Incomparability and uncertainty in preference aggregation

Francesca Rossi
University of Padova, Italy

COMSOC 2006, Amsterdam, Dec. 2006

Joint work with ...

- K. R. Apt, CWI, Amsterdam
- Jerome Lang, IRIT-CNRS, France
- Maria Silvia Pini, Univ. Padova, Italy
- K. Brent Venable, Univ. Padova, Italy
- Toby Walsh, NICTA and Univ. of New South Wales, Australia

Outline

- Representing constraints and preferences:
  - Soft constraints (quantitative preferences)
  - CP nets (qualitative conditional preferences)
- Aggregating partially ordered preferences
  - Fairness: possibility and impossibility results
  - Non-manipulability
- Adding uncertainty to incomparability
  - Complexity of finding the winners
  - Sequential majority voting
- Back to CP nets and soft constraints
  - to find optimals in preference aggregation

How to represent preferences compactly?

- Preferences define an ordering over a set of objects
- The set can be esponentially large w.r.t. some given input size
  - Instantiation of n variables over their domains
  - Configurations of n objects
- We need ways to specify the ordering compactly

AI formalisms for modelling preferences compactly

Many, but I will focus on two of them:

- Soft Constraints
  - Quantitative preferences
  - Preferences + constraints
- CP-nets (Conditional Preference Networks)
  - Qualitative conditional preferences
  - No constraints

Soft constraints

Soft Constraints: the c-semiring framework

- Variables $\{X_1,\ldots,X_n\}=X$
- Domains $\{D(X_1),\ldots,D(X_n)\}=D$
- Soft constraints
  - each constraint involves some of the variables
  - a preference is associated with each assignment of the variables
- Set of preferences A
  - Totally or partially ordered (indiced by +)
  - Combination operator (x)
  - Top and bottom element (**1**, **0**)
  - Formally defined by a c-semiring $<A,+,x,\mathbf{0},\mathbf{1}>$

[Bistarelli, Montanari, Rossi, IJCAI 1995]

Instances of soft constraints

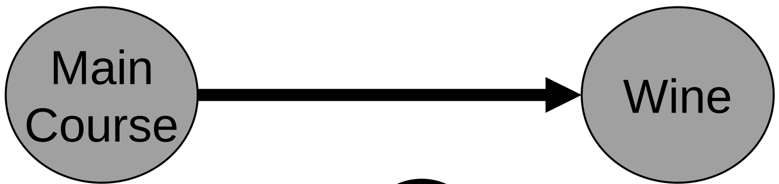
- Each instance is characterized by a c-semiring <A, +, x, 0, 1>
- Classical constraints: <{0,1},logical or,logical and,0,1>
- Fuzzy constraints: <[0,1],max,min,0,1>
- Lexicographic CSPs: <$[0,1]^k$,lex-max,min,0^k,1^k>
- Weighted constraints (N):<N∪+∞,+, min,+∞,0>
- Weighted constraints (R):<R∪+∞,+, min,+∞,0>
- Max CSP: weight =1 when constraint is not satisfied and 0 is satisfied
- Probabilistic constraints: <[0,1],x,max,0,1>
- Valued CSPs: any semiring with a total order
- Multi-criteria problems: Cartesian product of semirings

Solutions

- Global evaluation: preference associated to a complete assignment
- How to obtain a global evaluation?
  - By combining (via x) the preferences of the partial assignments given by the constraints

Fuzzy-SCSP example

{Fish, Meat} {White, red}



(Fish, white) → **1** (Meat, white) → **0.3**

(Fish, red) → 0.8 (Meat, red) → 0.7

{12 am, 1 pm} {2 pm, 3 pm}

(12 am, 2 pm) → **1** (1 pm, 2 pm) → **0**

(12 am, 3 pm) → 1 (1 pm , 3 pm) → 1

Fuzzy semiring

$$S \;=\; \langle A,\; +,\; x,\; \mathbf{0}, \mathbf{1} \rangle$$

$$\downarrow$$

$$S_{FCSP} = \langle [0,1], max, min, 0, 1 \rangle$$

Solution *S*	
Lunch=	1 pm
Main course =	meat
Wine=	white
Swim =	2 pm

pref(S)=min(0.3,0)=0

Solution *S'*	
Lunch=	12 pm
Main course =	fish
Wine=	white
Swim =	2 pm

pref(S)=min(1,1)=1

Solution ordering

- A soft CSP induces an ordering over the solutions, from the ordering of the preference set
- Totally ordered → total order over solutions (possibly with ties)
- Partially ordered → total or partial order over solutions (possibly with ties)
- Any ordering can be obtained

Typical questions

- Find an optimal solution
  - Difficult: NP-hard

  (ex.: branch and bound + adapted constraint propagation)
- Is t an optimal solution?
  - Difficult: NP-hard

  (we first have to find the optimal preference level)
- Is t better than t'?
  - Easy: Linear in the number of constraints

  (if + and x are easy to compute: compute (x) the two pref. levels and compare (+) them)

CP nets

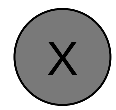
CP nets

- Conditional preference statements
  - If it is fish I prefer white wine to red
  - syntax: fish: white wine > red wine
- Ceteris paribus interpretation
  - all else being equal
  - {fish, white wine, ice cream} > (preferred to) {fish, red wine, ice cream}
  - {fish, white wine, ice cream} ? {fish, red wine, fruit}

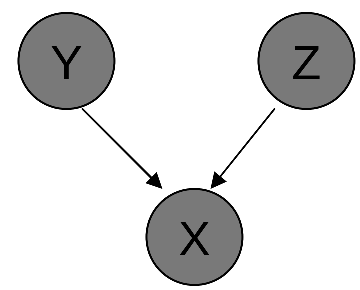
[Boutilier, Brafman, Hoos, Poole UAI99]
[Boutilier, Bacchus, Brafman UA01]
[Domshlak, Brafman KR02]

CP nets

- Variables $\{X_1, \ldots , X_n\}$ with domains
- For each variable, a total order over its values
- Indipendent variable:
  - X=v1 > X=v2 > ... > X=vk



