: Sandholm (2006)

Uninformed Search

Uninformed search algorithms (in particular depth-first search) can be used to find a solution with a given minimum revenue: traverse the tree and keep the best solution encountered so far in memory.

Optimality can only be guaranteed if we traverse the entire search tree (not feasible for interesting problem instances).

Heuristic-guided Search

In the worst case, any algorithm may have to search the entire search tree. But good **heuristics**, that tell us which part of the tree to explore next, often allow us to avoid this in practice.

For any node N in the search tree, let $g(N)$ be the revenue generated by accepting (only) the bids accepted according to N.

We are going to need a heuristic that allows us to estimate for every node N how much revenue over and above $g(N)$ can be expected if we pursue the path through N. This will be denoted as $h(N)$. The more accurate the estimate, the better – but the only strict requirement is that h never underestimates the true revenue.

Several algorithms use such heuristics: branch-and-bound, A*...

Heuristic Upper Bounds on Revenue

Sandholm (2006) discusses several ways of defining a heuristic function h such that $g(N) + h(N)$ is guaranteed to be an upper bound on revenue for any path through node N.
Here is one such heuristic function:

- For each object o, compute its maximum contribution as:

$$c(o) = \max\{\frac{w}{|\pi|} \mid (\pi, w) \in \mathit{Bids} \text{ and } o \in \pi\}$$

- Then define $h(N)$ as the sum of all $c(o)$ for those goods o that have not yet been allocated in N.

This is indeed an upper bound (why?).

Depth-first Branch-and-Bound

This algorithm works like basic (uninformed) depth-first search, except that branches that have no chance of developing into an optimal solution get pruned on the fly:

- Traverse the search tree in depth-first order.
- Keep track of which of the nodes encountered so far would generate maximum revenue. Call that node N^* .
- If a node N with $g(N) + h(N) \leq g(N^*)$ is encountered, remove that node and all its offspring from the search tree.

This is **correct** (guarantees that the optimal solution does not get removed) whenever the heuristic function h is guaranteed never to underestimate expected marginal revenue (why?).

Another popular approach is the A* algorithm.

Branching Heuristics

So far, we have not specified which bid to select for branching in each round (for any of our algorithms). This choice does not affect correctness, but it may affect speed.

There are two basic heuristics for bid selection:

- Select a bid with a high price and a low number of items.
- Select a bid that would decompose the conflict graph of the remaining bids (if available).

Solving multiagent resource allocation problems

Constraint networks

Constraint network [Montanari, 1974]

A constraint network is based on :

- a set of variables $\mathscr{X} = \{\mathbf{x_1}, \ldots, \mathbf{x_p}\}$;
- a set of domains $\mathscr{D} = \{\mathscr{D}_{\mathbf{x_1}}, \ldots, \mathscr{D}_{\mathbf{x_p}}\}$;
- a set of constraints $\mathscr{C}$, with, for all $c \in \mathscr{C}$:
  - $X(c)$ the scope of the constraint,
  - $R(c)$ the set of allowed tuples of the constraint.

Montanari, U. (1974).
Network of constraints: Fundamental properties and applications to picture processing.
Inf. Sci., 7:95–132.

The Constraint Satisfaction Problem

Classical CSP

Given : A constraint network $(\mathscr{X}, \mathscr{D}, \mathscr{C})$.

Is there a complete consistent instantiation v of $(\mathscr{X}, \mathscr{D}, \mathscr{C})$ *?*

$\rightsquigarrow$ NP-complete.

The Constraint Satisfaction Problem

Classical CSP

Given : A constraint network $(\mathscr{X}, \mathscr{D}, \mathscr{C})$.

Is there a complete consistent instantiation v of $(\mathscr{X}, \mathscr{D}, \mathscr{C})$ *?*

$\rightsquigarrow$ NP-complete.

CSP with objective variable

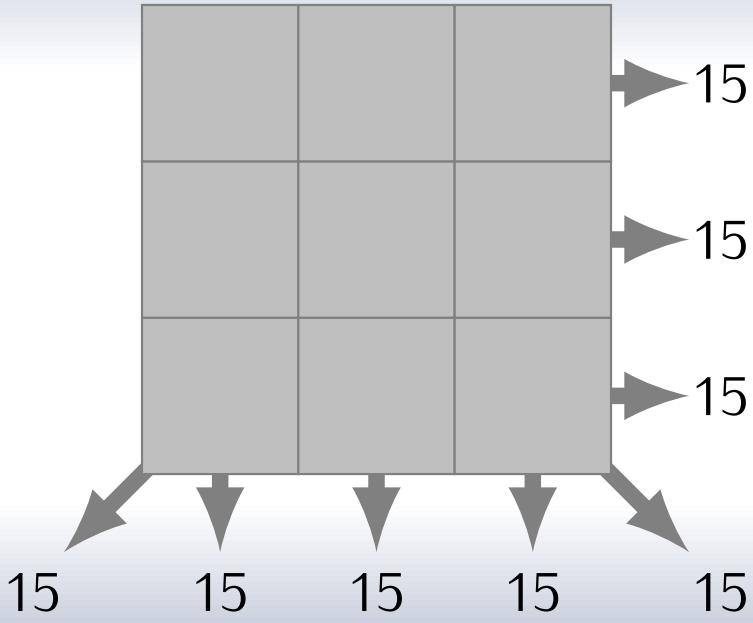
Given : A constraint network $(\mathscr{X}, \mathscr{D}, \mathscr{C})$ and an objective variable $\mathbf{o} \in \mathscr{X}$, such that $\mathscr{D}_{\mathbf{o}} \subset \mathbb{N}$.

What is the maximal value α *of* $\mathscr{D}_{\mathbf{o}}$ *such that there is a complete consistent instantiation* $\widehat{v}$ *with* $\widehat{v}(\mathbf{o}) = \alpha$ *?*

$\rightsquigarrow$ NP-complete (decision problem).

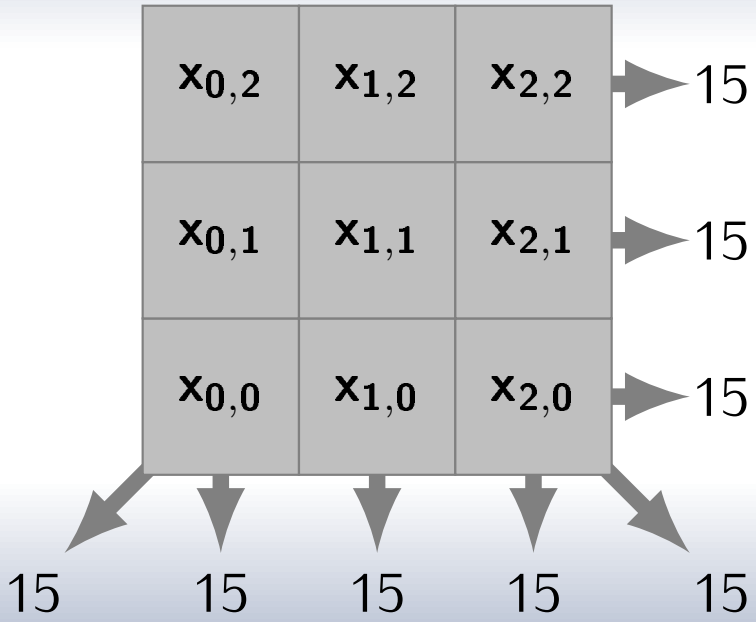
Example of a CSP (1)

The magic square, aka. *Lo Shu* square



Example of a CSP (1)

The magic square, aka. *Lo Shu* square



Variables:

$\mathbf{x_{i,j}}$, with $(i,j) \in \{0,1,2\}^2$.

Domains:

$\forall i,j,\ \mathscr{D}_{\mathbf{x_{i,j}}} = \{1,\ldots,9\}$.

Constraints:

$\mathbf{alldiff}(\mathbf{x_{i,j}} | (i,j) \in \{0,1,2\}^2)$,

$\forall i,\ \sum_{j=0}^{2} \mathbf{x_{i,j}} = 15$,

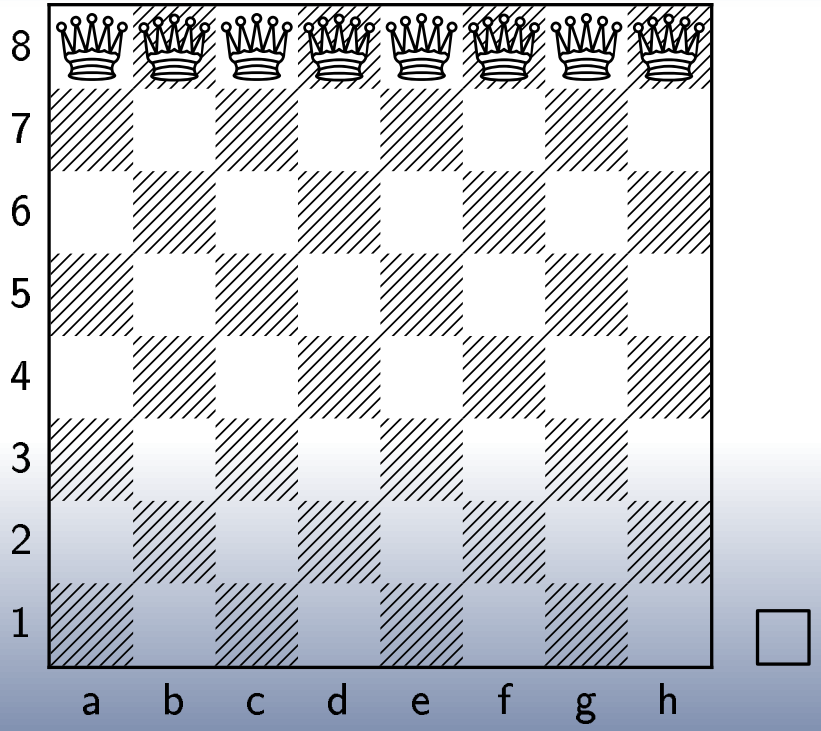
$\forall j,\ \sum_{i=0}^{2} \mathbf{x_{i,j}} = 15$,

$\mathbf{x_{0,0}} + \mathbf{x_{1,1}} + \mathbf{x_{2,2}} = 15$,

$\mathbf{x_{2,0}} + \mathbf{x_{1,1}} + \mathbf{x_{0,2}} = 15$.

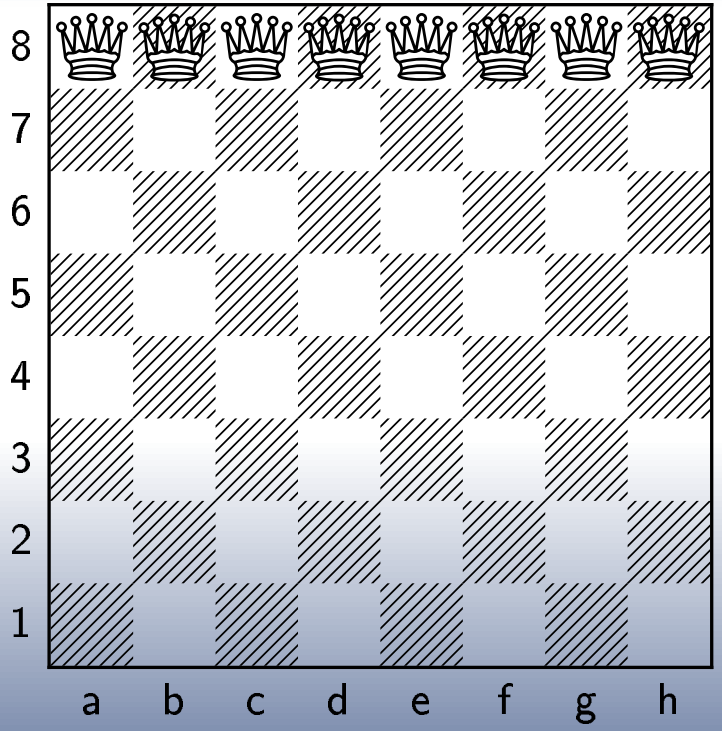
Example of a CSP (2)

The n-queens problem



Example of a CSP (2)

The n-queens problem



Variables:
$\mathbf{x_i}$, with $i \in \{0, \ldots, n\}$.
Domains:
$\forall i, j,\ \mathscr{D}_{\mathbf{x_i}} = \{1, \ldots, n\}$.
Constraints:
$\mathbf{alldiff}(\mathbf{x_i} | i \in \{0, \ldots, n\})$,
$\forall i \neq j,\ x_i - i \neq x_j - j$,
$\forall i \neq j,\ x_i + i \neq x_j + j$.

Resolution

- Search tree exploration
- Main Filtering techniques:
  - Forward checking
  - Arc-consistency
- Other kinds of consistency techniques.
  - Path-consistency,
  - k-consistency,
  - bound-consistency,...

Arc consistency and n-ary constraints

- There exists many efficient algorithms for maintaining arc-consistency (AC3, AC2001, ...).
- These solving approaches are efficient on problems **binary constraints**.
- But...

Arc consistency and n-ary constraints

- There exists many efficient algorithms for maintaining arc-consistency (AC3, AC2001, ...).
- These solving approaches are efficient on problems **binary constraints**.
- But...

The alldiff constraint

- $\mathbf{x_1} \ldots \mathbf{x_n}$;
- $\mathscr{D}_{\mathbf{x_1}} = \cdots = \mathscr{D}_{\mathbf{x_n}} = [\![1, n-1]\!]$;
- $\mathbf{alldiff}(\mathbf{x_i} | i \in [\![1, n]\!])$ vs $\{x_i \neq x_j | (i, j) \in [\![1, n]\!]^2 \text{ and } i \neq j\}$.

Arc consistency and n-ary constraints

- There exists many efficient algorithms for maintaining arc-consistency (AC3, AC2001, ...).
- These solving approaches are efficient on problems **binary constraints**.
- But...

The alldiff constraint

- $\mathbf{x_1} \ldots \mathbf{x_n}$;
- $\mathscr{D}_{\mathbf{x_1}} = \cdots = \mathscr{D}_{\mathbf{x_n}} = [\![1, n-1]\!]$;
- $\mathbf{alldiff}(\mathbf{x_i} | i \in [\![1, n]\!])$ vs $\{x_i \neq x_j | (i, j) \in [\![1, n]\!]^2 \text{ and } i \neq j\}$.

- Conversely, maintaining general n-consistency for problems with n-ary constraints is very expansive !

Global constraints

Many n-ary (global) constraints are ubiquitous in real-world problems (e.g. **Alldiff**).

Idea : use the constraints' semantics to design efficient propagation algorithms.

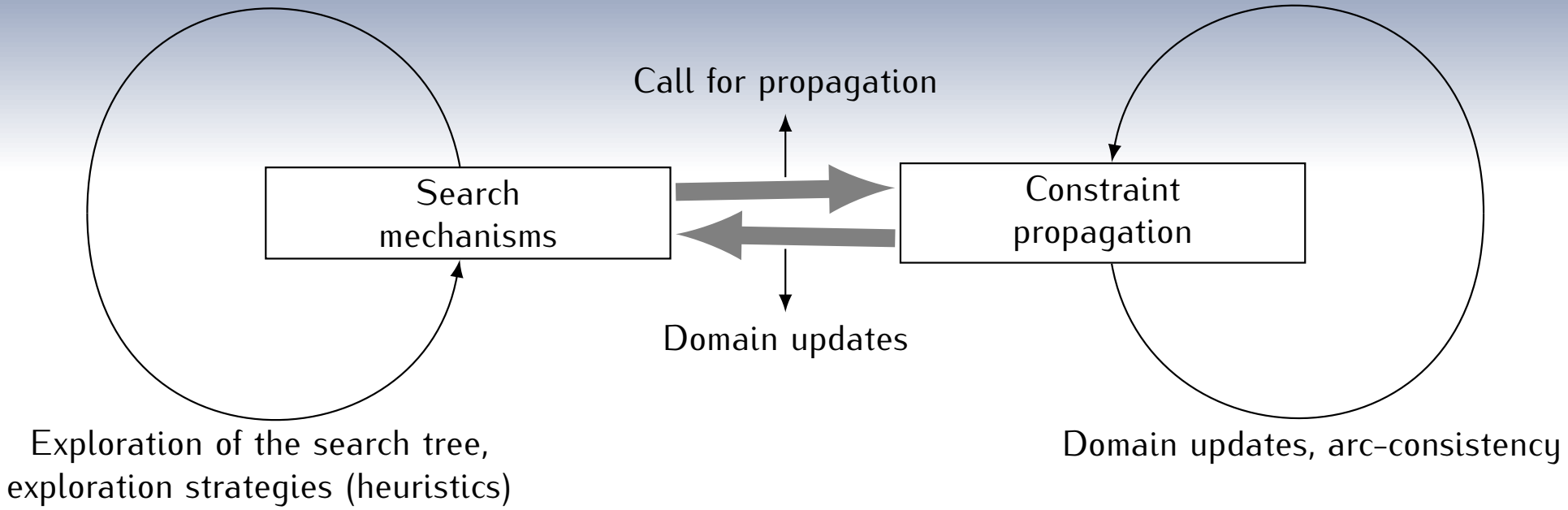
Constraint Alldiff

Propagation algorithm running in time $O(n^2 d^2)$ and space $O(nd)$

Possibly design algorithms that are only run in response to **specific events** :

- variable instantiation ;
- value removal ;
- lower or upper bound update.

Constraint Programming



What we can do:

- Set up the problem (declare variables, domains, constraints).
- Implement new constraint propagation algorithms.
- Make calls to functions **solve** or **maximize** (black boxes).

