Le choix social computationnel rencontre l'informatique mathématique

Jérôme Lang
LAMSADE
CNRS – Université Paris-Dauphine

GDR Informatique Mathématique, 11 mars 2019

Plan

1. Introduction: computational social choice
2. Voting rules, easy and hard
3. Multi-winner rules
4. Voting protocols
5. Fair division
6. Conclusion

Plan

1. Introduction: computational social choice
2. Voting rules, easy and hard
3. Multi-winner rules
4. Voting protocols
5. Fair division
6. Conclusion

Social choice theory

- Social choice: *designing and analysing methods for collective decision making*
- Some examples of social choice problems:
  - political elections. Voting
  - finding a date for a meeting. Voting
  - deciding where and when to have dinner altogether tonight. Voting
  - in a high school: deciding who gets which class and who teaches when. Fair division
  - in a company: find a partition of employees in groups of people who will work together. Coalition structure formation
  - in a jury: agreeing on a verdict. Judgment aggregation

Social choice theory

- Social choice: *designing and analysing methods for collective decision making*
- Some examples of social choice problems:
  - political elections. Voting
  - finding a date for a meeting. Voting
  - deciding where and when to have dinner altogether tonight. Voting
  - in a high school: deciding who gets which class and who teaches when. Fair division
  - in a company: find a partition of employees in groups of people who will work together. Coalition structure formation

↑ aggregating preferences

↓ aggregating beliefs

  - in a jury: agreeing on a verdict. Judgment aggregation

Preferences

- each agent i has some preferences on the alternatives

Most usual models:

- *cardinal preferences*: each agent has a *utility function* $u : C \to \mathbb{R}$
- *ordinal preferences*: each agent has a *preference relation* $\succeq$ on C (most common assumption in social choice)
- *dichotomous preferences*: each agent has a partition $\{Good, Bad\}$ of C

A very rough history of social choice

1. end of 18th century: early stage, with Condorcet and Borda (session spéciale, réunion du GDR IM, Versailles, juin 1789)
2. 1951: birth of modern social choice
   - results are mainly *axiomatic* (economics/mathematics)
     - impossibility theorems: *incompatibility of a small set of seemingly innocuous conditions*, such as Arrow's theorem:

       With at least 3 alternatives, an aggregation function satisfies *unanimity* and *independence of irrelevant alternatives* if and only if it is a *dictatorship*.
   - computational issues are neglected
3. early 90's: computer scientists come into play

   $\Rightarrow$ **Computational social choice**: using computational notions and techniques (mainly from Artificial Intelligence, Operations Research, Theoretical Computer Science) for solving complex collective decision making problems.

Plan

1. Introduction: computational social choice
2. Voting rules, easy and hard
3. Multi-winner rules
4. Voting protocols
5. Fair division
6. Conclusion

Voting

1. a finite *set of voters* $N = \{1, \dots, n\}$;
2. a finite *set of candidates* C
3. a *profile* = a collection of n preference relations

$$P = (\succ_1, \dots, \succ_n)$$

$\succ_i$ = linear order over C = *vote* expressed by voter i.

Here is a 100-voter profile over $C = \{a, b, c, d, e\}$

33 votes:	$a \succ b \succ c \succ d \succ e$
16 votes:	$b \succ d \succ c \succ e \succ a$
3 votes:	$c \succ d \succ b \succ a \succ e$
8 votes:	$c \succ e \succ b \succ d \succ a$
18 votes:	$d \succ e \succ c \succ b \succ a$
22 votes:	$e \succ c \succ b \succ d \succ a$

Resolute vs. irresolute rules

The usual way of defining voting rules:

- we first define an irresolute voting rule F

$$P \mapsto F(P) \in 2^C \setminus \{\emptyset\} \text{ (cowinners)}$$

- a resolute rule is defined from F by using a tie-breaking priority T
- usual assumption: T = linear order on C
- $F_T(P) = \max(T, F(P))$: F_T resolute rule

Example:

- $P = \langle a \succ b, b \succ a \rangle$
- Maj irresolute voting rule: $Maj(P) = \{a, b\}$
- $Maj_{a>b}$ and $Maj_{b>a}$ resolute voting rules
- $Maj_{a>b}(P) = a$

In the rest of the talk, we usually define irresolute rules, from which resolute rules are induced by a tie-breaking priority.