- Conditioned variable: a total order for each combination of values of some other variables (conditional preference table)
  - Y=a, Z=b: X=v1 > X=v2 > ... > X=vk
  - X depends on Y and Z (parents of X)



- Graphically: directed graph over $X_1, \ldots , X_n$
  - Possibly cyclic

CP-net semantics

- Worsening flip: changing the value of an attribute in a way that is less preferred in some statement
- An outcome O_1 is preferred to O_2 iff there is a sequence of worsening flips from O_1 to O_2
- Optimal outcome: if no other outcome is preferred

Preorder over solutions

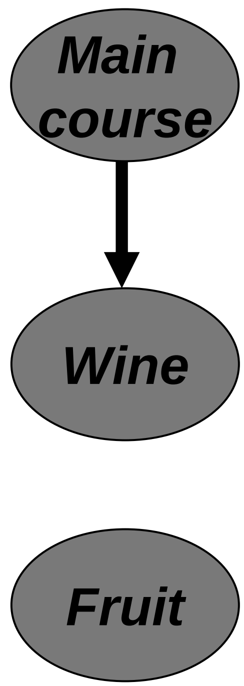
- A CP net induces an ordering over the solutions (directly)
- In general, a preorder
- Some solutions can be in a cycle: for each of them, there is another one which is better
- Acyclic CP net: one optimal solution
- Not all orderings can be obtained with CP nets
  - Outcomes which are one flip apart must be ordered

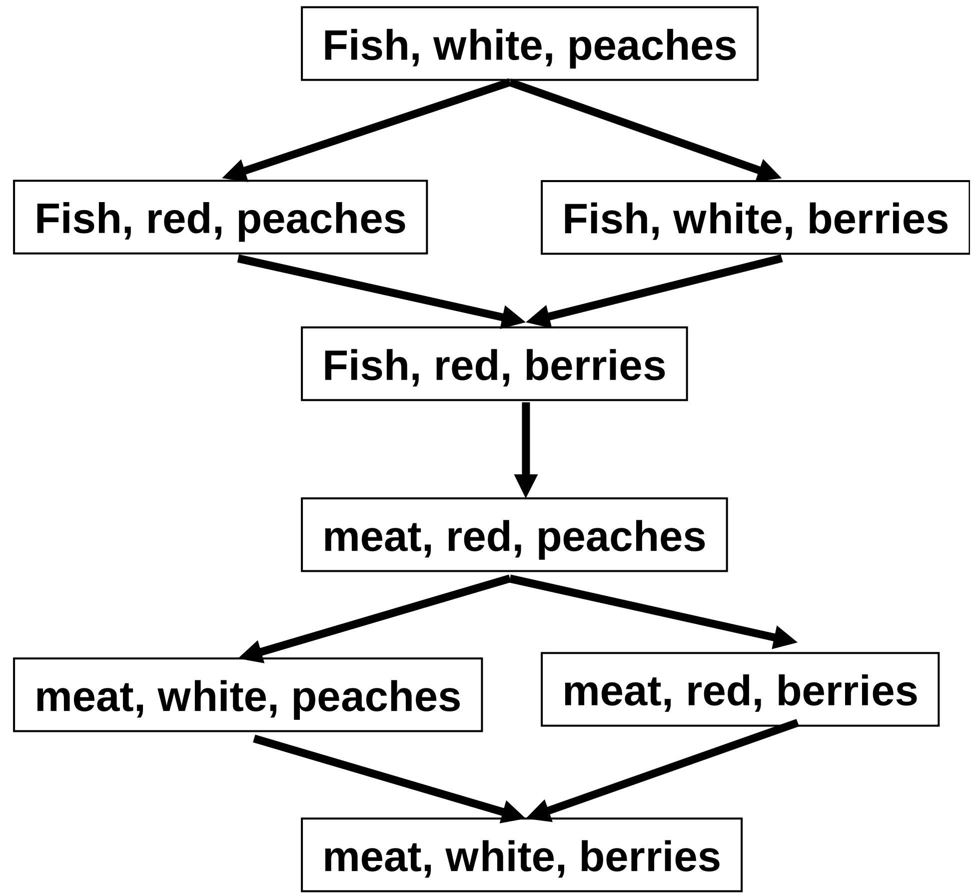
Example: solution ordering

fish>meat

Main course	*Wine*
fish	white > red
meat	red > white

peaches > strawberries





Typical questions in CP nets

- Find an optimal outcome
  - In general, difficult (as solving a CSP)
  - Acyclic networks always have one
    - Sweep forward in linear time
    - Example: a>-a, -b>b, ab:-c>c, -a-b:c>-c

      a then -b then c
- Does O1 dominate O2?
  - Difficult even for acyclic CP nets
  - Not even known to be in NP
- Is O optimal?
  - Easy (test O against a CSP)

Summary of preference representation formalisms

- CP nets
  - pros: conditional, qualitative
  - cons: comparing outcomes
- Soft constraints
  - pros: comparing outcomes, hard constraints
  - cons: quantitative
- Both may produce partially ordered solution sets

Multi-agent setting: aggregating partially ordered preferences

The considered setting

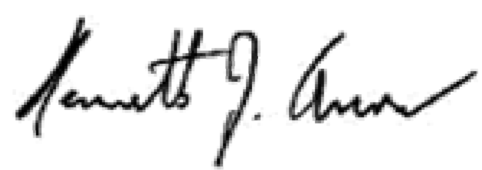
- Several agents (people, software agents, etc.) expressing their preferences over a set of scenarios (solutions, outcomes, etc.)
- We need to aggregate their preferences to obtain a result which satisfies all
- Result can be:
  - A preference ordering over the scenarios (social welfare)
  - A set of scenarios (social choice)
- Preferences (of one agent, or in the result) are expressed via partial orders

Some desired properties

- Unanimity
  - If all agents say A better than B, the result must say the same
- Independence to irrelevant alternatives
  - Final ordering of two outcomes only depends on how agents order these two outcomes
- Non-dictatorship
  - Dictatorial: for any election, the resulting ordering depends on just one agent (he cannot be contradicted)

Arrow's theorem

- Voting system: function from a set of total orders with ties to a total order with ties
- A voting system is fair is it is unanimous, independent to irrelevant alternatives, and non-dictatorial
- Theorem: there is no voting system, with at least 2 voters and 3 outcomes, which is fair
- Proof: if we assume unanimity and independence, then the voting system must be dictatorial



[Arrow, 1951]

Fairness

- Can we fairly combine the agents' preferences?
- Does Arrow's theorem hold also in our context?
- Not directly: voters (and result) may include incomparability
- Arrow's theorem assumes a total order for each agent and for the result

What is a dictator when we use POs?