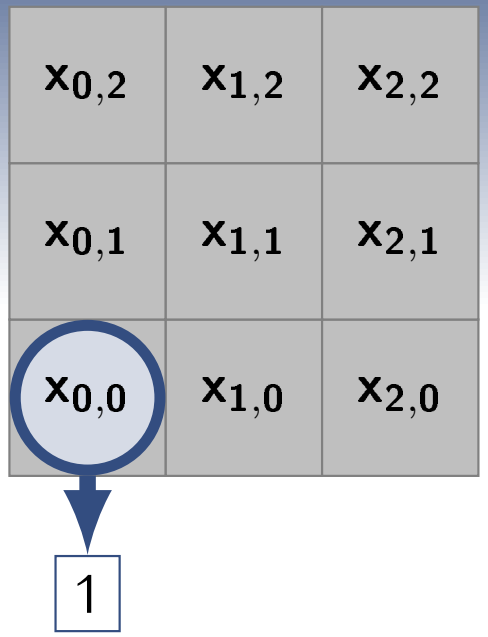
Coming back to the example

$x_{0,2}$	$x_{1,2}$	$x_{2,2}$
$x_{0,1}$	$x_{1,1}$	$x_{2,1}$
$x_{0,0}$	$x_{1,0}$	$x_{2,0}$

$x_{0,0}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{0,1}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{0,2}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{1,0}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{1,1}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{1,2}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{2,0}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{2,1}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{2,2}$: $\{1,2,3,4,5,6,7,8,9\}$

Events queue:

Coming back to the example

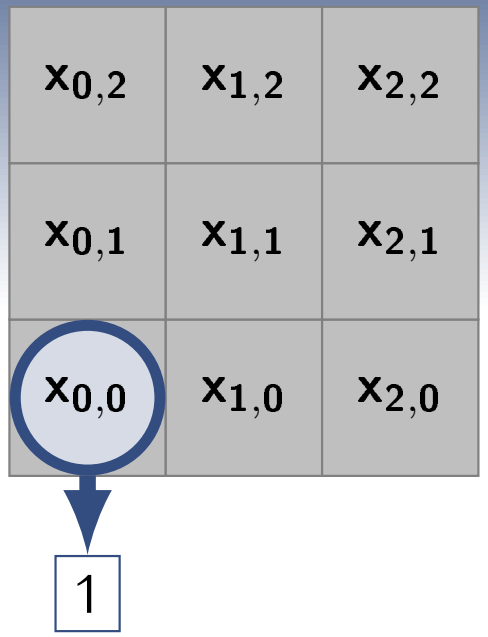


$x_{0,0}$: $\{1\}$
$x_{0,1}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{0,2}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{1,0}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{1,1}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{1,2}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{2,0}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{2,1}$: $\{1,2,3,4,5,6,7,8,9\}$
$x_{2,2}$: $\{1,2,3,4,5,6,7,8,9\}$

Events queue:

awakeOnInst($\mathbf{alldiff}(x_{i,j} | (i,j) \in \{0,1,2\}^2)$, $x_{0,0}$),
awakeOnInst($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{0,0}$),
awakeOnInst($\sum_{j=0}^{2} x_{0,j} = 15$, $x_{0,0}$),
awakeOnInst($x_{0,0} + x_{1,1} + x_{2,2} = 15$, $x_{0,0}$).

Coming back to the example

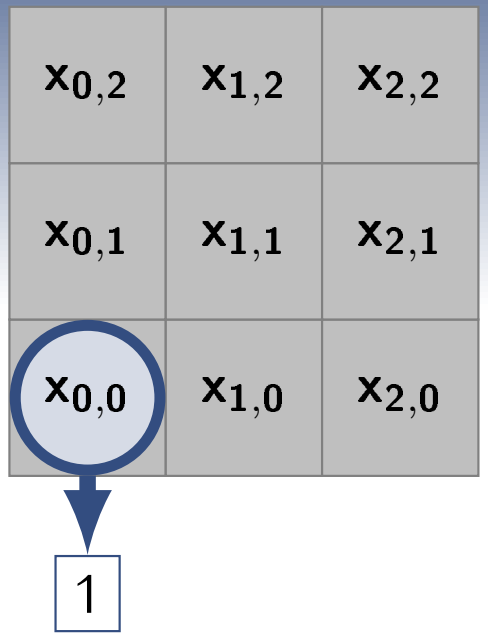


$x_{0,0}$: $\{1\}$
$x_{0,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{0,2}$: $\{2,3,4,5,6,7,8,9\}$
$x_{1,0}$: $\{2,3,4,5,6,7,8,9\}$
$x_{1,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{1,2}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,0}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,2}$: $\{2,3,4,5,6,7,8,9\}$

Events queue:

awakeOnInst($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{0,0}$),
awakeOnInst($\sum_{j=0}^{2} x_{0,j} = 15$, $x_{0,0}$),
awakeOnInst($x_{0,0} + x_{1,1} + x_{2,2} = 15$, $x_{0,0}$),
awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{0,1}$),
...

Coming back to the example

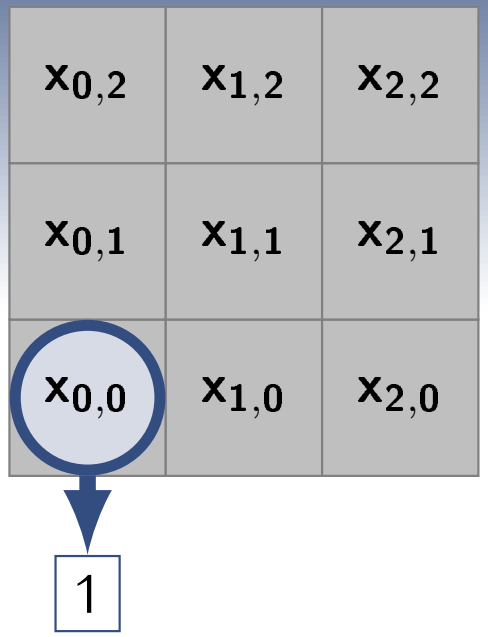


$x_{0,0}$: $\{1\}$
$x_{0,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{0,2}$: $\{2,3,4,5,6,7,8,9\}$
$x_{1,0}$: $\{5,6,7,8,9\}$
$x_{1,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{1,2}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,0}$: $\{5,6,7,8,9\}$
$x_{2,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,2}$: $\{2,3,4,5,6,7,8,9\}$

Events queue:

awakeOnInst($\sum_{j=0}^{2} x_{0,j} = 15$, $x_{0,0}$),
awakeOnInst($x_{0,0} + x_{1,1} + x_{2,2} = 15$, $x_{0,0}$),
awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{0,1}$),
awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{0,2}$),
...

Coming back to the example

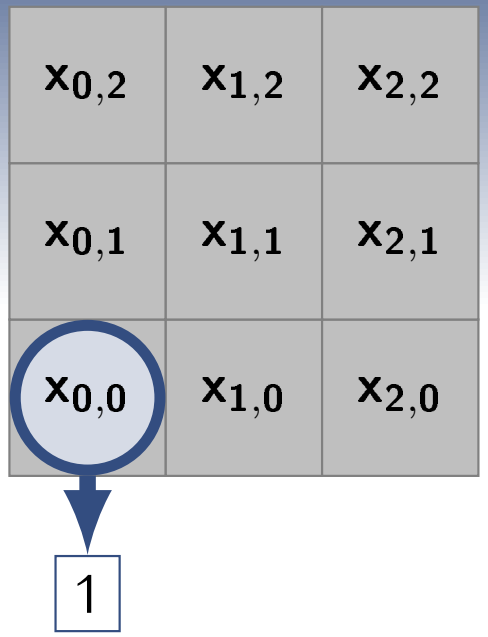


$x_{0,0}$: $\{1\}$
$x_{0,1}$: $\{5,6,7,8,9\}$
$x_{0,2}$: $\{5,6,7,8,9\}$
$x_{1,0}$: $\{5,6,7,8,9\}$
$x_{1,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{1,2}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,0}$: $\{5,6,7,8,9\}$
$x_{2,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,2}$: $\{2,3,4,5,6,7,8,9\}$

Events queue:

awakeOnInst($x_{0,0} + x_{1,1} + x_{2,2} = 15$, $x_{0,0}$),
awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{0,1}$),
awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{0,2}$),
awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{1,0}$),
...

Coming back to the example

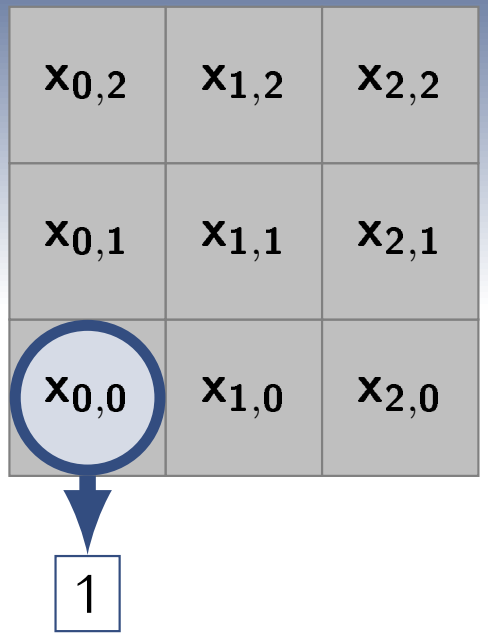


$x_{0,0}$: $\{1\}$
$x_{0,1}$: $\{5,6,7,8,9\}$
$x_{0,2}$: $\{5,6,7,8,9\}$
$x_{1,0}$: $\{5,6,7,8,9\}$
$x_{1,1}$: $\{5,6,7,8,9\}$
$x_{1,2}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,0}$: $\{5,6,7,8,9\}$
$x_{2,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,2}$: $\{5,6,7,8,9\}$

Events queue:

awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{0,1}$),
awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{0,2}$),
awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{1,0}$),
awakeOnInf($\sum_{i=0}^{2} x_{i,0} = 15$, $x_{1,1}$),
...

Coming back to the example

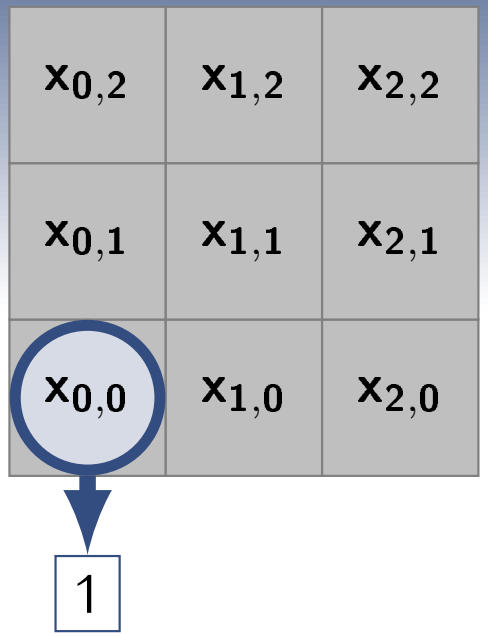


$\mathbf{x_{0,0}}$: $\{1\}$
$\mathbf{x_{0,1}}$: $\{5,6,7,8,9\}$
$\mathbf{x_{0,2}}$: $\{5,6,7,8,9\}$
$\mathbf{x_{1,0}}$: $\{5,6,7,8,9\}$
$\mathbf{x_{1,1}}$: $\{5,6,7,8,9\}$
$\mathbf{x_{1,2}}$: $\{2,3,4,5,6,7,8,9\}$
$\mathbf{x_{2,0}}$: $\{5,6,7,8,9\}$
$\mathbf{x_{2,1}}$: $\{2,3,4,5,6,7,8,9\}$
$\mathbf{x_{2,2}}$: $\{5,6,7,8,9\}$

Events queue:

...
awakeOnInf($\mathbf{alldiff}(\mathbf{x_{i,j}} | (i,j) \in \{0,1,2\}^2)$, $\mathbf{x_{1,0}}$),
...

Coming back to the example



$x_{0,0}$: $\{1\}$
$x_{0,1}$: $\{5,6,7,8,9\}$
$x_{0,2}$: $\{5,6,7,8,9\}$
$x_{1,0}$: $\{5,6,7,8,9\}$
$x_{1,1}$: $\{5,6,7,8,9\}$
$x_{1,2}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,0}$: $\{5,6,7,8,9\}$
$x_{2,1}$: $\{2,3,4,5,6,7,8,9\}$
$x_{2,2}$: $\{5,6,7,8,9\}$

Events queue:

...

awakeOnInf(**alldiff**$(x_{i,j}|(i,j) \in \{0,1,2\}^2)$, $x_{1,0}$) $\rightsquigarrow$ **Inconsistent!**,

...

Back to Earth Observing Satellites...

Input of the problem...

- A set of **logical requests** from the agents:
  - $\Delta_i = \{\langle \varphi_i^1, w_i^1 \rangle, \ldots, \langle \varphi_i^{k_i}, w_i^{k_i} \rangle\}$ is the set of requests from agent i ;
  - $\{o_i^1, \ldots, o_i^{p_i}\}$ is the set of variables (objects) appearing in the formulas in Δ_i ;
  - each object o_i^k is an **single image request**, simply specified by a starting time $s(o_i^k)$ and a length $l(o_i^k)$.
- A set of **transition times** between objects: for each pair o_i^j, o_k^l, the transition time is written $t(o_i^j, o_k^l)$.

A model of the problem

- **Variables**:
  - one boolean variable $\mathbf{x}_i^j$ for each object o_i^j.
  - one boolean variable $\mathbf{y}_i^j$ for each formula φ_i^j.
  - one integer variable $\mathbf{u}_i$ per agent $i \in \mathcal{N}$.
- **Constraints**:
  - transition constraints: for each o_i^j, o_k^l such that $t(o_i^j, o_k^l) > s(o_k^l) - l(o_i^j) - s(o_i^j)$ and $t(o_k^l, o_i^j) > s(o_i^j) - l(o_k^l) - s(o_k^l)$, $\mathbf{x}_i^j + \mathbf{x}_k^l \leq 1$.
  - logical formulas: use logical constraints or use min ($\vee$) or max ($\wedge$) constraints.
  - individual utility computation: for each agent i, $\mathbf{u}_i = \sum_{\langle \varphi_i^j, w_i^j \rangle \in \Delta_i} w_i^j \times \mathbf{y}_i^j$ (linear constraints)
  - collective utility computation $\rightsquigarrow$?
    - egalitarian (min) and utilitarian CUF $\rightsquigarrow$ easy.
    - OWA $\rightsquigarrow$ a bit more tricky.
    - leximin $\rightsquigarrow$ a bit more tricky than OWA.
    - generalized averages $\rightsquigarrow$ hard to encode.

Multicriteria optimisation...

- Several criteria to optimize (maximize) : $\langle u_1, \ldots, u_n \rangle$.
- Goal : **balance** the criteria, while preserving a kind of **efficiency** (they have to be maximized).
- Exemples :
  - fair resource allocation (fair share among agents) ;
  - balanced time-tables ;
  - balanced job assignement,...

Encoding the leximin preorder with a collective utility function

In the general case, there is **no** CUF g such that

$$\overrightarrow{u} \preceq_{leximin} \overrightarrow{v} \Leftrightarrow g(\overrightarrow{u}) \leq g(\overrightarrow{v}).$$

Encoding the leximin preorder with a collective utility function

In the general case, there is **no** CUF g such that

$$\overrightarrow{u} \preceq_{leximin} \overrightarrow{v} \Leftrightarrow g(\overrightarrow{u}) \leq g(\overrightarrow{v}).$$

However, when the set of utility profiles is finite :

- $\overrightarrow{u} \mapsto \sum_{i=1}^{n} n^{-u_i}$,
- $\overrightarrow{u} \mapsto -\sum_{i=1}^{n} u_i^{-q}$, with a very high q,
- $\overrightarrow{u} \mapsto \sum_{i=1}^{n} w_i \times u_i^{\uparrow}$, with $w_1 >> w_2 >> \cdots >> w_n$ (OWA).

Encoding the leximin preorder with a collective utility function

In the general case, there is **no** CUF g such that

$$\overrightarrow{u} \preceq_{leximin} \overrightarrow{v} \Leftrightarrow g(\overrightarrow{u}) \leq g(\overrightarrow{v}).$$

However, when the set of utility profiles is finite :

- $\overrightarrow{u} \mapsto \sum_{i=1}^{n} n^{-u_i}$,
- $\overrightarrow{u} \mapsto -\sum_{i=1}^{n} u_i^{-q}$, with a very high q,
- $\overrightarrow{u} \mapsto \sum_{i=1}^{n} w_i \times u_i^{\uparrow}$, with $w_1 >> w_2 >> \cdots >> w_n$ (OWA).

However, any g representing the leximin preorder on $[\![1, m]\!]^n$ is such that:

$$Card(g([\![1, m]\!]^n)) = \frac{(m+n+1)!}{(m-1)!n!} = O_{m \to +\infty}(m^n).$$

The Constraint Satisfaction Problem

Classical CSP

Given : A constraint network $(\mathscr{X}, \mathscr{D}, \mathscr{C})$.
Is there a complete consistent instantiation v of $(\mathscr{X}, \mathscr{D}, \mathscr{C})$?

CSP with objective variable

Given : A constraint network $(\mathscr{X}, \mathscr{D}, \mathscr{C})$ and an objective variable $\mathbf{o} \in \mathscr{X}$, such that $\mathscr{D}_{\mathbf{o}} \subset \mathbb{N}$.
What is the maximal value α of $\mathscr{D}_{\mathbf{o}}$ such that there is a complete consistent instantiation $\widehat{v}$ with $\widehat{v}(\mathbf{o}) = \alpha$?

Leximin-CSP (as a multi-objective CSP)

Given : A constraint network $(\mathscr{X}, \mathscr{D}, \mathscr{C})$ and a vector of variables $\overrightarrow{u} = \langle \mathbf{u_1}, \ldots, \mathbf{u_n} \rangle$ ($\forall i,\ \mathbf{u_i} \in \mathscr{X}$ and $\mathscr{D}_{\mathbf{u_i}} \in \mathbb{N}$) called **objective vector**.
What is the leximin-optimal vector $\langle \alpha_1, \ldots, \alpha_n \rangle$ of $\langle \mathscr{D}_{\mathbf{u_1}}, \ldots, \mathscr{D}_{\mathbf{u_n}} \rangle$ such that there is a complete consistent instantiation $\widehat{v}$ with $\widehat{v}(\mathbf{u_i}) = \alpha_i$ forall i ?

Algorithm 1

Sort and conquer

Sorted version of the objective vector

Initial idea

Maximize the objective vector using the leximin preorder $\Leftrightarrow$ maximize the successive components of the **ordered** objective vector.

$\rightsquigarrow$ We have to introduce the sorted version of the objective vector:

- **A vector of variables** $(\mathbf{y_1}, \ldots, \mathbf{y_n})$.
- **A constraint Sort**$(\overrightarrow{\mathbf{u}}, \overrightarrow{\mathbf{y}})$ ([Mehlhorn and Thiel, 2000] (filtering in time $O(n \log(n))$).

Mehlhorn, K. and Thiel, S. (2000).
Faster algorithms for bound-consistency of the sortedness and the alldifferent constraint.
In Dechter, R., editor, *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming (CP-00)*, pages 306–319, Singapore. Springer.

Sorted version of the objective vector

Initial idea

Maximize the objective vector using the leximin preorder $\Leftrightarrow$ maximize the successive components of the **ordered** objective vector.

$\rightsquigarrow$ We have to introduce the sorted version of the objective vector:

- **A vector of variables** $(\mathbf{y_1}, \ldots, \mathbf{y_n})$.
- **A constraint Sort**$(\overrightarrow{\mathbf{u}}, \overrightarrow{\mathbf{y}})$ ([Mehlhorn and Thiel, 2000] (filtering in time $O(n \log(n))$).

1. Maximize $\mathbf{y_1}$: $\widehat{y_1}$.

Mehlhorn, K. and Thiel, S. (2000).
Faster algorithms for bound-consistency of the sortedness and the alldifferent constraint.
In Dechter, R., editor, *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming (CP-00)*, pages 306–319, Singapore. Springer.

Sorted version of the objective vector

Initial idea

Maximize the objective vector using the leximin preorder $\Leftrightarrow$ maximize the successive components of the **ordered** objective vector.

$\rightsquigarrow$ We have to introduce the sorted version of the objective vector:

- **A vector of variables** $(\mathbf{y_1}, \ldots, \mathbf{y_n})$.
- **A constraint Sort**$(\overrightarrow{\mathbf{u}}, \overrightarrow{\mathbf{y}})$ ([Mehlhorn and Thiel, 2000] (filtering in time $O(n \log(n))$).