Voting

$X = \{a, b, c, d, e\}$

33 votes:	$a \succ b \succ c \succ d \succ e$
16 votes:	$b \succ d \succ c \succ e \succ a$
3 votes:	$c \succ d \succ b \succ a \succ e$
8 votes:	$c \succ e \succ b \succ d \succ a$
18 votes:	$d \succ e \succ c \succ b \succ a$
22 votes:	$e \succ c \succ b \succ d \succ a$

Who should be elected?

Positional scoring rules

- n voters, m candidates
- fixed list of m integers $s_1 \geq \ldots \geq s_m$, with $s_1 > s_m$
- if voter i ranks candidate x in position j then $score_i(x) = s_j$
- winner(s): candidate(s) maximizing

$$s(x) = \sum_{i=1}^{n} score_i(x)$$

plurality $s_1 = 1,\ s_2 = \ldots = s_m = 0 \mapsto$ winner: a

Borda $s_1 = m-1,\ s_2 = m-2,\ \ldots s_m = 0 \mapsto$ winner: b

Majority graph

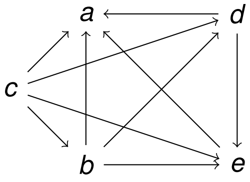
Generalizing simple majority:

pairwise majority
: given any two alternatives $x, y \in X$, use simple majority to determine whether the group prefers x to y or vice versa.

Does this work? Sometimes yes:

33 votes:	$a \succ b \succ c \succ d \succ e$
16 votes:	$b \succ d \succ c \succ e \succ a$
3 votes:	$c \succ d \succ b \succ a \succ e$
8 votes:	$c \succ e \succ b \succ d \succ a$
18 votes:	$d \succ e \succ c \succ b \succ a$
22 votes:	$e \succ c \succ b \succ d \succ a$

associated majority graph



Collective preference relation: $c \succ b \succ d \succ e \succ a$

Winner: c

Majority graph

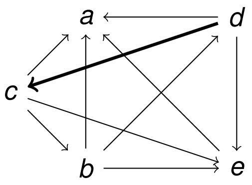
Generalizing simple majority:

pairwise majority
: given any two alternatives $x, y \in X$, use simple majority to determine whether the group prefers x to y or vice versa.

Does this work? Sometimes no:

33 votes:	$a \succ b \succ \mathbf{d} \succ \mathbf{c} \succ e$
16 votes:	$b \succ d \succ c \succ e \succ a$
3 votes:	$c \succ d \succ b \succ a \succ e$
8 votes:	$c \succ e \succ b \succ d \succ a$
18 votes:	$d \succ e \succ c \succ b \succ a$
22 votes:	$e \succ c \succ b \succ d \succ a$

associated majority graph

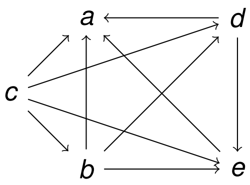


Collective preference relation: $\{b \succ c \succ d \succ b \succ ...\} \succ e \succ a$;

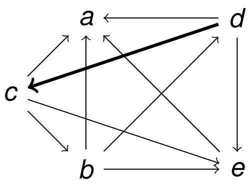
Winner: ?

Condorcet winner

- $N(x, y) = \#\{i, x \succ_i y\}$ number of voters who prefer x to y.
- x *Condorcet winner* if for all $y \neq x$, $N(x, y) > \frac{n}{2}$



c Condorcet winner



no Condorcet winner

- sometimes there is no Condorcet winner
- when there is a Condorcet winner, it is unique
- a rule is *Condorcet-consistent* if it outputs the Condorcet winner whenever there is one.

Rules based on the majority graph

- P profile $\mapsto M(P)$ directed graph associated with P
- A voting rule r is *based on the majority graph* if $r(P) = f(M(P))$ for some function f.
- For simplicity, assume an odd number of voters: the majority graph is a complete asymmetric graph (a *tournament*).