- Strong dictator: a voter such that his ordering is the result
- Dictator: if he says A better than B, then the result is A better than B
  - But if he says that A and B are incomparable or indifferent, then they can be ordered in the result
  - Same notion as for TOs in Arrow's theorem
- Weak dictator: if he says A better than B, then the result cannot be B better than A
  - But it can be A incomparable/indifferent to B
- At most one strong dictator or dictator, possibly many weak dictators
- Strong dictator ➔ dictator ➔ weak dictator
- Weak fairness ➔ fairness ➔ strong fairness

Strong dictators

- No strong dictator: very weak property
- Example: Lex
  - It is free, monotonic, independent, and does not have any strong dictator
  - The first agent does not dictate indifference, so it is not a strong dictator
  - It is a dictator however
- So, with partial orders it is possible to be strongly fair

Dictators

- It is possible to be fair
- Example: Pareto
  - It is free, monotonic, transitive, independent, and does not have any dictator
  - The only way one agent can force the result is by stating that all outcomes are incomparable
  - All agents are weak dictators however

Weak fairness is not possible

- Theorem: If
  - At least 2 agents and 3 outcomes,
  - Social welfare function unanimous and IIA,
  - Agents express their preferences as POs
  - The resulting ordering is an rPO (unique top **or** unique bottom)

➔ There is at least one weak dictator

(➔ it is impossible to be weakly fair)

[Pini, Rossi, Venable, Walsh, TARK 2005]

Optimals only (social choice)

- Result is a set of winners, not an ordering of the outcomes
- Unanimous: given any profile p,
  - If a in top(p_i) for every i, then a in f(p)
  - If {a} = top(p_i) for every i, then f(p) = {a}
- Monotonic: given two profiles p, p'
  - If a in f(p) and for any b, a improves over b from p to p' in all agents, then a in f(p')
  - If f(p) = S and for all s in S, s improves over any b from p to p' in all agents i, then f(p') = S

Dictators for social choice functions

- Strong dictator: agent i such that, for all profiles p, $f(p) = top(p_i)$
- Dictator: agent i such that, for all profiles p, $f(p) \subseteq top(p_i)$
- Weak dictator: agent i such that, for all profiles p, $f(p) \cap top(p_i) \neq \emptyset$
- Consistent with corresponding notions for social welfare function f', where $f(p) = top(f'(p))$

Impossibility result for weak fairness (extension of Muller-Satterthwaite thm.)

- If
  - At least 2 agents and 3 outcomes,
  - Social choice function with no ties unanimous and monotonic,
  - Agents express their preferences as POs
  - The resulting ordering is a PO
- ➔ There is at least one weak dictator

(➔ it is impossible to be weakly fair)

[Pini, Rossi, Venable, Walsh, TARK 2005]

Strategy proofness

- Agents should not be able to make an outcome win by lowering its position in their preference ordering
- For every agent i, for every two profiles p and p', which differ on p_i only, for every a in f(p)-f(p'), for every b in f(p'),
  - $a \bowtie_{pi} b \Rightarrow a \bowtie_{p'i} b$ or $a <_{p'i} b$
  - $a <_{pi} b \Rightarrow a <_{p'i} b$
- There is at least an element b in f(p') such that
  - $(a >_{pi} b)$ and $(a \bowtie_{p'i} b$ or $a <_{p'i} b)$, or
  - $(a \bowtie_{pi} b)$ and $(a <_{p'i} b)$
- One agent can remove an element (a) from the set of winners only by worsening it with respect to at least one of the new winners (b)

Results on strategy proofness
(extension of Gibbard-Satterthwaite thm.)

- Social choice function from POs to PO
- Strategy proofness ➔ monotonicity
- Onto + monotonicity ➔ unanimity

➔ Strategy proofness + onto ➔ unanimity + monotonicity

➔ Strategy proofness + onto ➔ at least one weak dictator

[Pini, Rossi, Venable, Walsh, AAMAS 2006]

Adding uncertainty to incomparability

Our notion of uncertainty

- Uncertainty: we don't know the relationship between two candidates
  - They could be ordered, tied, or incomparable
  - Complete absense of knowledge (no possibilities, no probabilities, etc.)
- Maybe we will know later
  - On-going preference elicitation
- At any given point in time, four kinds of relation between A and B
  - A above B (A>B) or B above A (B>A)
  - A incomparable to B (A ~ B)
  - A indifferent to B (A=B)
  - Unknown: it could be any of the above

Possible and necessary winners

- Since there are incomplete preferences, we focus on computing possible (PW) and necessary winners (NW)
- Necessary winners
  - outcomes which are maximal in every completion
    - winners no matter how incompleteness is resolved
- Possible winners
  - outcomes which are maximal in at least one of the completions
    - winners in at least one way in which incompleteness is resolved

[Konczac and Lang, 2005]