1. Maximize $\mathbf{y_1}$: $\widehat{y_1}$.
2. Maximize $\mathbf{y_2}$ under the constraint $\mathbf{y_1} = \widehat{y_1}$: $\widehat{y_2}$.

Mehlhorn, K. and Thiel, S. (2000).
Faster algorithms for bound-consistency of the sortedness and the alldifferent constraint.
In Dechter, R., editor, *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming (CP-00)*, pages 306–319, Singapore. Springer.

Sorted version of the objective vector

Initial idea

Maximize the objective vector using the leximin preorder $\Leftrightarrow$ maximize the successive components of the **ordered** objective vector.

$\rightsquigarrow$ We have to introduce the sorted version of the objective vector:

- **A vector of variables** $(\mathbf{y_1}, \ldots, \mathbf{y_n})$.
- **A constraint Sort**$(\overrightarrow{\mathbf{u}}, \overrightarrow{\mathbf{y}})$ ([Mehlhorn and Thiel, 2000] (filtering in time $O(n \log(n))$).

1. Maximize $\mathbf{y_1}$: $\widehat{y_1}$.
2. Maximize $\mathbf{y_2}$ under the constraint $\mathbf{y_1} = \widehat{y_1}$: $\widehat{y_2}$.

$\vdots$

Mehlhorn, K. and Thiel, S. (2000).
Faster algorithms for bound-consistency of the sortedness and the alldifferent constraint.
In Dechter, R., editor, *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming (CP-00)*, pages 306–319, Singapore. Springer.

Sorted version of the objective vector

Initial idea

Maximize the objective vector using the leximin preorder $\Leftrightarrow$ maximize the successive components of the **ordered** objective vector.

$\rightsquigarrow$ We have to introduce the sorted version of the objective vector:

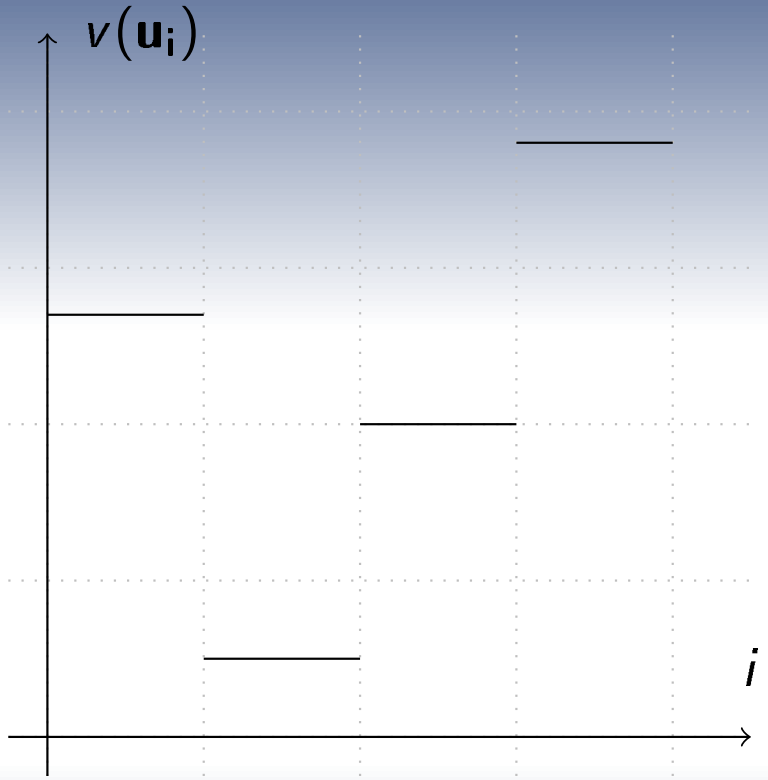
- **A vector of variables** $(\mathbf{y_1}, \ldots, \mathbf{y_n})$.
- **A constraint Sort**$(\overrightarrow{\mathbf{u}}, \overrightarrow{\mathbf{y}})$ ([Mehlhorn and Thiel, 2000] (filtering in time $O(n \log(n))$).

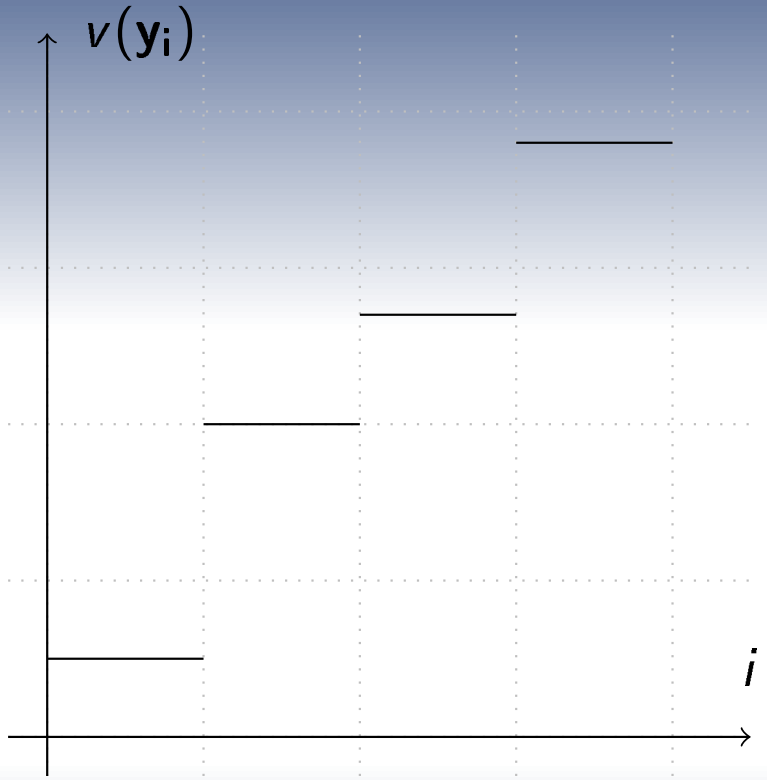
1. Maximize $\mathbf{y_1}$: $\widehat{y_1}$.
2. Maximize $\mathbf{y_2}$ under the constraint $\mathbf{y_1} = \widehat{y_1}$: $\widehat{y_2}$.

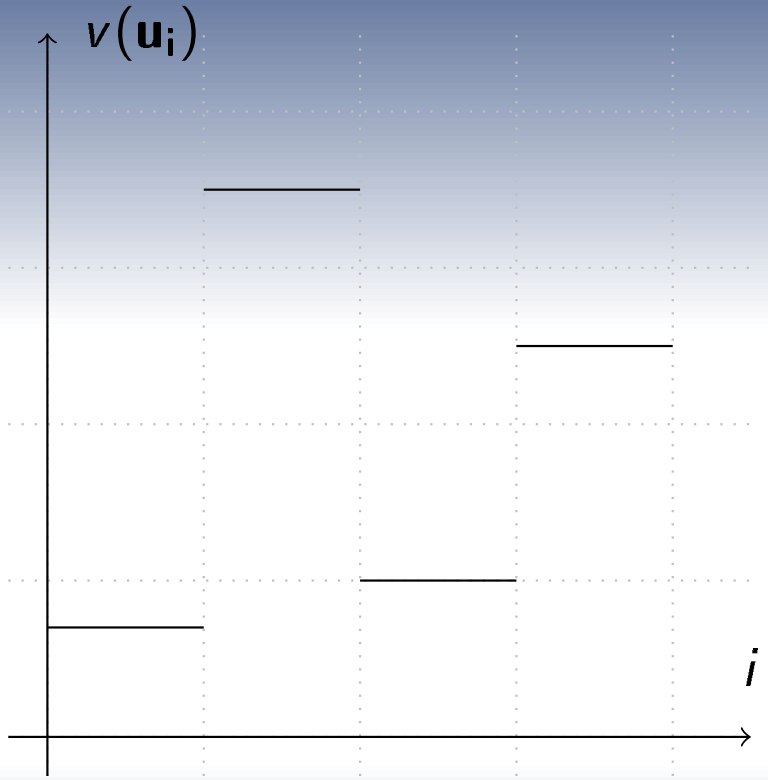
$\vdots$

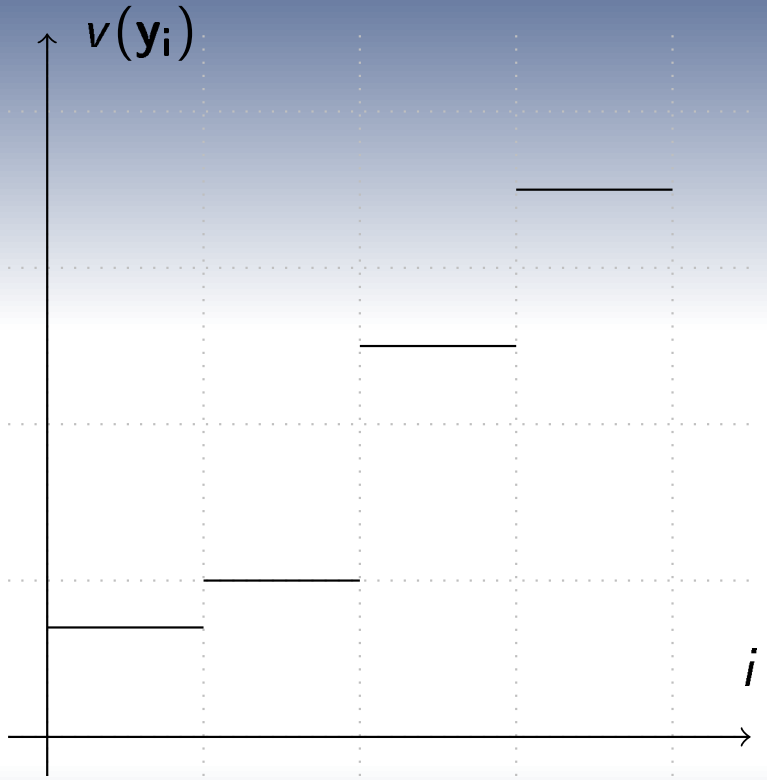
n. Maximize $\mathbf{y_n}$ under the constraints $\mathbf{y_1} = \widehat{y_1}, \ldots, \mathbf{y_{n-1}} = \widehat{y_{n-1}}$.

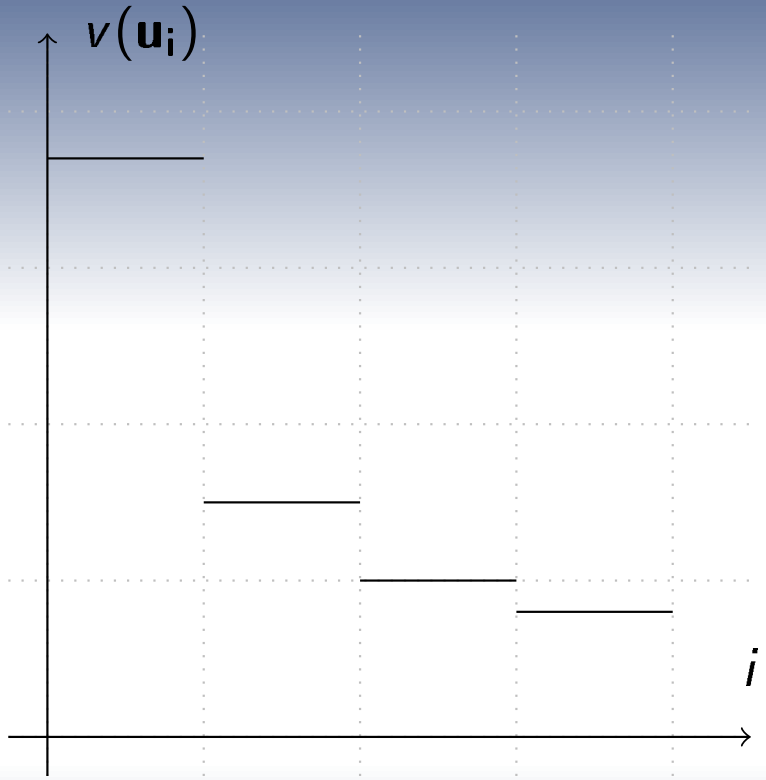
Mehlhorn, K. and Thiel, S. (2000).
Faster algorithms for bound-consistency of the sortedness and the alldifferent constraint.
In Dechter, R., editor, *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming (CP-00)*, pages 306–319, Singapore. Springer.