Copeland rule

- $Cop(x) =$ number of candidates y such that $M(P)$ contains $x \longrightarrow y$.
- Copeland winner(s): $\text{argmax}_{c \in C} Cop(x)$.

$$\begin{tikzcd}
a \arrow[r] \arrow[d] & c \arrow[d] \\
b \arrow[r] \arrow[ur] & d \arrow[ul]
\end{tikzcd}$$

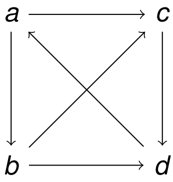
$$C(a) = 2$$
$$C(b) = 2$$
$$C(c) = 1$$
$$C(d) = 1$$

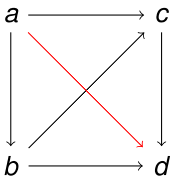
Copeland cowinners: a, b

Rules based on the majority graph

Slater rule

- Slater ranking = linear order on C obtained by inverting as few edges as possible in $M(P)$
- Slater winner: best candidate in some Slater ranking





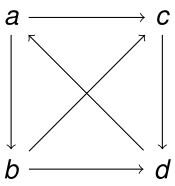
Slater winner: a

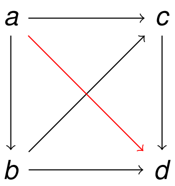
- finding a Slater ranking is equivalent to finding an instance of the *minimum feedback arc set problem*

Rules based on the majority graph

Slater rule

- Slater ranking = linear order on C obtained by inverting as few edges as possible in $M(P)$
- Slater winner: best candidate in some Slater ranking





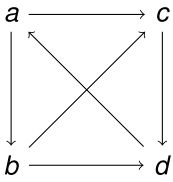
Slater winner: a

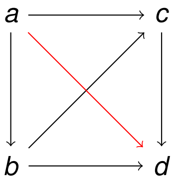
- finding a Slater ranking is equivalent to finding an instance of the *minimum feedback arc set problem*
- deciding whether an alternative is a Slater cowinner is NP-hard

Rules based on the majority graph

Slater rule

- Slater ranking = linear order on C obtained by inverting as few edges as possible in $M(P)$
- Slater winner: best candidate in some Slater ranking





Slater winner: a

- finding a Slater ranking is equivalent to finding an instance of the *minimum feedback arc set problem*
- deciding whether an alternative is a Slater cowinner is NP-hard
- it is not known whether the problem is in NP; the best upper bound we know is Θ_2^p.

Rules based on the majority graph

Banks rule

- look for the maximal subsets C' or C such that the restriction of $M(P)$ to C is transitive.
- the restriction of $M(P)$ to these subsets are called *maximal transitive subtournaments* of $M(P)$
- x is a Banks winner if x is dominating in some maximal subtournament of $M(P)$.

$$
\begin{tikzcd}
a \arrow[r] \arrow[d] & c \arrow[d] \\
b \arrow[r] \arrow[ur] & d \arrow[ul]
\end{tikzcd}
$$

Maximal subtournaments of $M(P)$:

- $\{a, b, c\}$ winner: a
- $\{b, c, d\}$ winner: b
- $\{a, d\}$ winner: d

Banks cowinners: a, b, d

- deciding whether x is a Banks winner is NP-complete

Rules based on the majority graph

Banks rule

- look for the maximal subsets C' or C such that the restriction of $M(P)$ to C is transitive.
- the restriction of $M(P)$ to these subsets are called *maximal transitive subtournaments* of $M(P)$
- x is a Banks winner if x is dominating in some maximal subtournament of $M(P)$.

$$\begin{tikzcd} a \arrow[r] \arrow[d] & c \arrow[d] \\ b \arrow[r] \arrow[ur] & d \arrow[ul] \end{tikzcd}$$

Maximal subtournaments of $M(P)$:

- $\{a, b, c\}$ winner: a
- $\{b, c, d\}$ winner: b
- $\{a, d\}$ winner: d

Banks cowinners: a, b, d

- deciding whether x is a Banks winner is NP-complete
- but *some* Banks cowinner can be found in polynomial time by a greedy algorithm