Computational aspects

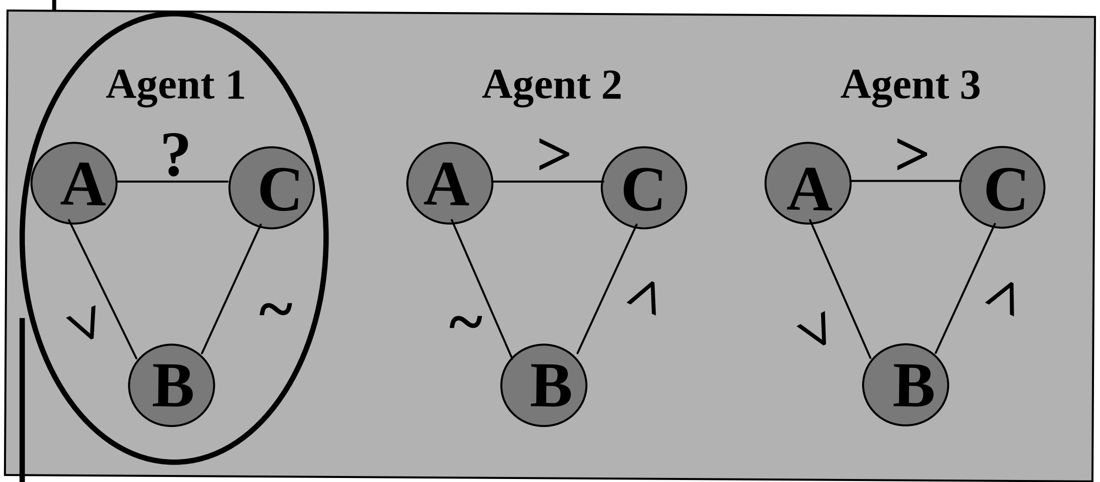
- Possibility and impossibility results still hold
- Without uncertainty, if preference aggregation is easy, computing the winners is easy
- With uncertainty, there is an exponential number of profile completions to consider
  - If preference aggregation is polynomial, is it still easy to compute the winners?

Main results

- Computing PW and NW: *difficult*
- Approximating PW and NW: *difficult*
- Sufficient conditions on preference aggregation such that computing PW and NW is *easy*
- How knowing PW and NW can be useful in preference elicitation

[Pini, Rossi, Venable, Walsh, IJCAI 2007]

Preference aggregation function: example with Pareto

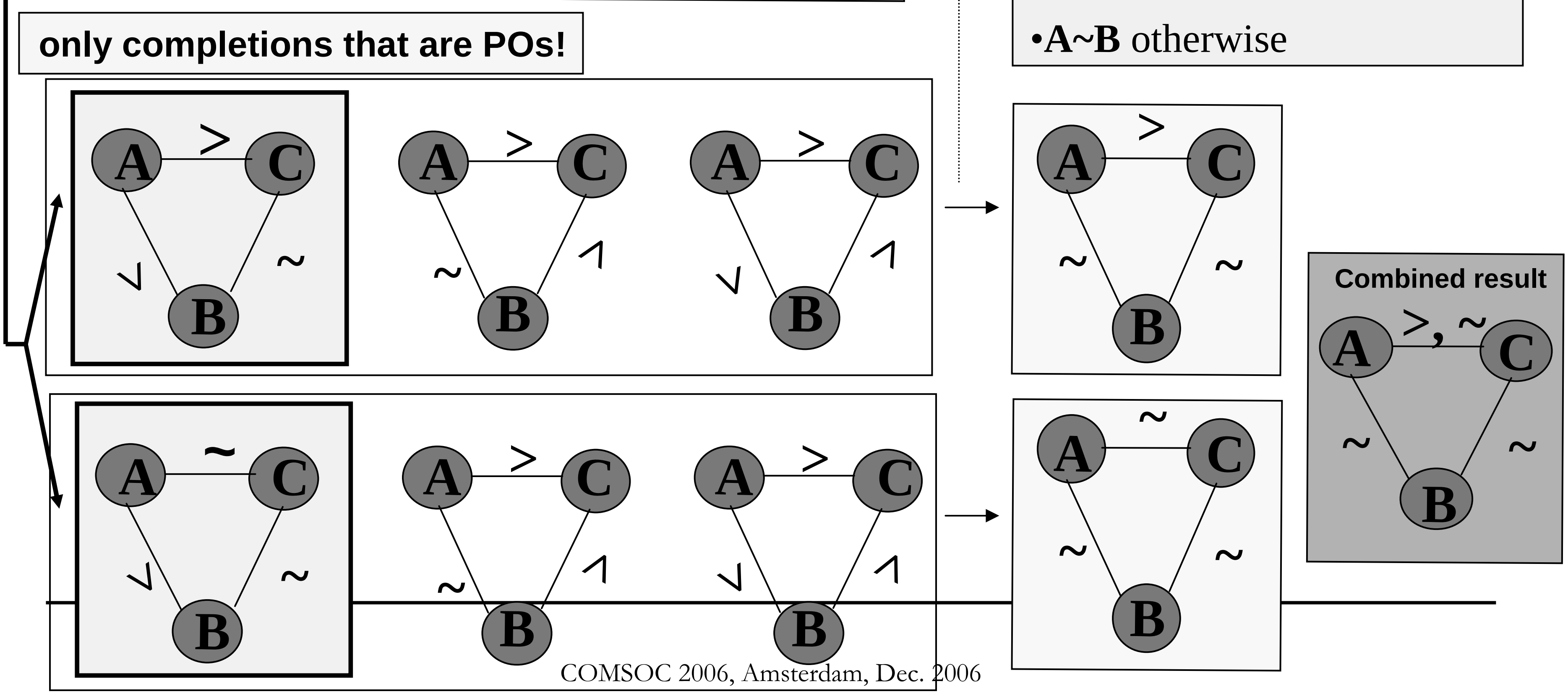


Pref. aggr. function:

incomplete profiles → sets of P0s

Pareto: POs → PO

- **A>B** iff A>B or A=B for all agents, and A>B for at least one
- **A~B** otherwise