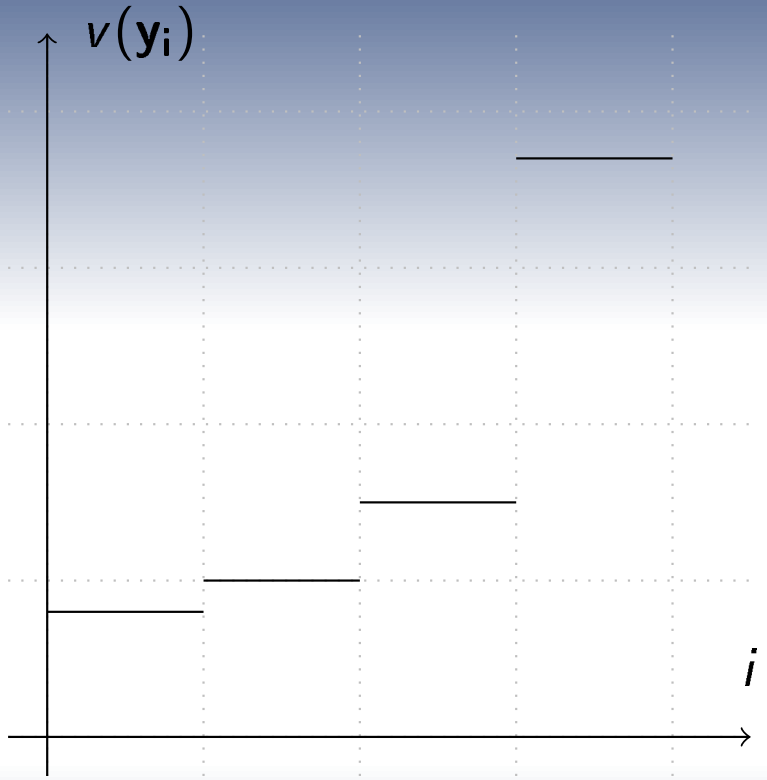

$v(\mathbf{u}_i)$
i

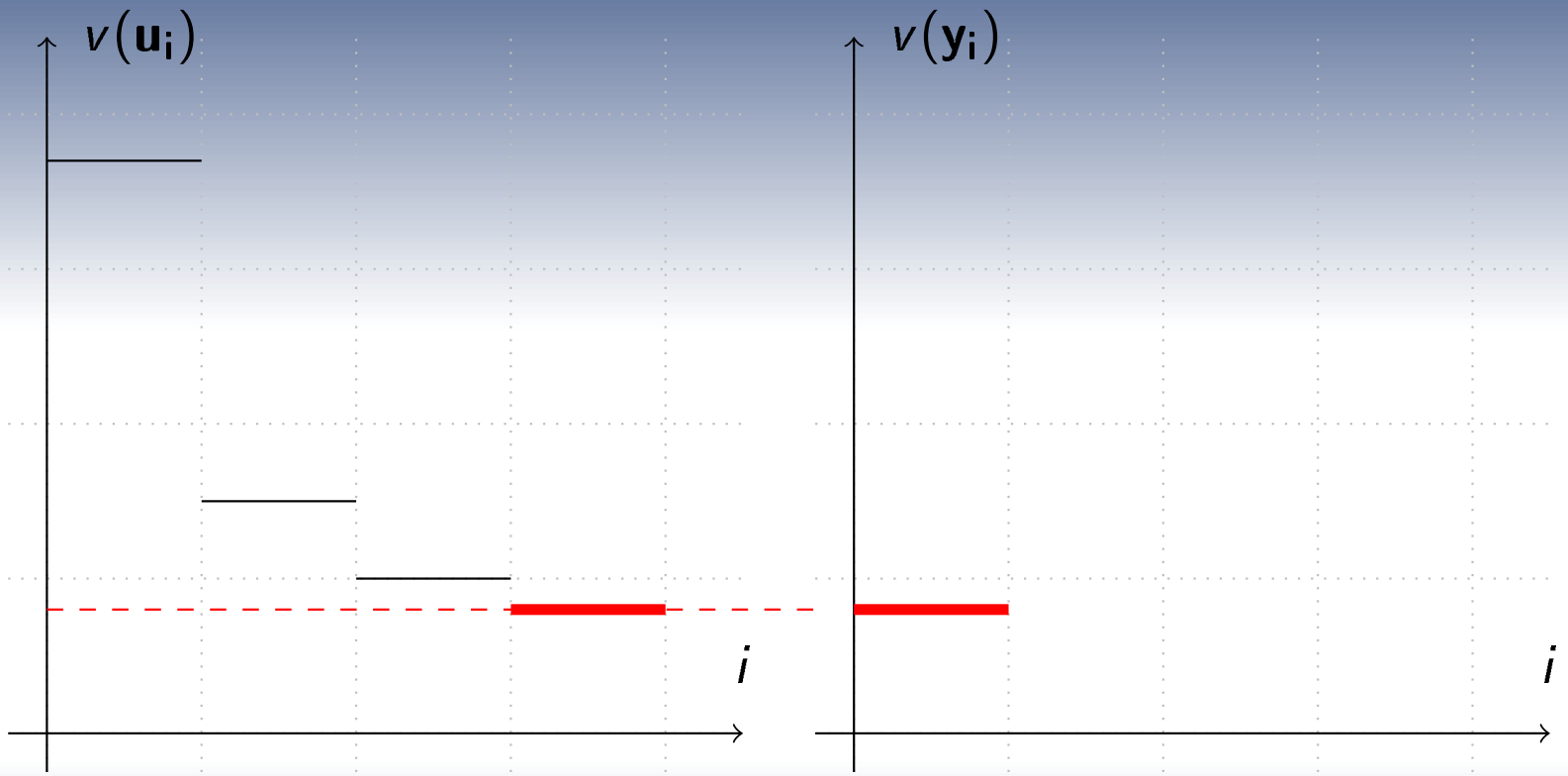

$v(\mathbf{y}_i)$
i

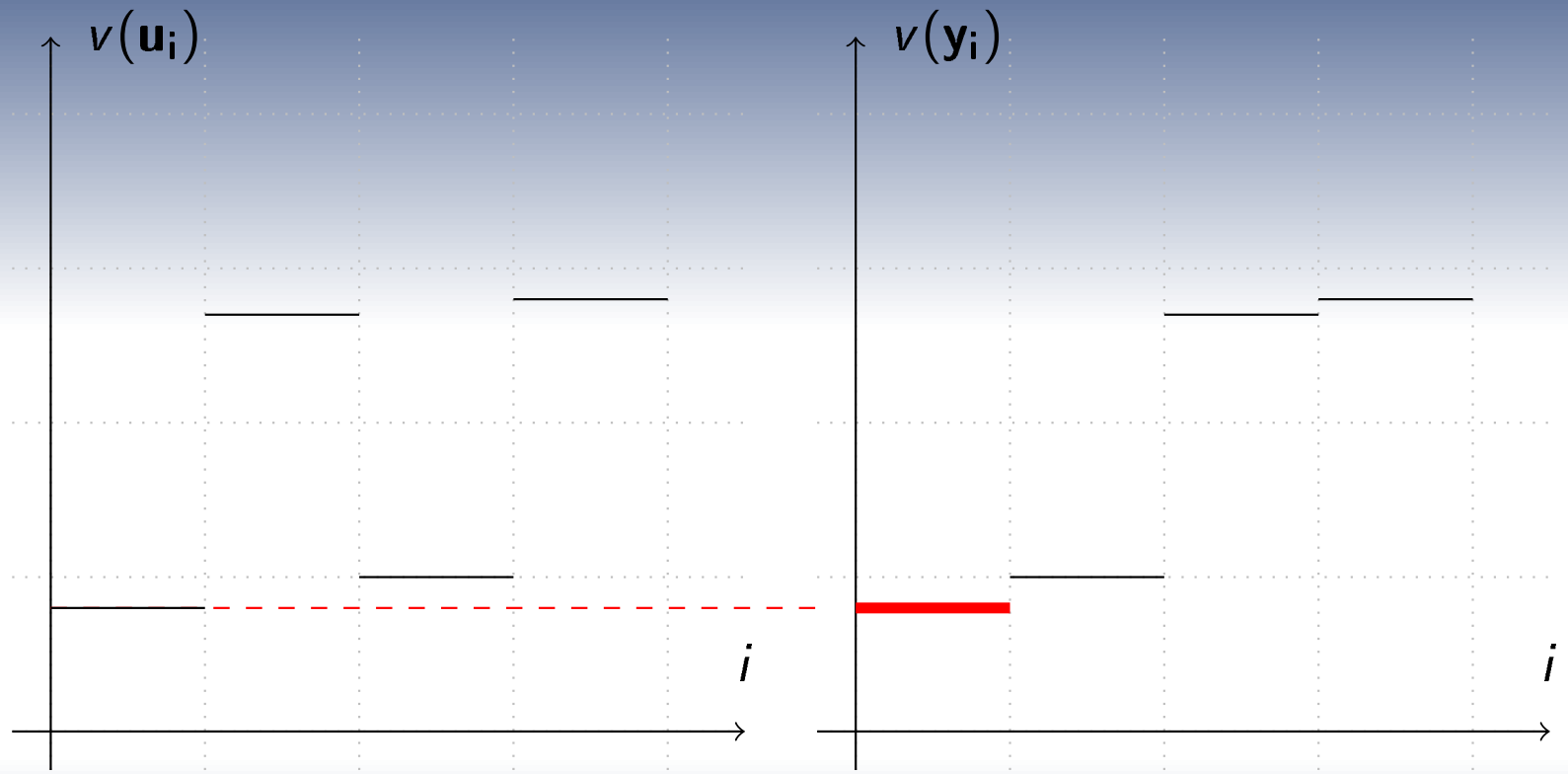

$v(\mathbf{u}_i)$
i

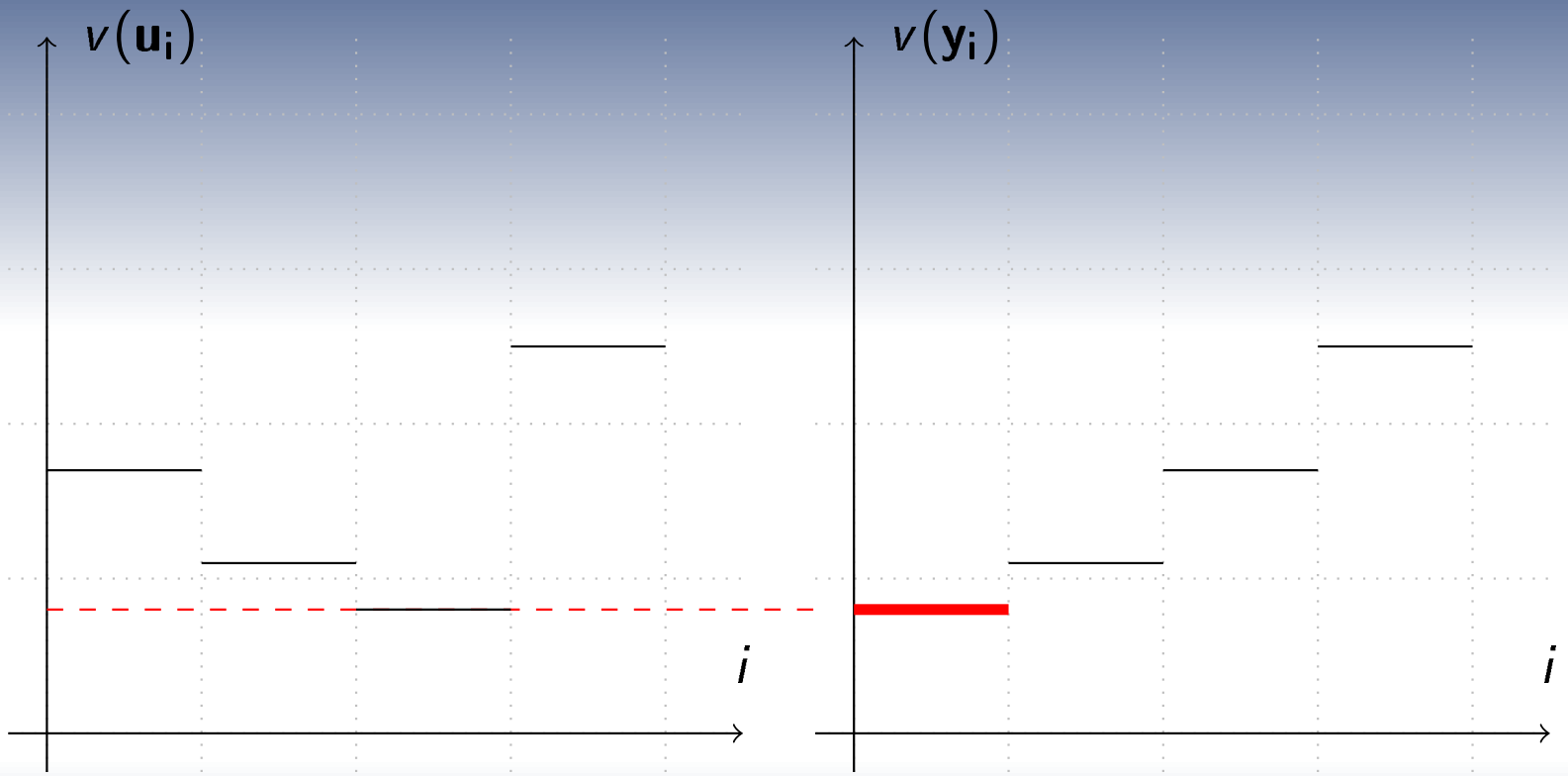

$v(\mathbf{y}_i)$
i

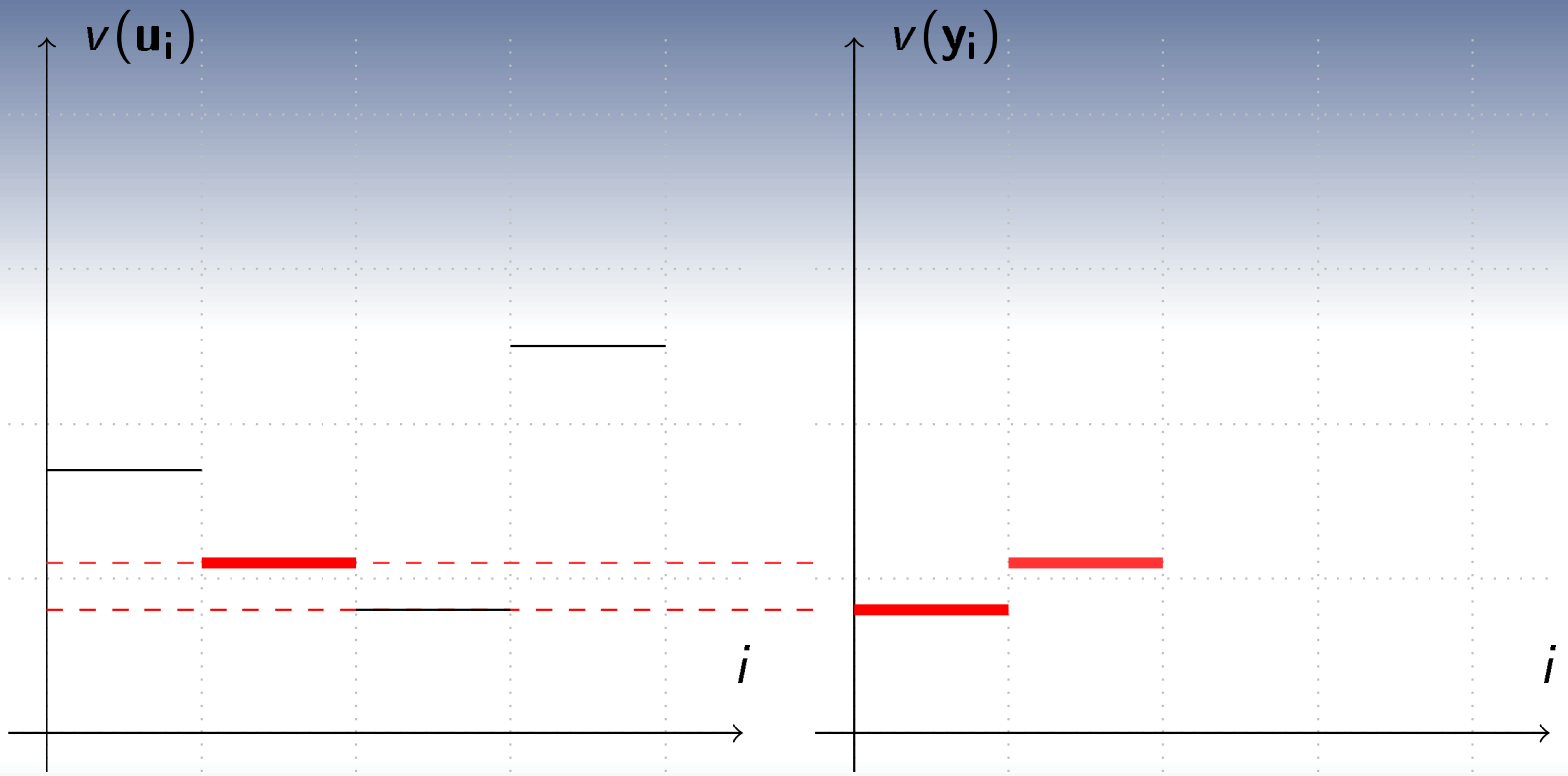

$v(\mathbf{u}_i)$
i

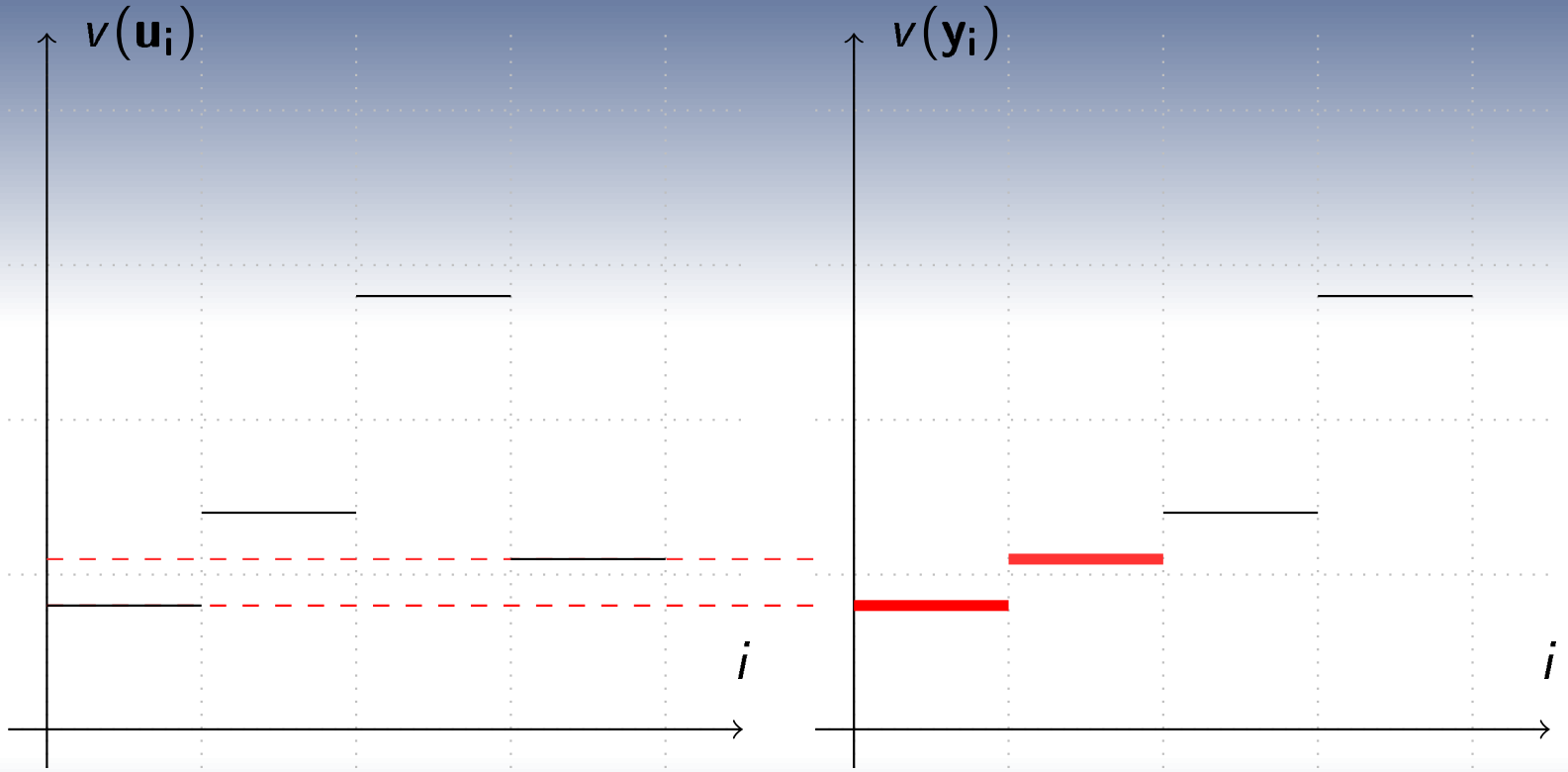

$v(\mathbf{y}_i)$
i

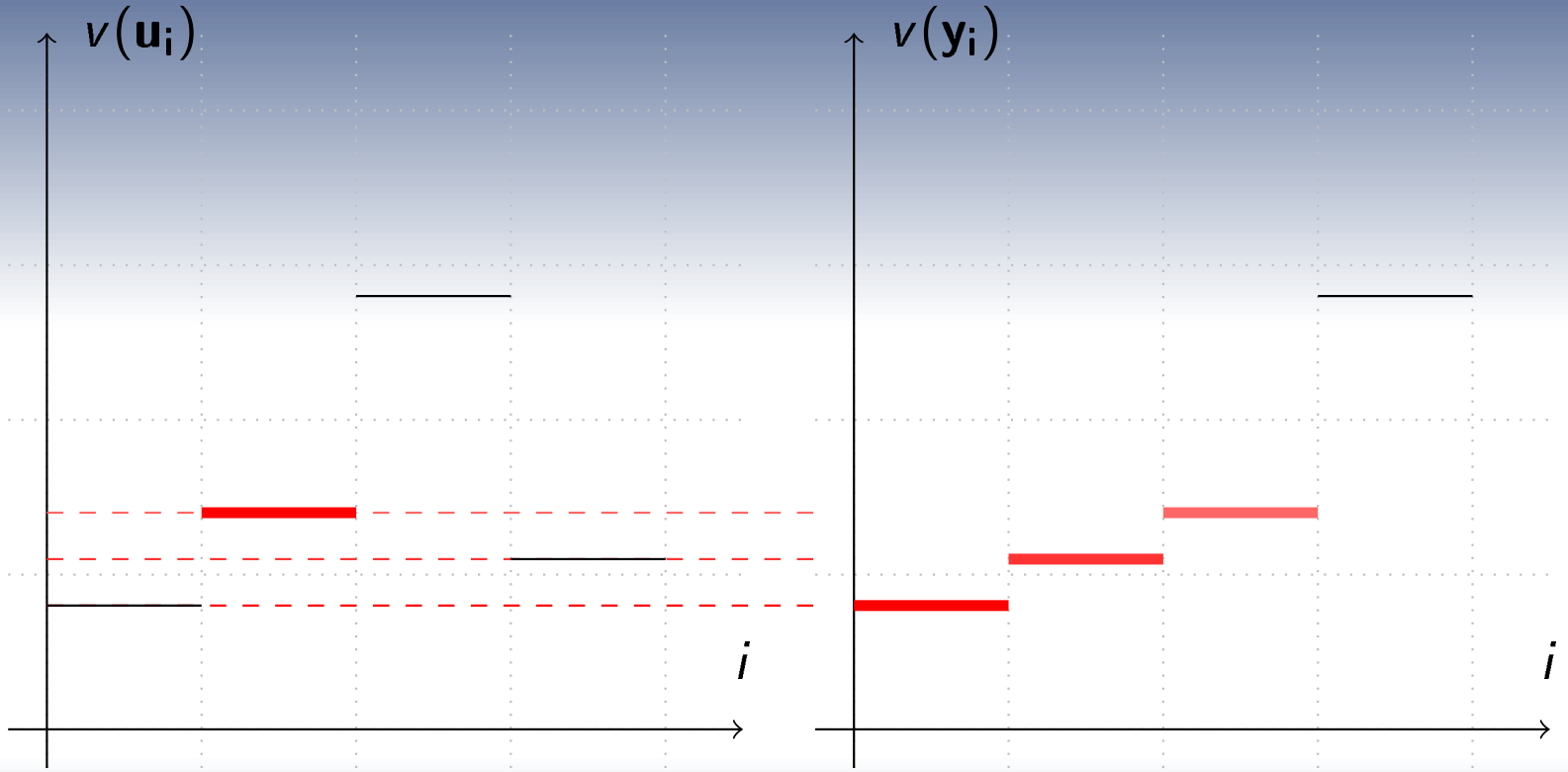

$v(\mathbf{u}_i)$
i
$v(\mathbf{y}_i)$
i

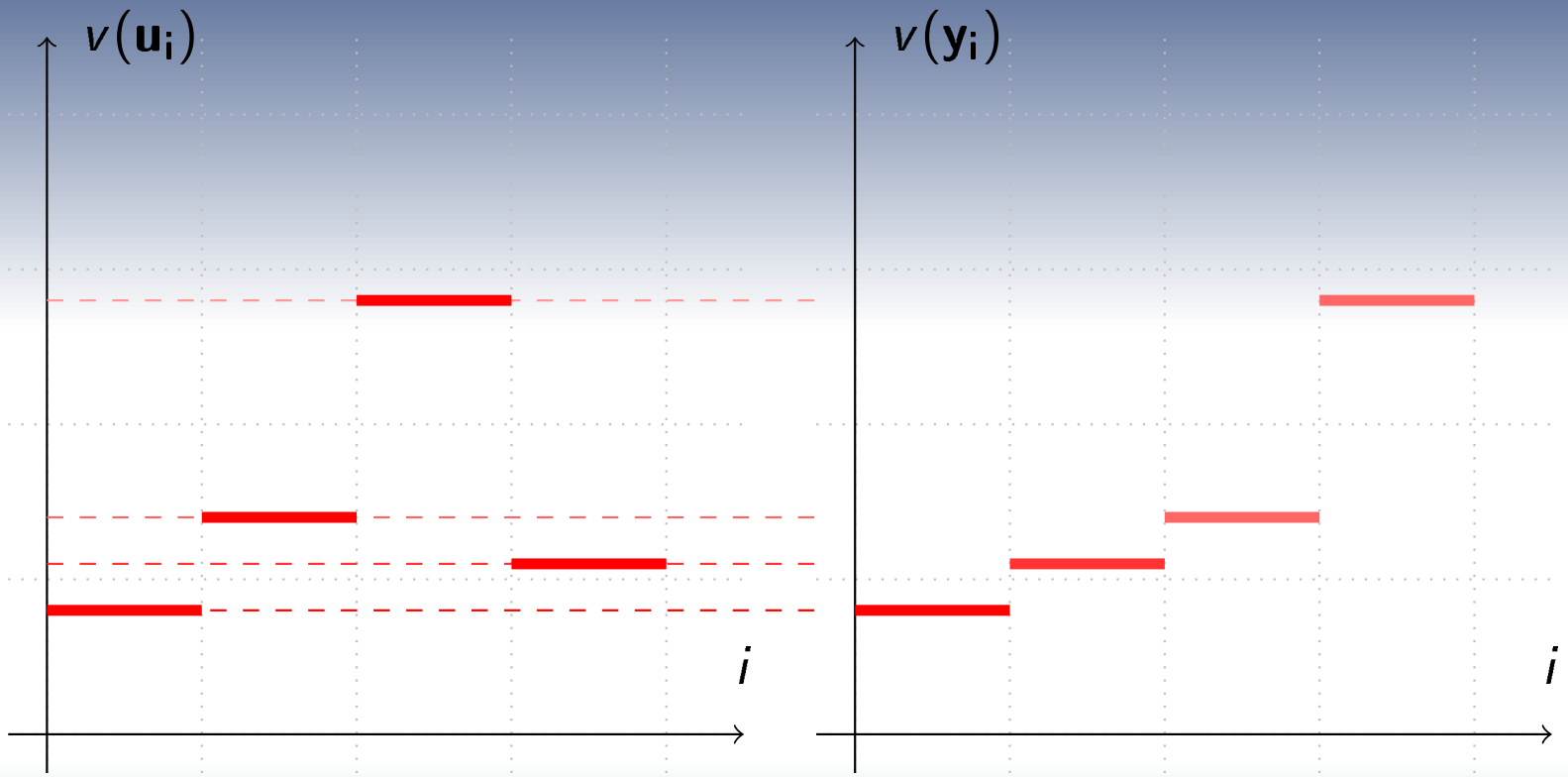

$v(\mathbf{u}_i)$
i
$v(\mathbf{y}_i)$
i


$v(\mathbf{u}_i)$
i
$v(\mathbf{y}_i)$
i


$v(\mathbf{u}_i)$
i
$v(\mathbf{y}_i)$
i


$v(\mathbf{u}_i)$
i
$v(\mathbf{y}_i)$
i


$v(\mathbf{u}_i)$
i
$v(\mathbf{y}_i)$
i


$v(\mathbf{u}_i)$
i
$v(\mathbf{y}_i)$
i

Algorithm 2

Using cardinality combinators...

An alternative definition of sorting...

Proposition

$\langle y_1, \ldots, y_n \rangle$ is the permutation of $\langle u_1, \ldots, u_n \rangle$ sorted in non-decreasing order if and only if:

- y_1 is the maximal value such that all the u_i are g.eq to y_1;