Rules based on the weighted majority graph

- $N_P(x,y) = \#\{i, x \succ_i y\}$ number of voters who prefer x to y.
- A voting rule r is *based on the weighted majority graph* if $r(P) = g(N_P)$ for some function g.

maximin rule

- maximin score: $S_m(x) = \min_{y \neq x} N_P(x,y)$
- winner(s) maximize $S_m(x)$

N_P	a	b	c	d	e	S_m
a	–	33	33	33	36	33
b	67	–	49	79	52	49
c	67	51	–	33	60	33
d	67	21	67	–	70	21
e	66	48	40	30	–	30

maximin winner: b

Rules based on the weighted majority graph

Kemeny rule

- for each ranking of candidates R, the Kemeny score of R is

$$K(R) = \sum \{N_P(x,y) \mid (x,y) \text{ such that } x >_R y\}$$

- Kemeny consensus = ranking with maximal Kemeny score
- Kemeny winner: best candidate in some Kemeny consensus

N_P	a	b	c	d	e
a	–	33	33	33	36
b	67	–	49	79	52
c	67	51	–	33	60
d	67	21	67	–	70
e	66	48	40	30	–

$K(bcdea) = ?$

Rules based on the weighted majority graph

Kemeny rule

- for each ranking of candidates R, the Kemeny score of R is

$$K(R) = \sum_{x,y \mid x >_R y} N(x,y)$$

- Kemeny consensus = ranking with maximal Kemeny score
- Kemeny winner: best candidate in some Kemeny consensus

N_P	a	b	c	d	e
a	–	33	33	33	36
b	67	–	49	79	52
c	67	51	–	33	60
d	67	21	67	–	70
e	66	48	40	30	–

$K(bcdea) = 610$

Rules based on the weighted majority graph

Kemeny rule

- for each ranking of candidates R, the Kemeny score of R is

$$K(R) = \sum_{x,y|x>_R y} N(x,y)$$

- Kemeny consensus = ranking with maximal Kemeny score
- Kemeny winner: best candidate in some Kemeny consensus

N_P	a	b	c	d	e
a	–	33	33	33	36
b	67	–	49	79	52
c	67	51	–	33	60
d	67	21	67	–	70
e	66	48	40	30	–

$K(bcdea) = 610$
$K(bdcea) = 644$
Kemeny consensus: $bdcea$
Kemeny winner: b

Rules based on the weighted majority graph

Kemeny rule

- Complexity
  Winner determination is Θ_2^P-complete (needs logarithmically many NP-oracles)
- Polynomial approximation
  - a $4/3$-approximation algorithm based on linear programming
  - a $11/7$-approximation algorithm (more sophisticated)
  - existence of a polynomial-time approximation scheme (but not efficient in practice)
- Parametrized complexity
  Winner can be computed in time $O(2^m m^2 n)$
- Practical algorithms
  - translation into ILP
  - branch and bound,
  - heuristic search based on Borda scores
  - etc.

Computing voting rules

Three classes of rules:

- winner determination in P: easy to compute
  - positional scoring rule, Copeland, maximin, and others
- winner determination is NP-complete: not easy to compute but easy to verify a solution using a succinct certificate
  - Banks, and others
- winner determination is beyond NP: not even easy to verify.
  - Kemeny, probably Slater (?), and others

Is there a life after NP-hardness?

- efficient computation: design algorithms that do as well as possible, possibly using heuristics, or translations into well-known frameworks (such as integer linear programming).
- fixed-parameter complexity: isolate the components of the problem and find the main cause(s) of hardness.
- approximation: design algorithms that produce a (generally suboptimal) result, with some performance guarantee.
  - The approximation of a voting rule is a new voting rule that may be interesting *per se*.