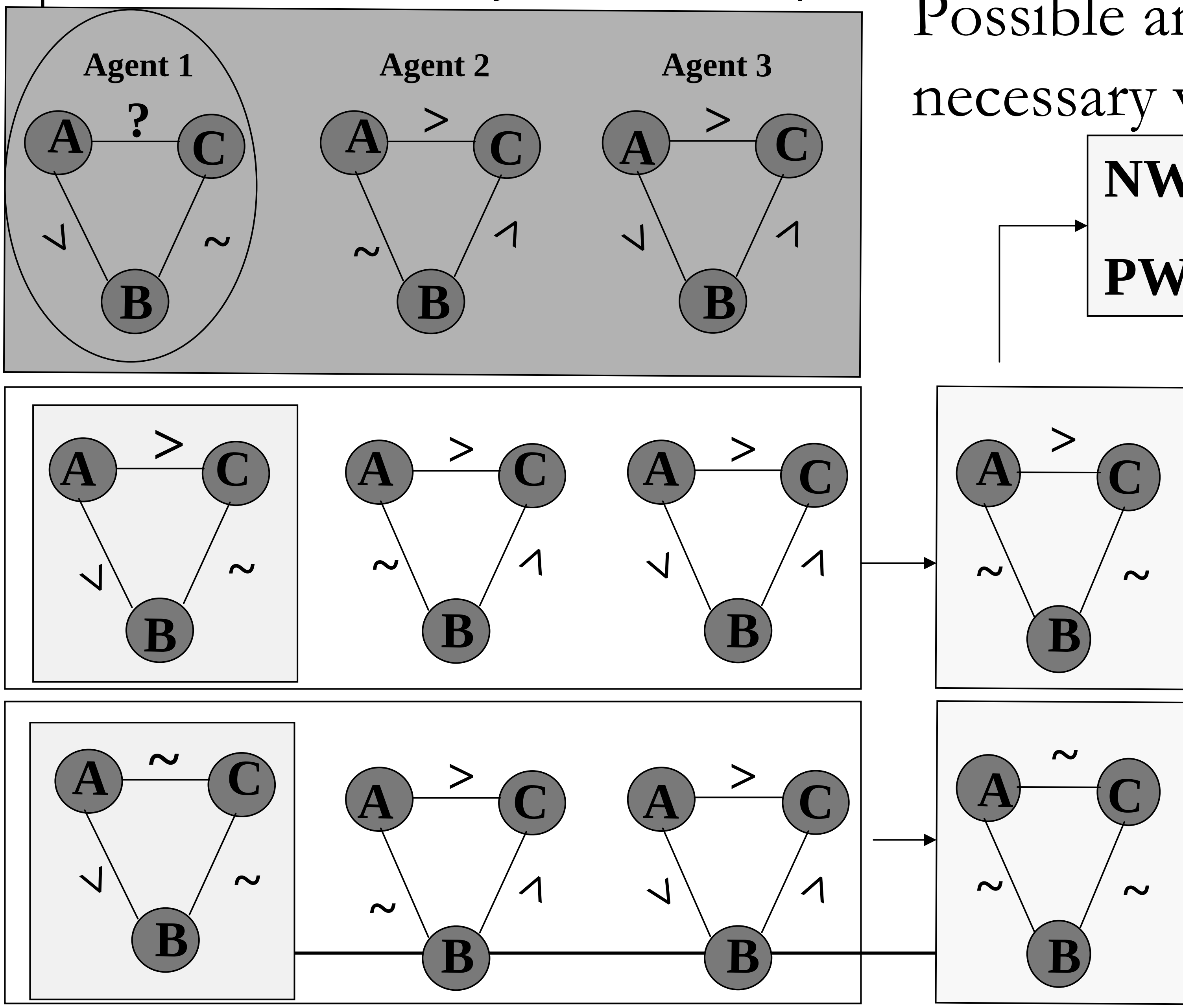


Combined result

- Graph where
  - nodes = candidates
  - all arcs
  - label of arc A-B: set of all relations between A and B, such that each relation in at least one result

Possible and necessary winners: example with Pareto

Possible and necessary winners



NW={A,B}

PW={A,B,C}

PW and NW: complexity results

- Computing PW and NW is NP-hard

(even restricting to incomplete TOs)

  - deciding if an outcome is
    - a possible winner: NP-complete
    - a necessary winner: coNP-complete

- Computing *good approximations* of PW and NW is NP-hard
  - good approximation: for all k integer >1, a superset PW* s.t. |PW*| < k |PW|

PW and NW: easy from combined result

- Combined result: graph where
  - nodes = candidates
  - all arcs
  - label of arc A-B: set of all relations between A and B, such that each relation in at least one result
- Given the combined result, PW and NW are easy to find
  - A in NW if no arc (A-B) with B>A
  - A in PW if all arcs (A-B) with B>A contain also other labels
- Computing the combined result: in general NP-hard