An alternative definition of sorting...

Proposition

$\langle y_1, \ldots, y_n \rangle$ is the permutation of $\langle u_1, \ldots, u_n \rangle$ sorted in non-decreasing order if and only if:

- y_1 is the maximal value such that all the u_i are g.eq to y_1;
- y_2 is the maximal value such that at least $n - 1$ values among the u_i are g.eq to y_2;

An alternative definition of sorting...

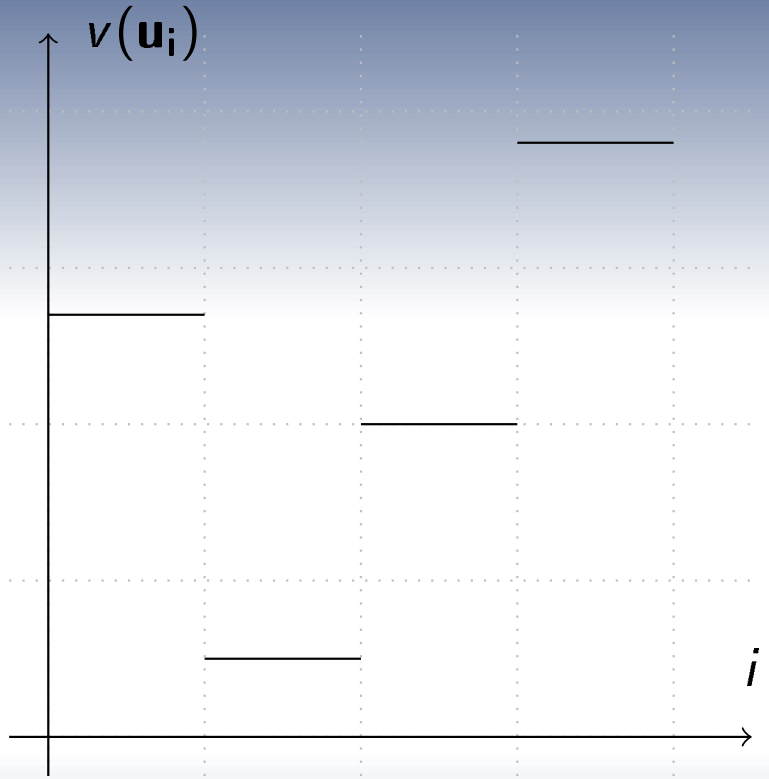
Proposition

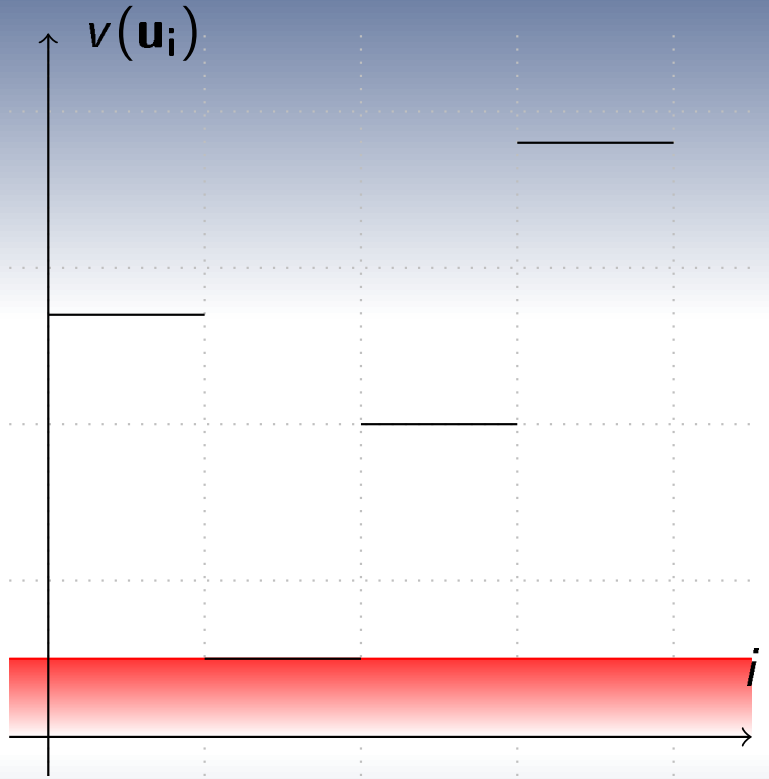
$\langle y_1, \ldots, y_n \rangle$ is the permutation of $\langle u_1, \ldots, u_n \rangle$ sorted in non-decreasing order if and only if:

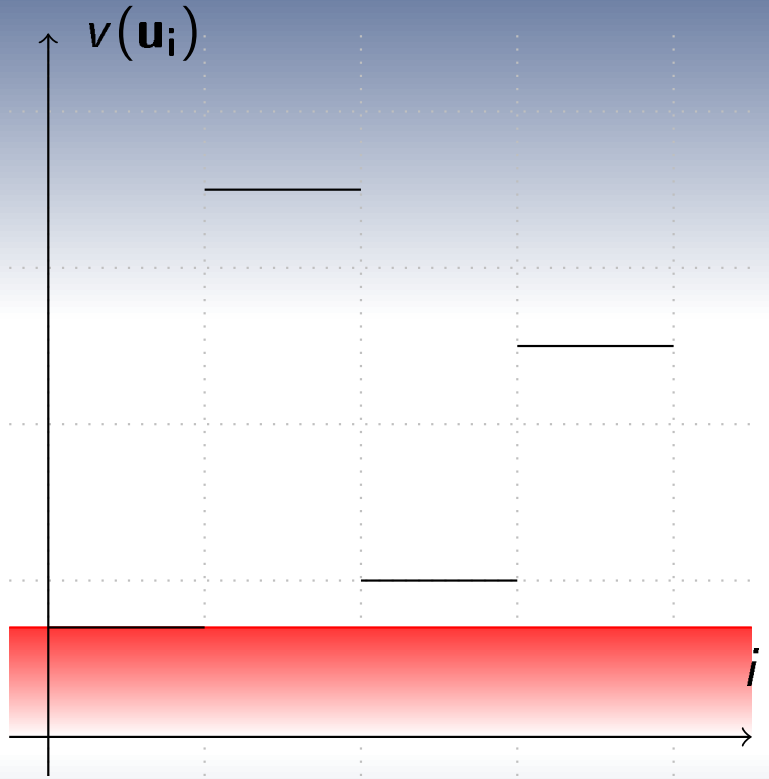
- y_1 is the maximal value such that all the u_i are g.eq to y_1;
- y_2 is the maximal value such that at least $n-1$ values among the u_i are g.eq to y_2;

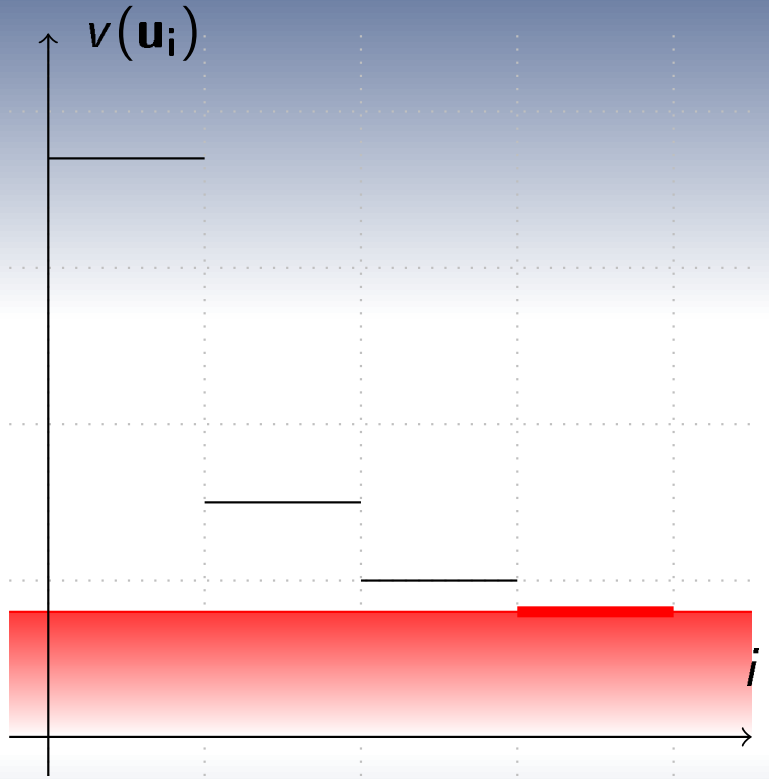
$$\vdots$$

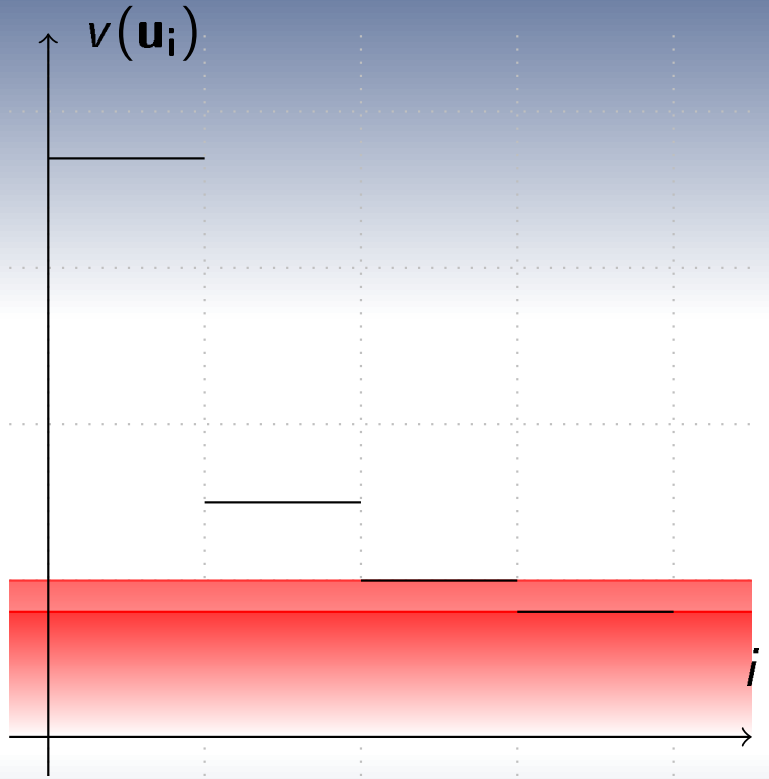
- y_n is the maximal value such that at least 1 value among the u_i is g.eq to y_n.