PW and NW: a tractable case

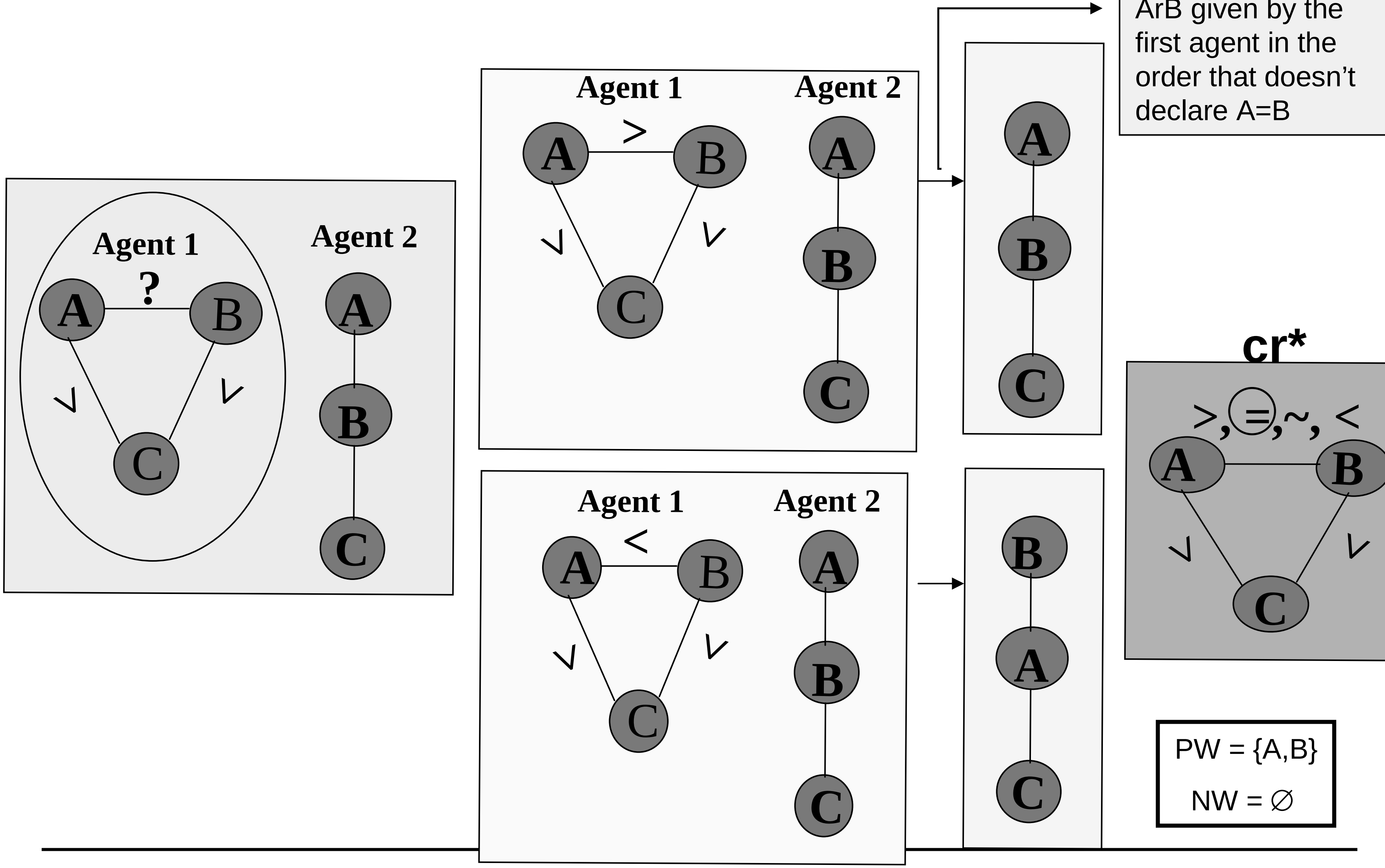
- If f is IIA and monotonic
  - we can compute an upper approximation (cr*) in polynomial time
  - Also, given cr*, polynomial to compute PW and NW
    - algorithm not affected by approximation
- **IIA**: when rel(A,B) in the result depends only by rel(A,B) given by the agents
- **Monotonic**: when we improve an outcome in a profile (for ex. we pass from A>B to A=B), then it improves also in the result

Cr*: upper approximation of the combined result

- Consider two profile completions:
  - (A?B) replaced with (A>B) for every agent
  - (A?B) replaced with (A<B) for every agent
  - Two results: (A r_1 B) and (A r_2 B)
- In cr*, put (A r B) where r is {r_1,r_2,everything between them}
- Order of relations: <, = and ~, >
- Thm.: f is IIA and monotonic ➔ cr* upper approx.of cr
- Approximation only on arcs with all four labels
  - involves only = and ~

Example: Lex

Lex:
agents are ordered, ArB given by the first agent in the order that doesn't declare A=B



Preference elicitation - (1)

- Process of asking queries to agents in order to determine their preferences over outcomes

  [Chen and Pu, 2004]

- At each stage in eliciting preference there is a set of possible and necessary winners
- PW = NW $\rightarrow$ preference elicitation is over, no matter how incompleteness is resolved
- Checking when PW = NW: hard in general

  [Conitzer and Sandholm, 2002]

- Pref.elicitation is easy if f IIA+ pol. computable

Preference elicitation - (2)

- PW = NW → preference elicitation is over
  - At the beginning: NW=∅ PW=Ω
  - As preferences are declared: NW ↑ PW ↓
  - If PW ⊃ NW, and A∈PW–NW, A can become a loser or a necessary winner
    - Enough to perform **ask(A,B),** ∀**B**∈**PW**
      - C∉PW is a loser → dominated
      - f is **IIA** → ask(A,B) involves only A-B preferences
  - O(|PW|^2) steps to remove enough incompleteness to know the winners

A specific voting rule with two kinds of uncertainty: sequential majority voting

Sequential Majority voting

- Knock-out competitions, modelled by a binary tree T
- Result of each competition given by majority graph
- r_T: majority graph G → candidate (winner)

Uncertainty

- In sequential majority voting, we consider two kinds of uncertainty
    1. No knowledge about the voting tree
    2. Partial knowledge about the agents' preferences
- We start with the first kind, then we add the second kind
- Complexity of finding possible/necessary winners

[Lang, Pini, Rossi, Venable, Walsh, IJCAI 2007]

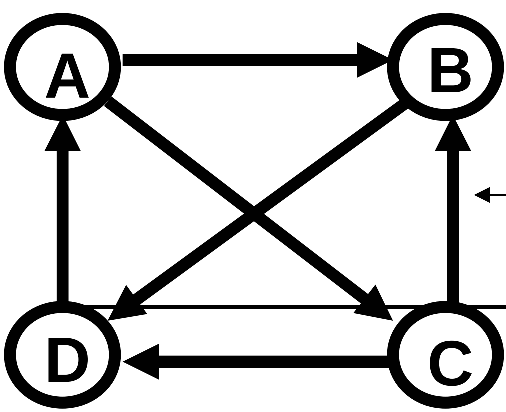
First type of uncertainty

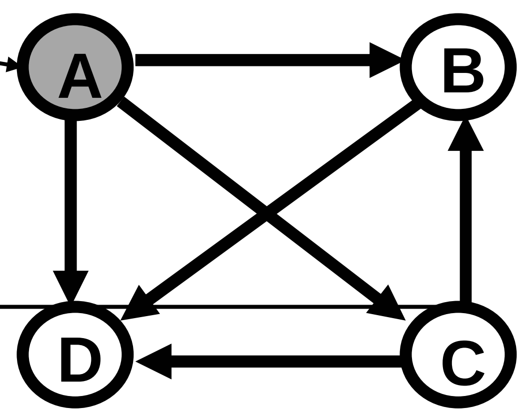
- Complete agents' preferences
- No knowledge of the tree of knock-out competitions

Condorcet (necessary) winner

- Given a complete profile P, a candidate A is a Condorcet winner iff $\forall$T, binary tree, $r_T(M(P))=A$.
- Given M(P), A is a Condorcet winner iff its node in M(P) has only outgoing edges
- Polynomial time

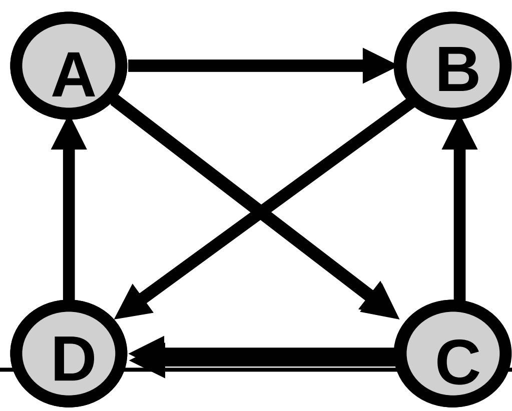
If ∃, then unique

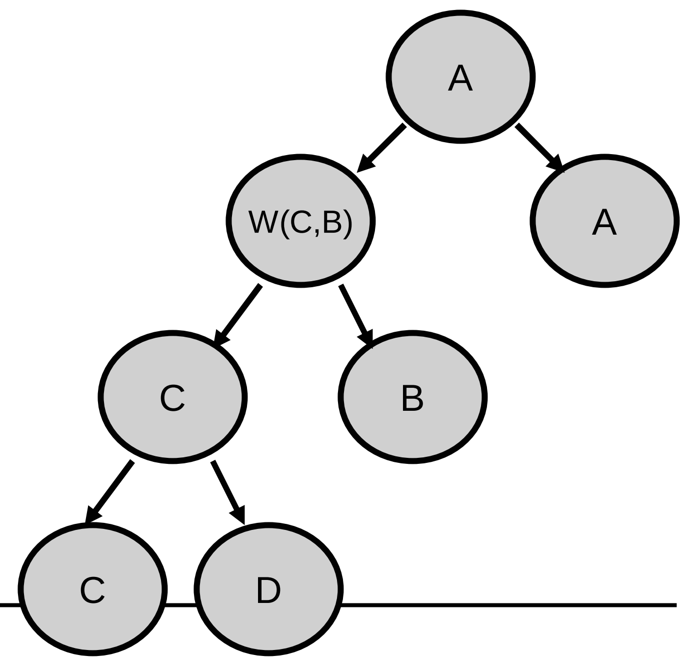




Possible winners

- Given a complete profile P, candidate A is a possible winnner iff ∃T, binary tree, such that $r_T(M(P))=A$.
- Given M(P), candidate A is a possible winner iff there is path from node A to every other node.
- Polynomial time





Manipulation

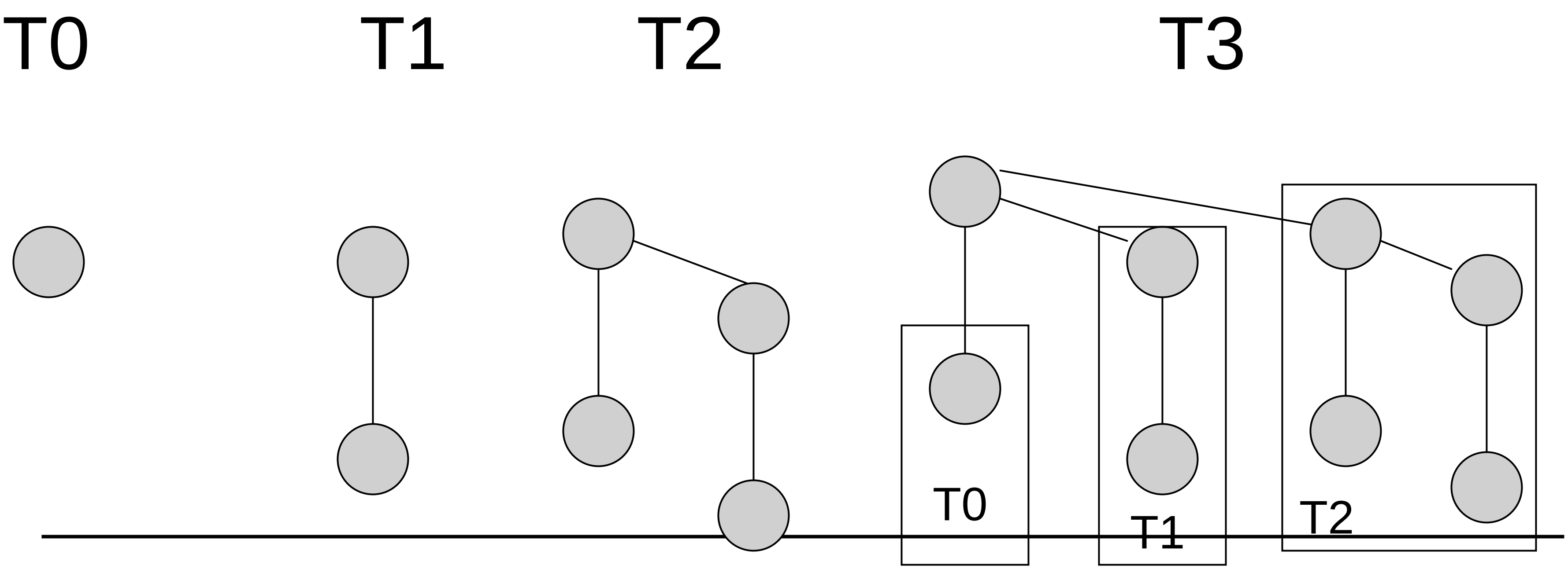
- Given the majority graph, the chair can easily check if A can win, and can find a tree where it wins ➔ easy to manipulate by the chair
- Can we make it difficult for the chair to manipulate the result?
- We can do that by imposing some restrictions on the trees

Fair Possible Winners

- Some possible winners may win only on very unbalanced trees, competing only few times
- Given majority graph M(P) of profile P, A is a fair possible winner iff ∃T balanced voting tree such that $r_T(M(P))=A$
- We want to know how difficult it is to recognize fair possible winners