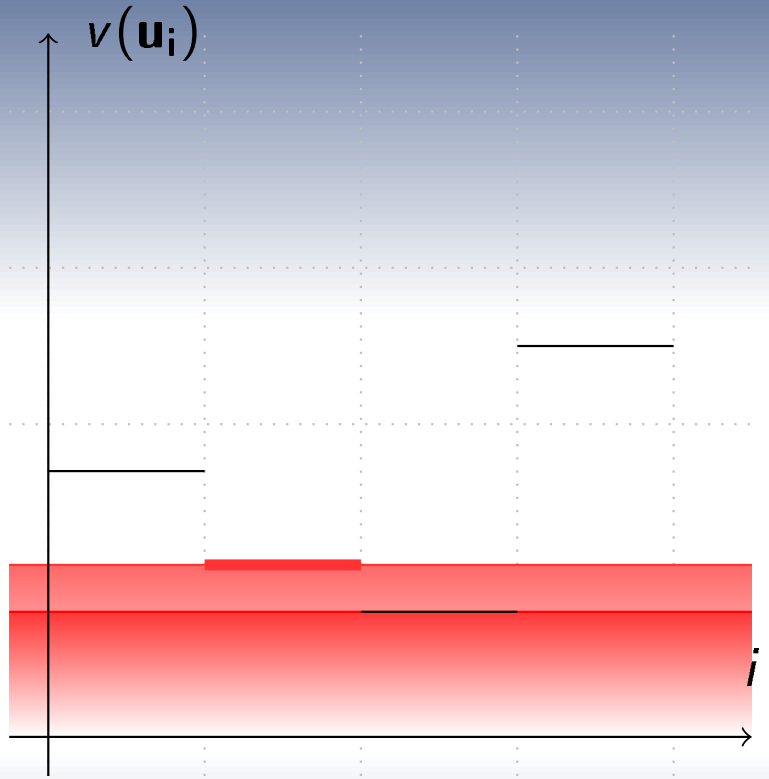

$v(\mathbf{u}_i)$
i

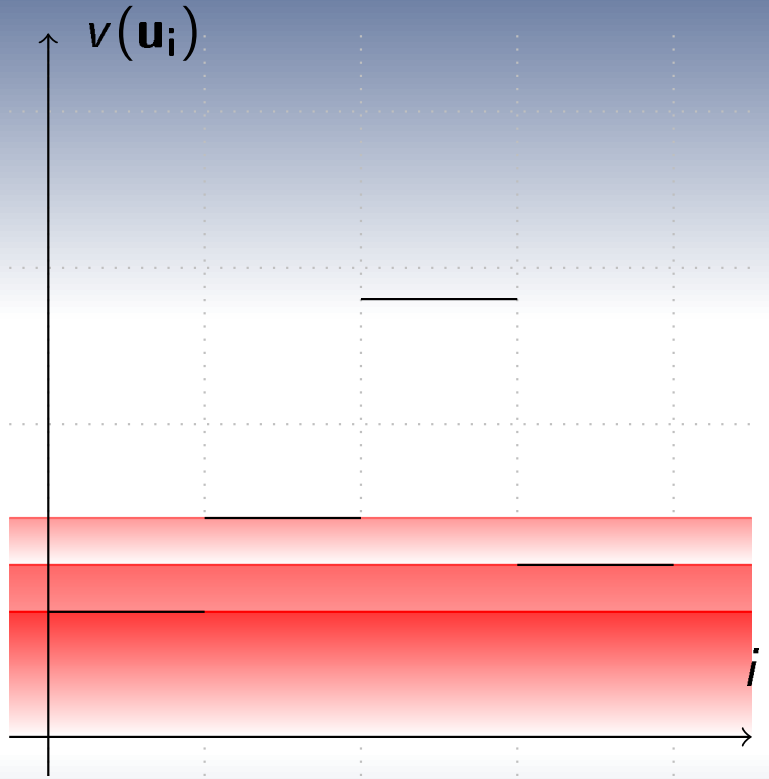

$v(\mathbf{u}_i)$
i

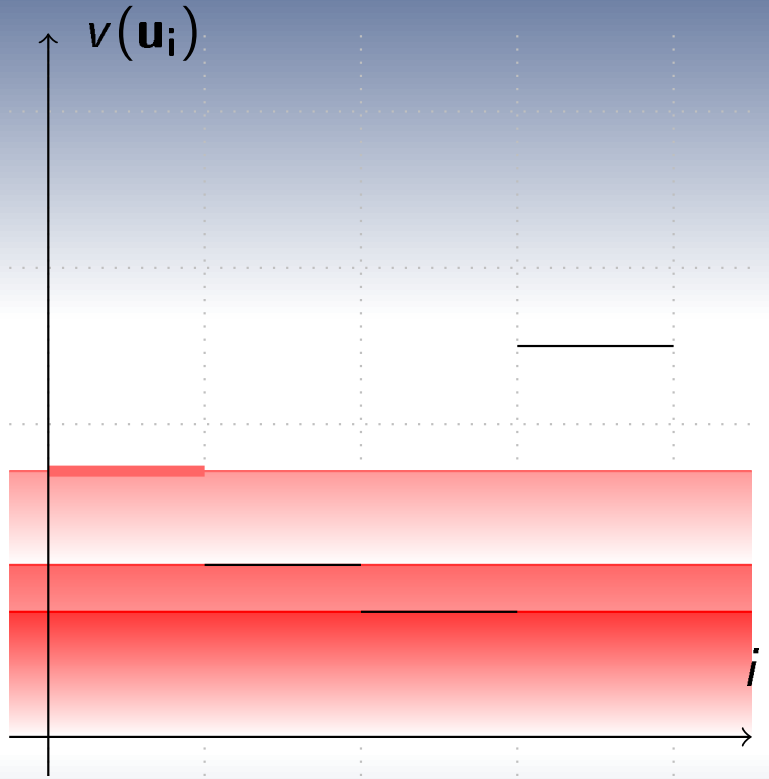

$v(\mathbf{u}_i)$
i

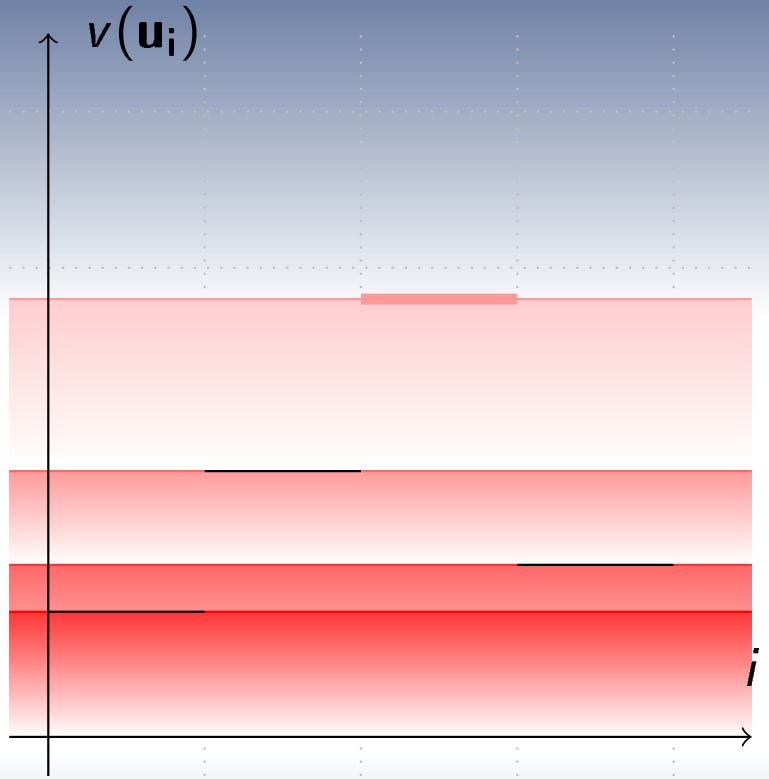

$v(\mathbf{u}_i)$
i


$v(\mathbf{u}_i)$
i


$v(\mathbf{u}_i)$
i


$v(\mathbf{u}_i)$
i


$v(\mathbf{u}_i)$
i


$v(\mathbf{u}_i)$
i

The meta-constraint AtLeast

"y_i is the maximal value such that at least $n - i + 1$ values among the u_i are g.e.q than y_i"

$\rightsquigarrow$ a particular cardinality meta-constraint [van Hentenryck et al., 1992]:

$$\textbf{AtLeast}(\{\mathbf{y_i} \geq u_1, \ldots, \mathbf{y_i} \geq u_n\}, n - i + 1)$$

van Hentenryck, P., Simonis, H., and Dincbas, M. (1992).
Constraint satisfaction using constraint logic programming.
Artificial Intelligence, 58(1-3):113–159.

The meta-constraint AtLeast

"y_i is the maximal value such that at least $n - i + 1$ values among the u_i are g.e.q than y_i"

$\rightsquigarrow$ a particular cardinality meta-constraint [van Hentenryck et al., 1992]:

$$\textbf{AtLeast}(\{\mathbf{y_i} \geq u_1, \ldots, \mathbf{y_i} \geq u_n\}, n - i + 1)$$

- A specific filtering algorithm running in $O(n)$.
- A possible implementation using linear constraints.

van Hentenryck, P., Simonis, H., and Dincbas, M. (1992).
Constraint satisfaction using constraint logic programming.
Artificial Intelligence, 58(1-3):113–159.

Algorithm 3

A branch-and-bound-like algorithm

A branch-and-bound-like algorithm

The classical branch-and-bound (integral criterion):

- A branching algorithm (exploration of the search tree).

A branch-and-bound-like algorithm

The classical branch-and-bound (integral criterion):

- A branching algorithm (exploration of the search tree).
- A lower bound on the criterion to maximize.

A branch-and-bound-like algorithm

The classical branch-and-bound (integral criterion):

- A branching algorithm (exploration of the search tree).
- A lower bound on the criterion to maximize.
- An upper bound and a pruning mechanism ($ub \leq lb$).

A branch-and-bound-like algorithm

The classical branch-and-bound (integral criterion):

- A branching algorithm (exploration of the search tree).
- A lower bound on the criterion to maximize.
- An upper bound and a pruning mechanism ($ub \leq lb$).

Our algorithm (vectorial criterion with leximin preorder):

- Branching algorithm given by the constraint solver (call to **solve**).

A branch-and-bound-like algorithm

The classical branch-and-bound (integral criterion):

- A branching algorithm (exploration of the search tree).
- A lower bound on the criterion to maximize.
- An upper bound and a pruning mechanism ($ub \leq lb$).

Our algorithm (vectorial criterion with leximin preorder):

- Branching algorithm given by the constraint solver (call to **solve**).
- Lower bound: the objective vector of the last solution found.

A branch-and-bound-like algorithm

The classical branch-and-bound (integral criterion):

- A branching algorithm (exploration of the search tree).
- A lower bound on the criterion to maximize.
- An upper bound and a pruning mechanism ($ub \leq lb$).

Our algorithm (vectorial criterion with leximin preorder):

- Branching algorithm given by the constraint solver (call to **solve**).
- Lower bound: the objective vector of the last solution found.
- Pruning mechanism given by a filtering procedure associated to the leximin preorder (we reject every solution whose objective vector is leximin-lower than the lower bound).

A constraint Leximin

We use a constraint **Leximin**: **Leximin**$(\overrightarrow{\lambda}, \overrightarrow{\mathbf{x}})$ (the vector $\overrightarrow{\mathbf{x}}$ must be leximin-greater than the integer vector $\overrightarrow{\lambda}$)

This constraint is based on the constraint **Multiset Ordering**, introduced in [Frisch et al., 2003] (filtering in $O(n \log(n))$).

Frisch, A. M., Hnich, B., Kiziltan, Z., Miguel, I., and Walsh, T. (2003).
Multiset ordering constraints.
In Gottlob, G. and Walsh, T., editors, *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, Acapulco, Mexico. Morgan Kaufmann.

Algorithm 4

Using the saturated subsets

Principle of the algorithm

This algorithm comes from the litterature on fuzzy CSPs [Dubois and Fortemps, 1999].

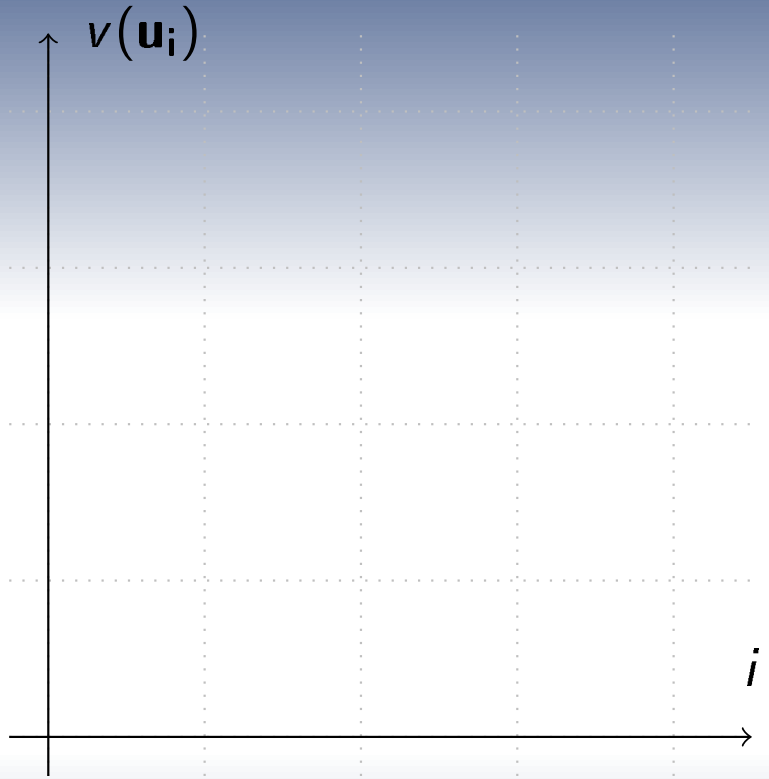
Saturated subset

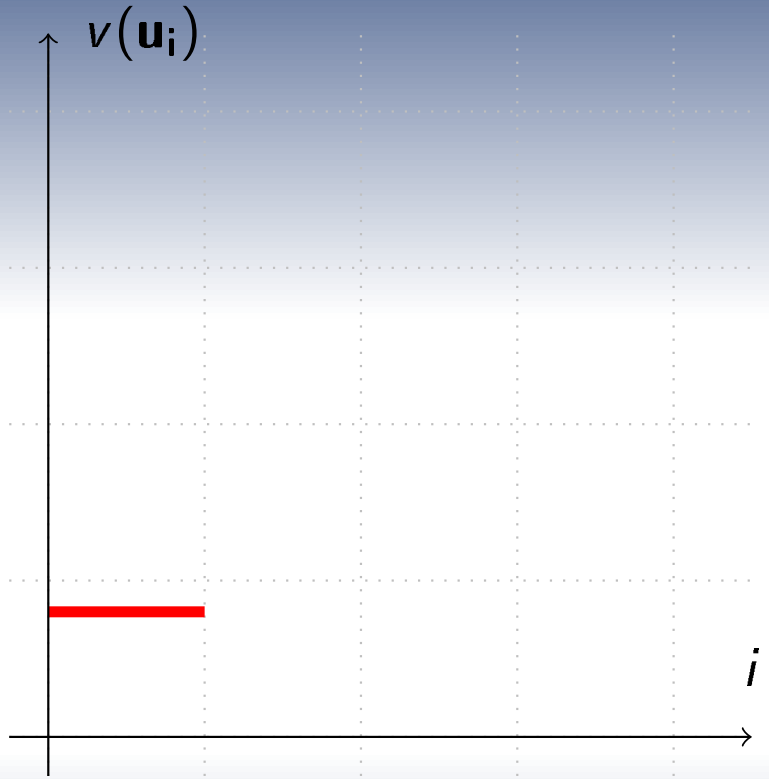
Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a constraint networtk, $\overrightarrow{u}$ be a set of objective variables, and α be the maximal admissible value of $\min(\overrightarrow{u})$.
A subset $\mathcal{S}$ of variables from $\overrightarrow{u}$ is said **saturated** if there is a solution with all $u_i \in \mathcal{S}$ instantiated to α and all $u_i \notin \mathcal{S}$ instantiated to a strictly greater value than α.

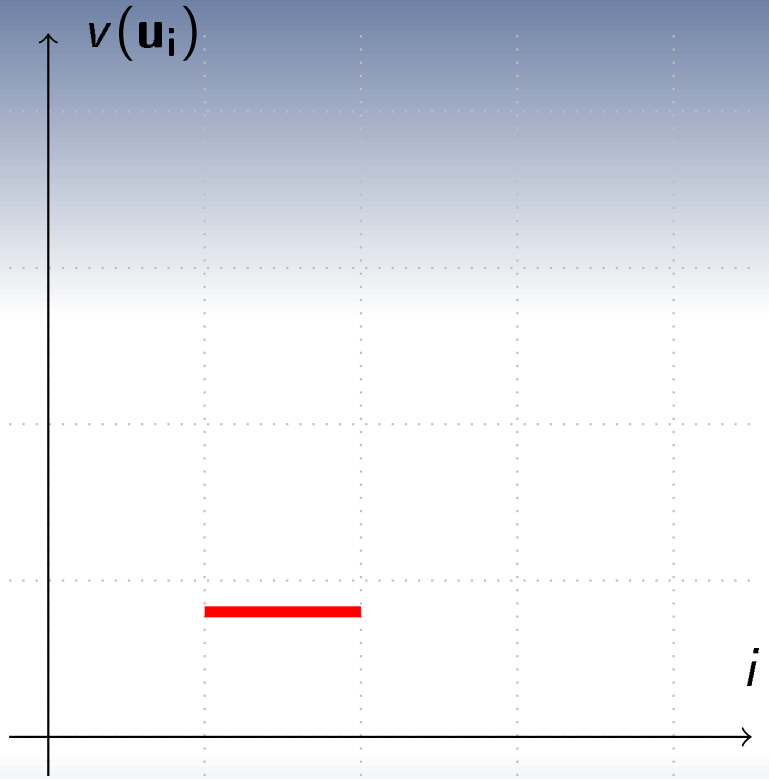
Dubois, D. and Fortemps, P. (1999).
Computing improved optimal solutions to max-min flexible constraint satisfaction problems.
European Journal of Operational Research.

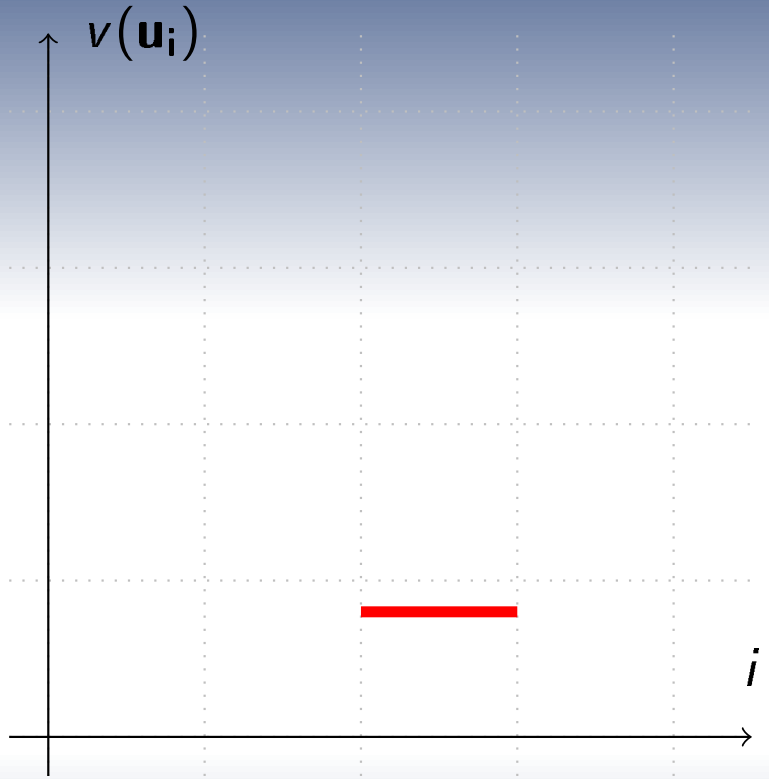
Principle of the algorithm

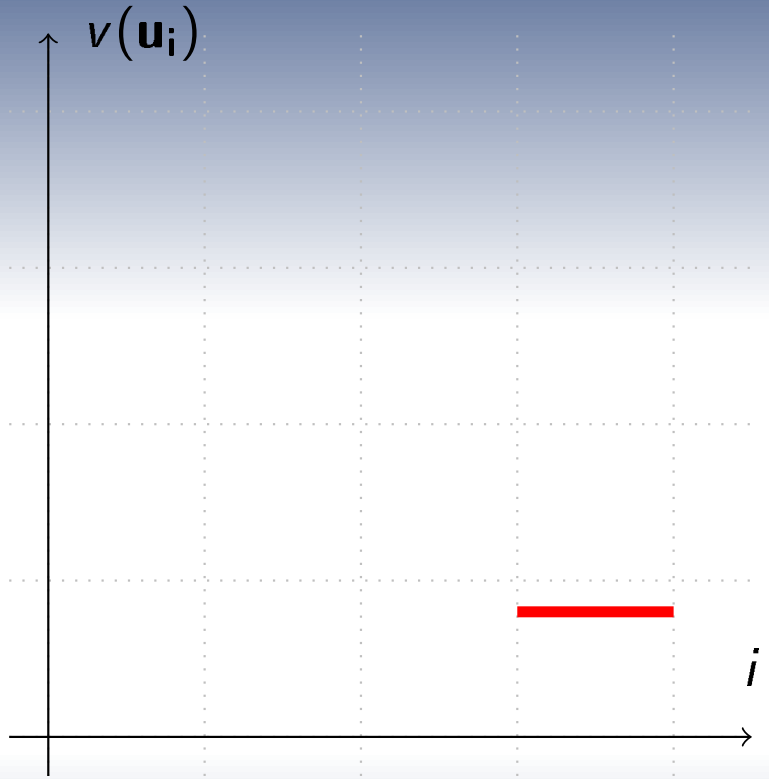
- At each step of the algorithm, we maximize the current component y_i of the leximin vector (as in the previous algorithms).
- For each cardinality-minimal saturated subset $\mathcal{S}$:
  - all the $u_i \in \mathcal{S}$ are fixed to $\widehat{y_i}$ and are removed from $\overrightarrow{u}$,
  - we restart the procedure on the new constraint network until no variable remains in $\overrightarrow{u}$.
- At the end, we compare all the potential leximin-optimal vectors found.