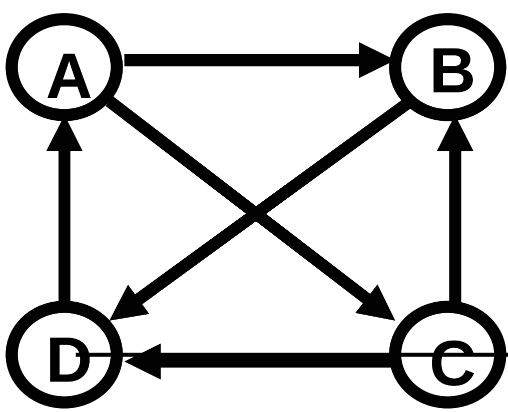
Binomial trees

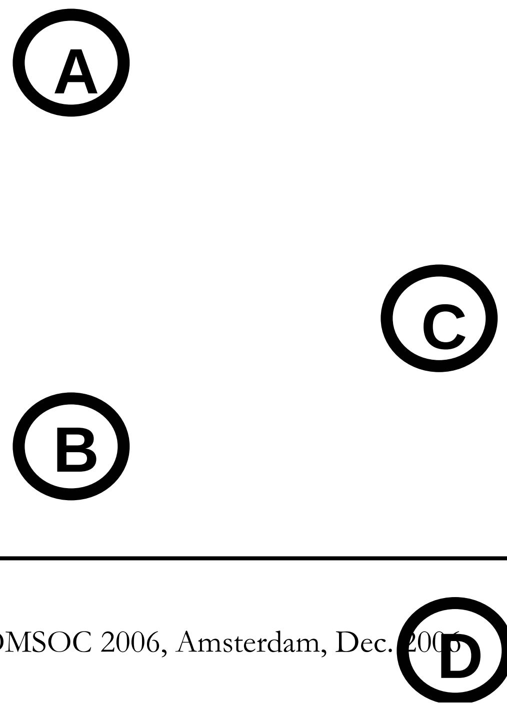
- Binomial tree
  - T_0 → 1 node
  - T_K → the root has k children and the i-th child is the root of a T_{k-I}
- T_k has 2^k nodes

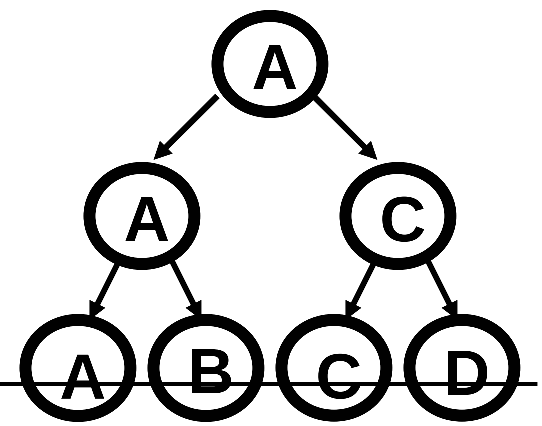


Determining fair possible winners

- Given a majority graph G with 2^k nodes, candidate A is a fair possible winner iff exists a binomial tree T_k:
  - Covering G (arrows from father to child)
  - Rooted at A







Complexity of determining fair possible winners

- Th.: Given a complete weighted majority graph G and a candidate A, testing if there is a spanning binomial tree with root A is NP-complete
  - Proof: Reduction from the Exact Cover problem

- Weighted majority graphs are used in social choice theory and weights may represent for example the amount of disagreement, or the elicitation cost

- A standard majority graphs = weighted maj. graphs with all identical weights

Both types of uncertainty

- Missing preferences ➔ Missing arcs in the majority graph
- Unknown voting tree
- New notions:
  - **Weak possible (WP) winner A**:
    ∃completion of maj. graph, ∃ voting tree s.t. A wins
  - **Strong possible (SP) winner A**:
    ∀completion of maj. graph, ∃ voting tree s.t. A wins
  - **Weak Condorcet (WC) winner A**:
    ∃ completion of maj. graph, ∀voting tree s.t. A wins
  - **Strong Condorcet (SC) winner A:**
    ∀completion of maj. graph, ∀voting tree s.t. A wins
- $SC \subseteq WC \cap SP$
- $WC \cup SP \subseteq WP$

Determining WP, SP, WC, SC is easy

- A is a strong possible (SP) winner iff, $\forall$B, there is a path from A to B in G
- A is a weak possible (WP) winner iff it is possible to complete the majority graph such that every outcome is reachable from A
- A is a strong Condorcet (SC) winner iff A has m-1 outgoing edges
- A is weak Condorcet (WC) winner iff A has no ingoing edges

Incomplete profiles

- Some completions of an incomplete majority graph do not correspond to any completion of the incomplete profile
  - Agents' preferences are transitive
- SP', WP', SC', WC' defined using incomplete profiles rather than incomplete majority graphs
- Results:
  - WC'=WC ➔ easy to compute (same for SC')
  - Conjecture: WP' and SP' difficult to compute
  - Fairness (balanced tree) ➔ difficult (with weights)

Another role for CP nets and soft constraints

- Not just for represently each agent's preferences compactly
- Solving tools to compute optimals when aggregating preferences as in game theory

Games vs. CP nets and soft constraints

- Nash equilibria in games = optimals in CP nets
  - Finding a Nash equilibrium is as difficult as finding an optimal solution in a CP net
  - Tractability results in CP nets and soft constraints can be exploited when finding Nash equilibria
- Optimals in soft constraints vs. other notions of optimality inm games (such as Pareto optimality)

[Apt, Rossi, Venable, Proc. CIRAS 2006]

Conclusions

- Compact preference modelling
  - Formalisms and solving tools
- Multi-agent setting: normative and computational properties
  - Incomparability does not help for fairness or strategy proofness: usual (im)possibility results
  - When preferences are incomplete: difficult to compute possible and necessary winners, but easy under certain conditions

Future work

- Modelling preferences: comparison/merge with other frameworks
  - E.g.: strategic games vs. CP nets and soft constraints
- Positive and negative preferences
  - Representing and aggregating them
- Compact preference formalisms in multi-agent preference aggregation
  - Related to judgement aggregation