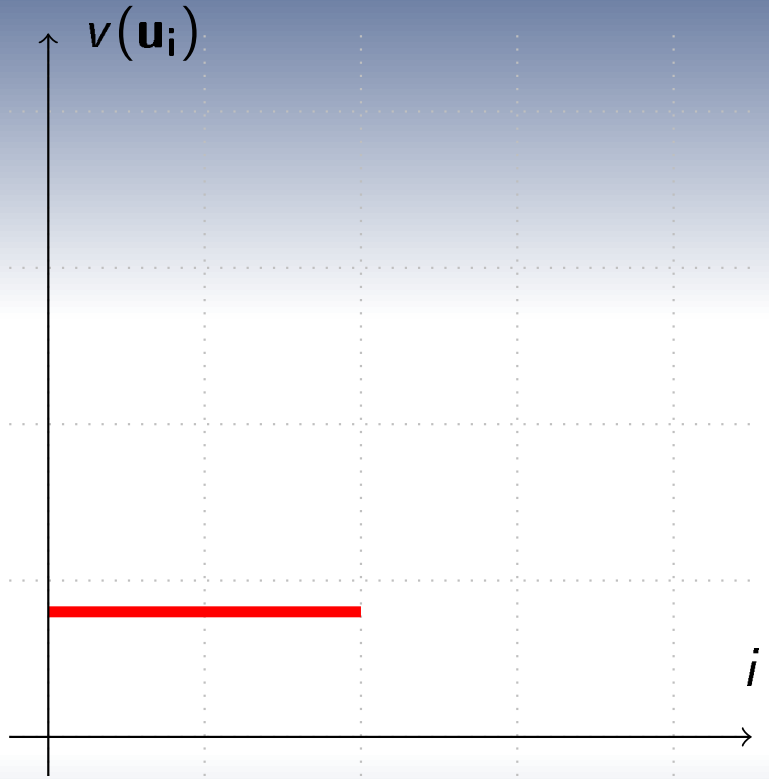

$v(\mathbf{u_i})$
i

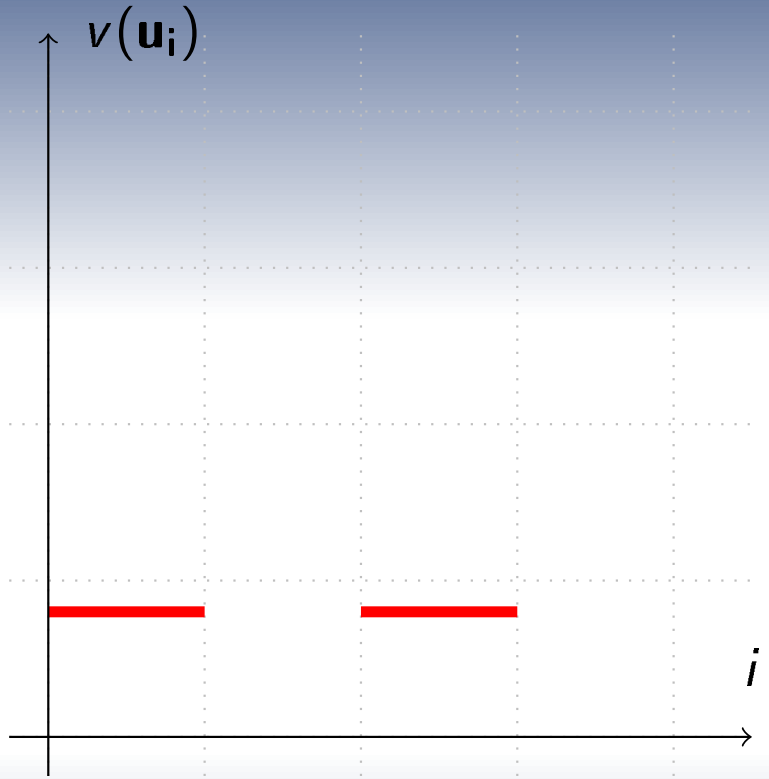

$v(\mathbf{u}_i)$
i


$v(\mathbf{u_i})$
i


$v(\mathbf{u_i})$
i


$v(\mathbf{u_i})$
i

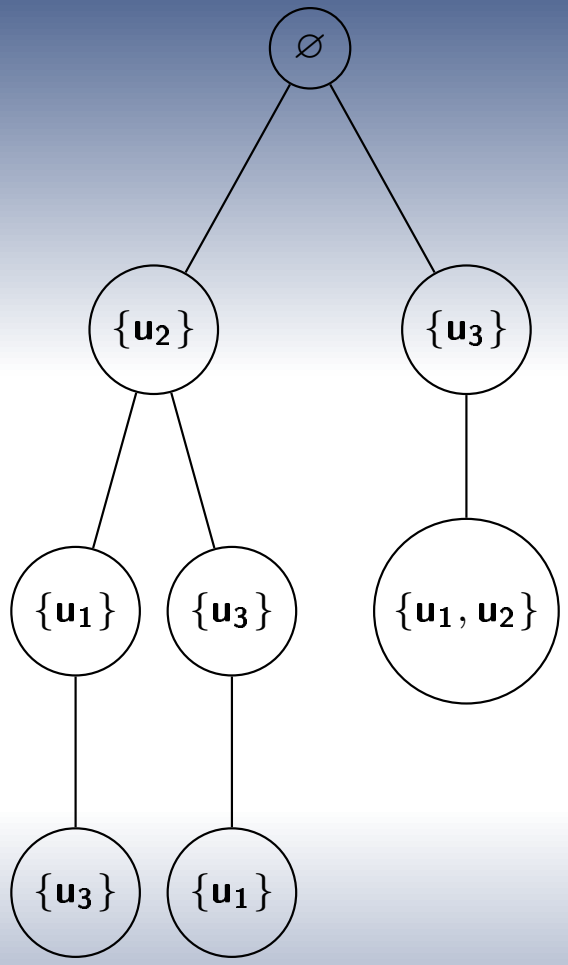

$v(\mathbf{u_i})$
i

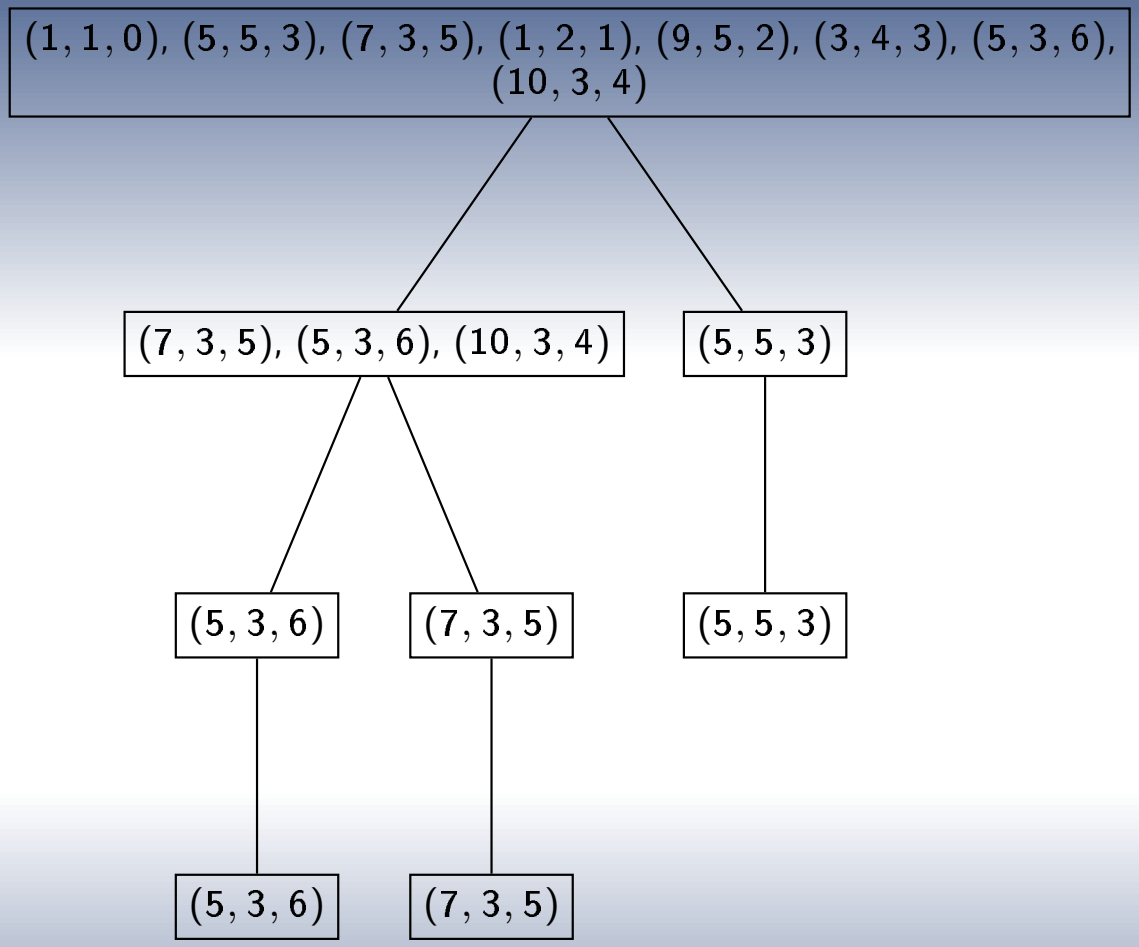

$v(\mathbf{u}_i)$
i

Example

Example

Let $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ be a constraint network and let $(\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3}) \in \mathcal{X}^3$ be an objective vector. We suppose that the set of solutions of the constraint network leads to the following set of possible values for the objective vector: $(1, 1, 0)$, $(5, 5, 3)$, $(7, 3, 5)$, $(1, 2, 1)$, $(9, 5, 2)$, $(3, 4, 3)$, $(5, 3, 6)$ and $(10, 3, 4)$.


∅
{u2}
{u3}
{u1}
{u3}
{u1, u2}
{u3}
{u1}


(1, 1, 0), (5, 5, 3), (7, 3, 5), (1, 2, 1), (9, 5, 2), (3, 4, 3), (5, 3, 6), (10, 3, 4)
(7, 3, 5), (5, 3, 6), (10, 3, 4)
(5, 5, 3)
(5, 3, 6)
(7, 3, 5)
(5, 5, 3)
(5, 3, 6)
(7, 3, 5)

Algorithm 5

Replacing objective variables

The previous algorithm...

In the worst case, we can have to explore all the possible subsets of $\overrightarrow{u}$
$\rightsquigarrow$ exponential number of resolutions.

Algorithm thus to be used only if we are sure that the cardinality-minimal saturated subsets are of **little size**.

The previous algorithm...

In the worst case, we can have to explore all the possible subsets of $\overrightarrow{u}$
$\rightsquigarrow$ exponential number of resolutions.

Algorithm thus to be used only if we are sure that the cardinality-minimal saturated subsets are of **little size**.

In some case, it is guaranteed to be a unique saturated subset of variables of size 1 at each step (dominating criterion).
Example : linear problems with convex set of alternatives.

Variable replacements

When this is not the case...

Proposition

Given a leximin-CSP $(\mathscr{X}, \mathscr{D}, \mathscr{C}, \overrightarrow{\mathbf{o}})$, the set of leximin-optimal solutions does not change when two objective variables $\mathbf{o_i}$ and $\mathbf{o_j}$ are replaced by $max(\mathbf{o_i}, \mathbf{o_j})$ and $min(\mathbf{o_i}, \mathbf{o_j})$.

Variable replacements

When this is not the case...

Proposition

Given a leximin-CSP $(\mathscr{X}, \mathscr{D}, \mathscr{C}, \overrightarrow{\mathbf{o}})$, the set of leximin-optimal solutions does not change when two objective variables $\mathbf{o_i}$ and $\mathbf{o_j}$ are replaced by $max(\mathbf{o_i}, \mathbf{o_j})$ and $min(\mathbf{o_i}, \mathbf{o_j})$.

$\rightsquigarrow$ Idea of the algorithm : at each step we apply the previous rule recursively until $min_i(\mathbf{o_i})$ appears explicitely [Maschler et al., 1992].

Maschler, M., Potters, J. A. M., and Tijs, S. H. (1992).
The general nucleolus and the reduced game property.
International Journal of Game Theory, 21:85–106.

u_1	u_2	u_3
1	1	0
5	5	3
7	3	5
1	2	1
9	5	2
3	4	3
5	3	6
10	3	4

u_1	u_2	u_3	$u_1^{(1)}$	$u_2^{(1)}$	$u_3^{(1)}$
1	1	0	1	1	0
5	5	3	5	5	3
7	3	5	7	5	3
1	2	1	2	1	1
9	5	2	9	5	2
3	4	3	4	3	3
5	3	6	5	6	3
10	3	4	10	4	3

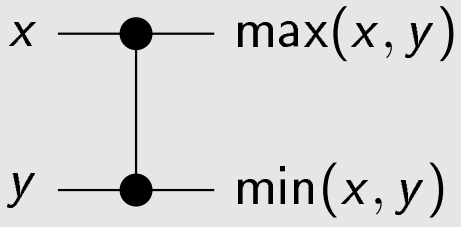
u_1	u_2	u_3	$u_1^{(1)}$	$u_2^{(1)}$	$u_3^{(1)}$	$u_1^{(2)}$	$u_2^{(2)}$
1	1	0	1	1	0		
5	5	3	5	5	3	5	**5**
7	3	5	7	5	3	7	**5**
1	2	1	2	1	1		
9	5	2	9	5	2		
3	4	3	4	3	3	4	3
5	3	6	5	6	3	6	**5**
10	3	4	10	4	3	10	4

u_1	u_2	u_3	$u_1^{(1)}$	$u_2^{(1)}$	$u_3^{(1)}$	$u_1^{(2)}$	$u_2^{(2)}$	$u_1^{(3)}$
1	1	0	1	1	0			
5	5	3	5	5	3	5	5	5
7	3	5	7	5	3	7	5	**7**
1	2	1	2	1	1			
9	5	2	9	5	2			
3	4	3	4	3	3	4	3	
5	3	6	5	6	3	6	5	6
10	3	4	10	4	3	10	4	

Sorting by comparison networks

Interestingly, this is closely related to well-known sorting algorithms based on **comparison networks** [Cormen et al., 2001, page 704].

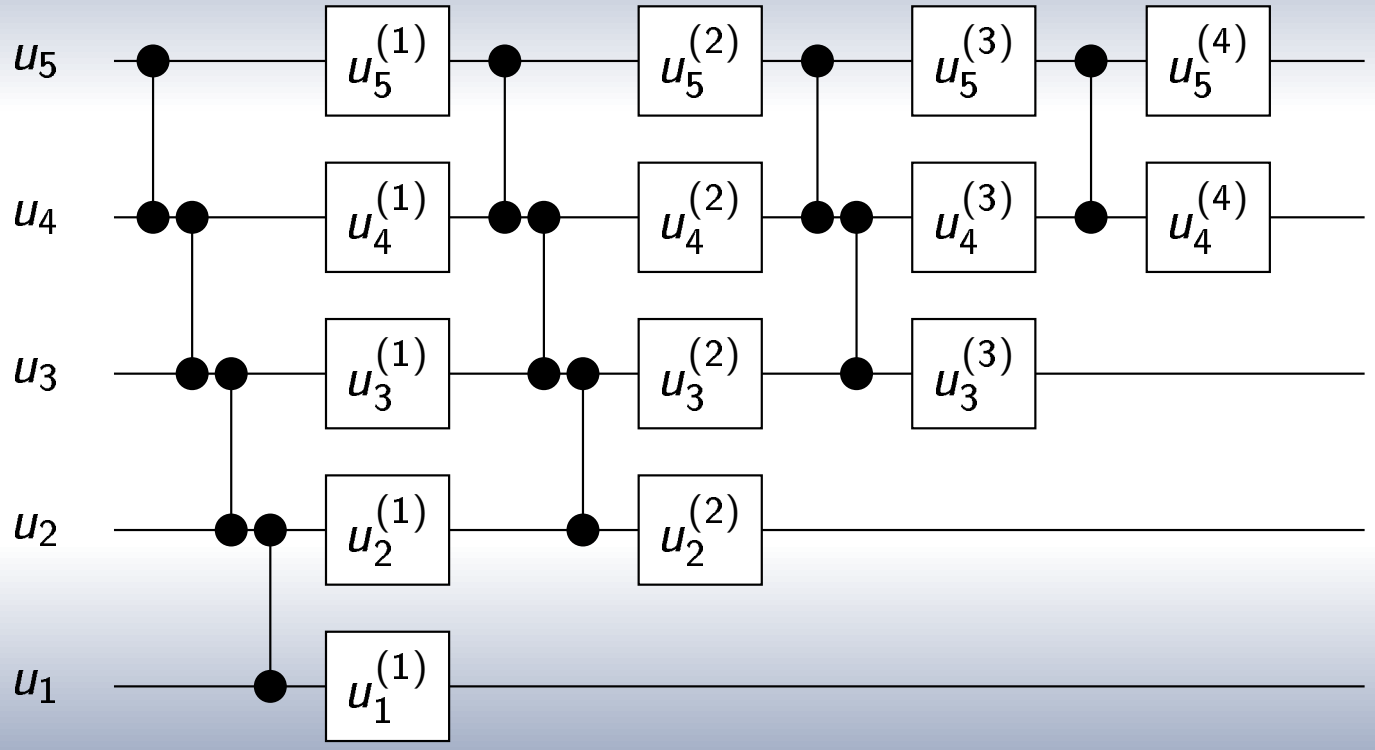
Comparator



The idea is to implement sorting algorithms by only making sequential use of comparators.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001).
Introduction to Algorithms, Second Edition.
MIT Press.

The algorithm expressed using a comparison network



Solving multiagent resource allocation problems

Decentralized allocation

- So far we have seen how to allocate the resource with a **centralized algorithm**.
- Orthogonal approach: no central entity, but a **decentralized negotiation** between the agents.

We are now going to quickly analyse a specific model of distributed negotiation (for concrete allocation protocols, see [Endriss et al., 2006]). The main question concerns the relationship between

- **the local view**: *what deals will agents make in response to their individual preferences?*
- **the global view**: *how will the overall allocation of resources evolve in terms of social welfare?*

Endriss, U., Maudet, N., Sadri, F., and Toni, F. (2006).
Negotiating socially optimal allocations of resources.
Journal of Artificial Intelligence Research, 25:315–348.

An Abstract Negotiation Framework

- Finite set of agents $\mathscr{N}$ and finite set of indivisible resources $\mathscr{O}$.
- An allocation $\overrightarrow{\pi}$ is a vector of **non overlapping bundles** (preemption constraint only).
- Every agent $i \in \mathscr{N}$ has got a utility function $u_i : 2^{\mathscr{O}} \to \mathbb{R}$ (no matter how it is represented).
- Agents may engage in **negotiation** to exchange resources in order to benefit either themselves or society as a whole.
- A **deal** $\delta = (\overrightarrow{\pi}, \overrightarrow{\pi}')$ is a pair of allocations (before/after).
- A deal may come with a number of **side payments** to compensate some of the agents for a loss in utility. A payment function is a function $p : \mathscr{N} \to \mathbb{R}$ with $\sum_{i \in \mathscr{N}} p(i) = 0$.
  Example: $p(i) = 5$ and $p(j) = -5$ means that agent i pays €5, while agent j receives €5.

Individual vs global perspective

The Individual Perspective:
A rational agent (who does not plan ahead) will only accept deals that improve its individual welfare:

Individually rational deal

A deal $\delta = (\overrightarrow{\pi}, \overrightarrow{\pi}')$ is called individually rational iff there exists a payment function p such that $u_i(\overrightarrow{\pi}') - u_i(\overrightarrow{\pi}) > p(i)$ for all $i \in \mathcal{N}$, except possibly $p(i) = 0$ for agents i with $\pi_i = \pi'_i$.

That is, an agent will only accept a deal iff it results in a gain in utility (or money) that strictly outweighs a possible loss in money (or utility).

The Global/Social Perspective:
The system designer is interested in maximizing a given criterion (suppose it is the utilitarian social welfare).

Example

Let $\mathscr{N} = \{ann, bob\}$ and $\mathscr{O} = \{chair, table\}$ and suppose our agents use the following utility functions:

$$\begin{array}{rcl} u_{ann}(\varnothing) & = & 0 \\ u_{ann}(chair) & = & 2 \\ u_{ann}(table) & = & 3 \\ u_{ann}(chair, table) & = & 8 \end{array} \qquad \begin{array}{rcl} u_{bob}(\varnothing) & = & 0 \\ u_{bob}(chair) & = & 3 \\ u_{bob}(table) & = & 3 \\ u_{bob}(chair, table) & = & 7 \end{array}$$

Furthermore, suppose the initial allocation of resources is $\overrightarrow{\pi}^0$ with $\pi_0^{ann} = \{chair, table\}$ and $\pi_{bob}^0 = \varnothing$.

Example

Let $\mathscr{N} = \{ann, bob\}$ and $\mathscr{O} = \{chair, table\}$ and suppose our agents use the following utility functions:

$$\begin{array}{rcl} u_{ann}(\varnothing) & = & 0 \\ u_{ann}(chair) & = & 2 \\ u_{ann}(table) & = & 3 \\ u_{ann}(chair, table) & = & 8 \end{array} \qquad \begin{array}{rcl} u_{bob}(\varnothing) & = & 0 \\ u_{bob}(chair) & = & 3 \\ u_{bob}(table) & = & 3 \\ u_{bob}(chair, table) & = & 7 \end{array}$$

Furthermore, suppose the initial allocation of resources is $\overrightarrow{\pi}^0$ with $\pi_0^{ann} = \{chair, table\}$ and $\pi_{bob}^0 = \varnothing$.

Social welfare for allocation $\overrightarrow{\pi}^0$ is 7, but it could be 8. By moving only a single resource from agent ann to agent bob, the former would lose more than the latter would gain (not individually rational).
The only possible deal would be to move the whole set $\{chair, table\}$.

Local and global perspectives

It turns out that individually rational deals are exactly those deals that increase social welfare:

Lemma [Rationality and social welfare]

A deal $\delta = (\overrightarrow{\pi}, \overrightarrow{\pi}')$ with side payments is individually rational iff $u_c(\overrightarrow{\pi}) < u_c(\overrightarrow{\pi}')$.

Local and global perspectives

It turns out that individually rational deals are exactly those deals that increase social welfare:

Lemma [Rationality and social welfare]

A deal $\delta = (\overrightarrow{\pi}, \overrightarrow{\pi}')$ with side payments is individually rational iff $u_c(\overrightarrow{\pi}) < u_c(\overrightarrow{\pi}')$.

Proof

- $\Rightarrow$ Rationality means that overall utility gains outweigh overall payments (which are $= 0$).
- $\Leftarrow$ The social surplus can be divided amongst all deal participants by using the following payment function:

$$p(i) = u_i(\overrightarrow{\pi}') - u_i(\overrightarrow{\pi}) - \underbrace{\frac{u_c(\overrightarrow{\pi}') - u_c(\overrightarrow{\pi})}{|\mathcal{N}|}}_{>0}$$

Convergence

It is now easy to prove the following convergence result (originally stated by Sandholm in the context of distributed task allocation):

Theorem [Sandholm, 1998]

Any sequence of individually rational deals will eventually result in an allocation with maximal utilitarian social welfare.

Convergence

It is now easy to prove the following convergence result (originally stated by Sandholm in the context of distributed task allocation):

Theorem [Sandholm, 1998]

Any sequence of individually rational deals will eventually result in an allocation with maximal utilitarian social welfare.

Proof: Termination follows from our lemma and the fact that the number of allocations is finite. So let $\vec{\pi}$ be the terminal allocation. Assume $\vec{\pi}$ is not optimal, *i.e.* there exists an allocation $\vec{\pi}'$ with $u_c(\vec{\pi}) < u_c(\vec{\pi}')$. Then, by our lemma, $\delta = (\vec{\pi}, \vec{\pi}')$ is individually rational $\Rightarrow$ contradiction.

Agents can act locally and need not be aware of the global picture (convergence towards a global optimum is guaranteed by the theorem).

Sandholm, T. W. (1998).
Contract types for satisficing task allocation: I. theoretical results.
In Sen, S., editor, *Proceedings of the AAAI Spring Symposium: Satisficing Models*, pages 68–75, Menlo Park, California. AAAI Press.

1-deals

The later convergence result involves **any kind of deals** (*e.g* bundles of arbitrary sizes).

Do we have similar results if we restrict the set of possible deals ?

1-deals

The later convergence result involves **any kind of deals** (*e.g* bundles of arbitrary sizes).

Do we have similar results if we restrict the set of possible deals ?

Convergence of 1-deal negotiation [Chevaleyre et al., 2005]

Any sequence of individually rational 1-deals will eventually result in an allocation with maximal utilitarian social welfare, if the agents have **additive** utilities.

(Moreover, the class of additive utilities is maximal (wrt inclusion) regarding this convergence property).

Chevaleyre, Y., Endriss, U., and Maudet, N. (2005).
On maximal classes of utility functions for efficient one-to-one negociation.
In Kaelbling, L. P. and Saffiotti, A., editors, *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, Edinburgh, Scotland. Professional Book Center.

Solving multiagent resource allocation problems

Solving resource allocation problems

Solving a multiagent resource allocation problems... Several approaches:

- **Centralized allocation**:
  - General purpose AI algorithms (Branch & Bound, A*...) with specific heuristics (see Combinatorial Auctions)
  - General modeling and solving frameworks: ILP, Constraint programming.
- **Decentralized negotiation**:
  - Convergence properties: individual vs global point of view.
  - Protocols / Implantations (not addressed here).

Part 5

Other issues, other applications

Other issues, other applications

What we have seen...

In this talk, we have seen...

- The basic ingredients of a resource allocation problem: **resource**, **agents** expressing **preferences** on the resource, **constraints** restricting the set of possible allocation
- Why representing the agents preferences is not straightforward, because there can be **dependencies** between objects, and representing them explicitly needs exponential time and space $\rightsquigarrow$ need for a **compact representation language**
- Using a compact representation language can render the classical problems more or less complex.
- How to **aggregate** the agents preferences into a **collective preference** using **social welfare orderings** that encode efficiency and fairness.
- How to **solve** multiagent resource allocation problems using AI techniques or frameworks.

Other issues, other applications

Computer networks

- Resource allocation has been extensively studied in **computer networks**.
- The resource to share is (usually) **the bandwidth**.
- The agents are the network users (or the applications themselves – FTP, real-time applications, ...).
- Equity is crucial and can intervene at different levels.
- Usually this resource allocation problem is tackled as a **continuous** and time-dependant problem.
- See [Denda et al., 2000] for a general study of equity in computer networks.

Denda, R., Banchs, A., and Effelsberg, W. (2000).
The fairness challenge in computer networks.
In *QofIS '00: Proceedings of the First COST 263 International Workshop on Quality of Future Internet Services*, pages 208–220, London, UK. Springer-Verlag.

Air traffic management

The huge increase in air traffic in the last decades needs to be addressed by a better sharing of airspace sectors and take off and landing airport slots.

- The resource to share is the set of **air sectors**, and of **take off and landing slots**.
- The agents can be **airlines** or **aircrafts** themselves.
- Constraints: several security norms / flight times / roads.
- Current solution to congestion: delay the flights on ground (unfair, leading to unacceptable delays)
- Solutions studied [Deschinkel, 2001]:
  - optimisation with global utilitarian criteria (minimize the costs)
  - flexible price setting.

Deschinkel, K. (2001).
Régulation du Trafic aérien par Optimisation Dynamique des Prix du Réseau.
PhD thesis, École Nationale Supérieure de l'Aéronautique et de l'Espace.

Crew scheduling problems

- Many applications concern crew scheduling: airlines, fire departments, nurse rostering....
- The agents here are the different members of the crew (stewards and pilots, nurses, firemen...).
- The resource to share is working slots, night shifts...
- The allocation has to satisfy some constraints (physical and legal).
- The allocation has to be **fair** (it would be unacceptable that a given nurse works every week-end, while another one never works during week-ends).
- In usual works, the fairness requirement is not modeled as is, but rather ensured by specific heuristics.

Allocating subjects

- A set of subjects of practical works must be allocated to students.
- 2 subjects per student (first semester, second semester).
- The students give their preferences on the subjects.
- Constraints: the same subject cannot be allocated twice the same semester. The students must have two subjects from different areas (e.g mathematics, physics).
- This (simple) problem arises in a lot of schools, universities, and so on.
- It is not so easy to find an optimal egalitarian allocation (often, the problem is not very well formalized).
- Students often try to manipulate...

Waste Water Treatment Problem

- A Waste Water Treatment Plant collects polluted waste water from different cities, industries... and is in charge of treating it.
- The plant has a **limited capacity**.
- This problem can be solved as a multiagent resource allocation problem, where the resource is the waste water discharge rights.
- More precisely, this problem has been solved using a sequential combinatorial auctions approach in [Murillo Espinar, 2010].
- Fairness mechanisms ensure that the agents will not leave the market and disobey.

Murillo Espinar, J. (2010).
Egalitarian Behaviour in Multiunit Combinatorial Auctions.
PhD thesis, Universitat de Girona.

Other issues, other applications

Other important issues...

Other important topics we have not dealt with...

- Unequal exogenous rights
- Repeated / sequential allocations
- Dealing with uncertainty
- Mechanism design / incentive compatibility and manipulation
- Continuous resource
- ...

Unequal exogenous rights

- In this talk, we assumed that the agents had **equal exogenous rights**.
- There are practical examples where it is not the case (*e.g* Earth Observing Satellites)
- **Question:** *How to extend the classical properties and SWO to deal with exogenous rights ?*
- **Example**: duplication of agents (works well with the utilitarian SWO, but not so well with the egalitarian one).
- Some insights in [Bouveret, 2007].

Bouveret, S. (2007).
Allocation et partage équitables de ressources indivisibles : modélisation, complexité et algorithmique.
PhD thesis, Université de Toulouse.

Repeated allocations

- Some real-world resource allocation problems are in fact **repeated resource allocation problems** (*e.g* Earth Observing Satellites...).
- Can we exploit this to ensure **fairness in average** ?
- Link with **unequal exogenous rights** ?
- This idea has already been studied in [Lemaître et al., 1999].

Lemaître, M., Verfaillie, G., and Bataille, N. (1999).
Exploiting a common property resource under a fairness constraint: a case study.
In Dean, T., editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 206–211, Stockholm, Sweden. Morgan Kaufmann.

Uncertainty

- In the classical Resource Allocation setting, resource (or bidder) availability is taken for sure.
- However, this is not the case in several problems (*e.g* Earth Observing Satellites), where:
  - an object can be **broken**, or **damaged** with a given probability
  - an agent can **withdraw** her bid at the last second.
- Dealing with uncertainty in fair resource allocation problems is classical in economics (well known timing effect), but not so much in computer science.

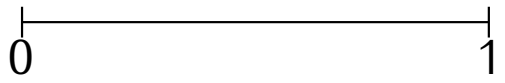
Mechanism Design

An important topic that we have not covered is the game-theoretical analysis of MARA problems, in particular mechanism design.

- While game theory analyses the strategic behaviour of rational agents in a given game, mechanism design uses these insights to design games inducing certain strategies (and hence outcomes).
- A central result is the incentive-compatibility of reporting your true valuation in the Vickrey-Clarke-Groves mechanism (which is a generalisation of second-price auctions).

Continuous resources

- Cake-cutting as a metaphor for the fair division of a single divisible (and heterogeneous) good between n agents (called players).
- Studied seriously since the 1940s (Banach, Knaster, Steinhaus). Simple model, yet still many open problems.
- The cake is represented by the interval $[0, 1]$.



- Each player has an **additive** utility function over intervals of $[0, 1]$ to $\mathbb{R}$.
- The problem is to find a **good procedure** (**e.g** that guarantees fair-share, envy-freeness, etc.).

Two examples

The classical approach for dividing a cake between two players:

I cut you choose

One player cuts the cake in two pieces (which she considers to be of equal value), and the other one chooses one of the pieces (the piece she prefers).

Two examples

The classical approach for dividing a cake between two players:

I cut you choose

One player cuts the cake in two pieces (which she considers to be of equal value), and the other one chooses one of the pieces (the piece she prefers).

Another procedure, for n players.

Dubins-Spanier procedure

1. A referee moves a knife slowly across the cake, from left to right. Any player may shout "stop" at any time. Whoever does so receives the piece to the left of the knife.
2. When a piece has been cut off, we continue with the remaining $n-1$ players, until just one player is left (who takes the rest).

This procedure is not envy-free. The last chooser is best off (she is the only one who can get more than $1/n$).

Other issues, other applications

Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H., and Poole, D. (2004).
CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements.
Journal of Artificial Intelligence Research, 21:135–191.

Bouveret, S. (2007).
Allocation et partage équitables de ressources indivisibles : modélisation, complexité et algorithmique.
PhD thesis, Université de Toulouse.

Brafman, R. I. and Domshlak, C. (2002).
Introducing variable importance tradeoffs into CP-nets.
In Darwiche, A. and Friedman, N., editors, *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI-02)*, pages 69–76, Edmonton, Alberta, Canada. Morgan Kaufmann.

Brafman, R. I., Domshlak, C., and Shimony, S. E. (2006).
On graphical modeling of preference and importance.
J. Artif. Intell. Res. (JAIR), 25:389–424.

Brams, S. J., Edelman, P. H., and Fishburn, P. C. (2004).
Fair division of indivisible items.
Theory and Decision, 5(2):147–180.

Brams, S. J. and King, D. (2005).
Efficient fair division—help the worst off or avoid envy?
Rationality and Society, 17(4):387–421.

Chen, L. and Pu, P. (2004).
Survey of preference elicitation methods.
Technical report, École Polytechnique Fédérale de Lausanne.

Chevaleyre, Y., Endriss, U., and Lang, J. (2006).
Expressive power of weighted propositional formulas for cardinal preference modelling.
In *Proc. of KR-06*.

Chevaleyre, Y., Endriss, U., and Maudet, N. (2005).
On maximal classes of utility functions for efficient one-to-one negociation.
In Kaelbling, L. P. and Saffiotti, A., editors, *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, Edinburgh, Scotland. Professional Book Center.

Coste-Marquis, S., Lang, J., Liberatore, P., and Marquis, P. (2004).
Expressive power and succinctness of propositional languages for preference representation.
In *Proc. of KR-04*.

Cramton, P., Shoham, Y., and Steinberg, R., editors (2006).
Combinatorial Auctions.
MIT Press.

Demko, S. and Hill, T. P. (1998).
Equitable distribution of indivisible items.
Mathematical Social Sciences, 16:145–158.

Denda, R., Banchs, A., and Effelsberg, W. (2000).
The fairness challenge in computer networks.
In *QofIS '00: Proceedings of the First COST 263 International Workshop on Quality of Future Internet Services*, pages 208–220, London, UK. Springer-Verlag.

Deschinkel, K. (2001).
Régulation du Trafic aérien par Optimisation Dynamique des Prix du Réseau.
PhD thesis, École Nationale Supérieure de l'Aéronautique et de l'Espace.

Endriss, U., Maudet, N., Sadri, F., and Toni, F. (2006).
Negotiating socially optimal allocations of resources.
Journal of Artificial Intelligence Research, 25:315–348.

Faltings, B. (2005).
A budget-balanced, incentive-compatible scheme for social choice.
In Faratin, P. and Rodriguez-Aguilar, J. A., editors, *Agent-Mediated Electronic Commerce VI*, volume 3435 of *LNAI*, pages 30–43. Springer.

Fodor, J. and Roubens, M. (1994).
Fuzzy Preference Modelling and Multicriteria Decision Support.
Kluwer Academic Publishers.

Foley, D. K. (1967).
Resource allocation and the public sector.
Yale Economic Essays, 7(1):45–98.

Frisch, A. M., Hnich, B., Kiziltan, Z., Miguel, I., and Walsh, T. (2003).

Multiset ordering constraints.
In Gottlob, G. and Walsh, T., editors, *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, Acapulco, Mexico. Morgan Kaufmann.

Grabisch, M. (1997).
k-order additive discrete fuzzy measure and their representation.
Fuzzy Sets and Systems, 92:167–189.

Herreiner, D. K. and Puppe, C. (2002).
A simple procedure for finding equitable allocations of indivisible goods.
Social Choice and Welfare, 19:415–430.

Keeney, R. L. and Raiffa, H. (1976).
Decisions with Multiple Objectives: Preferences and Value Tradeoffs.
John Wiley and Sons.

Lang, J. (2004).
Logical preference representation and combinatorial vote.
Annals of Mathematics and Artificial Intelligence, 42(1):37–71.

Lemaître, M., Verfaillie, G., and Bataille, N. (1999).
Exploiting a common property resource under a fairness constraint: a case study.
In Dean, T., editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 206–211, Stockholm, Sweden. Morgan Kaufmann.

Mehlhorn, K. and Thiel, S. (2000).
Faster algorithms for bound-consistency of the sortedness and the alldifferent constraint.
In Dechter, R., editor, *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming (CP-00)*, pages 306–319, Singapore. Springer.

Montanari, U. (1974).
Network of constraints: Fundamental properties and applications to picture processing.
Inf. Sci., 7:95–132.

Moulin, H. (2003).
Fair Division and Collective Welfare.
MIT Press.

Murillo Espinar, J. (2010).
Egalitarian Behaviour in Multiunit Combinatorial Auctions.
PhD thesis, Universitat de Girona.

Perny, P. and Roy, B. (1992).
The use of fuzzy outranking relations in preference modelling.
Fuzzy sets and systems, 49:33–53.

Pirlot, M. and Vincke, P. (1997).
Semi Orders.
Kluwer Academic Publishers, Dordrecht.

Rawls, J. (1971).
A Theory of Justice.
Harvard University Press, Cambridge, Mass.
Traduction française disponible aux éditions du Seuil.

Roberts, K. W. S. (1980).
Interpersonal comparability and social choice theory.
Review of Economic Studies, 47:421–439.

Sandholm, T. W. (1998).
Contract types for satisficing task allocation: I. theoretical results.
In Sen, S., editor, *Proceedings of the AAAI Spring Symposium: Satisficing Models*, pages 68–75, Menlo Park, California. AAAI Press.

Sandholm, T. W. (2006).
Optimal winner determination algorithms.
In Cramton, P., Shoham, Y., and Steinberg, R., editors, *Combinatorial auctions*. MIT Press.

Tinbergen, J. (1953).
Redeljke Inkomensverdeling.
N. V. DeGulden Pers., Haarlem.

Uckelman, J. (2008).
More Than the Sum of Its Parts. Compact Preference Representation of Combinatorial Domains.
PhD thesis, Universiteit van Amsterdam.

van Hentenryck, P., Simonis, H., and Dincbas, M. (1992).

Constraint satisfaction using constraint logic programming.
Artificial Intelligence, 58(1-3):113–159.

Vincke, P. (1978).
Quasi-ordres généralisés et représentation numérique.
Mathématiques et sciences humaines, 62:35–60.

Wilson, N. (2004).
Extending CP-nets with stronger conditional preference statements.
In *Proceedings of AAAI'04*.

Young, H. P. (1994).
Equity in Theory and Practice.
Princeton University Press.