Plan

1. Introduction: computational social choice
2. Voting rules, easy and hard
3. Multi-winner rules
4. Voting protocols
5. Fair division
6. Conclusion

Condorcet winning sets

- S is a Condorcet winning set for profile P if for each $x \notin S$, a majority of votes rank at least one element of S above x.
- $P = (abcd, cdab, dabc)$:
  - $\{a, c\}$ is a Condorcet winning set;
  - $\{b, c\}$ is not a Condorcet winning set because of a.
- Condorcet dimension of a profile P = cardinality of the smallest Condorcet winning set for P

Condorcet winning sets

- S is a Condorcet winning set for profile P if for each $x \notin S$, a majority of votes rank at least one element of S above x.
- $P = (abcd, cdab, dabc)$:
  - $\{a, c\}$ is a Condorcet winning set;
  - $\{b, c\}$ is not a Condorcet winning set because of a.
- Condorcet dimension of a profile P = cardinality of the smallest Condorcet winning set for P
- P has Condorcet dimension 1 if it has a Condorcet winner

Condorcet winning sets

- S is a Condorcet winning set for profile P if for each $x \notin S$, a majority of votes rank at least one element of S above x.
- $P = (abcd, cdab, dabc)$:
  - $\{a, c\}$ is a Condorcet winning set;
  - $\{b, c\}$ is not a Condorcet winning set because of a.
- Condorcet dimension of a profile P = cardinality of the smallest Condorcet winning set for P

- P has Condorcet dimension 1 if it has a Condorcet winner
- $P = (abcd, cdab, dabc)$ has Condorcet dimension 2: no Condorcet winner; $\{a, c\}$ Condorcet winning set.

Condorcet winning sets

- S is a Condorcet winning set for profile P if for each $x \notin S$, a majority of votes rank at least one element of S above x.
- $P = (abcd, cdab, dabc)$:
  - $\{a, c\}$ is a Condorcet winning set;
  - $\{b, c\}$ is not a Condorcet winning set because of a.
- Condorcet dimension of a profile P = cardinality of the smallest Condorcet winning set for P

- P has Condorcet dimension 1 if it has a Condorcet winner
- $P = (abcd, cdab, dabc)$ has Condorcet dimension 2: no Condorcet winner; $\{a, c\}$ Condorcet winning set.
- there exists a 6-candidate 6-voter profile of Condorcet dimension 3.

Condorcet winning sets

- S is a Condorcet winning set for profile P if for each $x \notin S$, a majority of votes rank at least one element of S above x.
- $P = (abcd, cdab, dabc)$:
  - $\{a, c\}$ is a Condorcet winning set;
  - $\{b, c\}$ is not a Condorcet winning set because of a.
- Condorcet dimension of a profile P = cardinality of the smallest Condorcet winning set for P

- P has Condorcet dimension 1 if it has a Condorcet winner
- $P = (abcd, cdab, dabc)$ has Condorcet dimension 2: no Condorcet winner; $\{a, c\}$ Condorcet winning set.
- there exists a 6-candidate 6-voter profile of Condorcet dimension 3.
- does there exist a profile of Condorcet dimension n, for all n?

Condorcet winning sets

- S is a Condorcet winning set for profile P if for each $x \notin S$, a majority of votes rank at least one element of S above x.
- $P = (abcd, cdab, dabc)$:
  - $\{a, c\}$ is a Condorcet winning set;
  - $\{b, c\}$ is not a Condorcet winning set because of a.
- Condorcet dimension of a profile P = cardinality of the smallest Condorcet winning set for P

- P has Condorcet dimension 1 if it has a Condorcet winner
- $P = (abcd, cdab, dabc)$ has Condorcet dimension 2: no Condorcet winner; $\{a, c\}$ Condorcet winning set.
- there exists a 6-candidate 6-voter profile of Condorcet dimension 3.
- does there exist a profile of Condorcet dimension n, for all n? *Nobody knows.*

Condorcet and θ-winning sets

θ-winning sets

- S is a θ-winning set if for each $x \notin S$, the proportion of votes that rank at least one element of S above x is at least θ
- Condorcet winning set = $\frac{1}{2}$-winning set
- for fixed k: $\theta(S,k)$ is the highest value of θ for which S is a θ-winning set of size k; output subsets maximizing $\theta(S,k)$

$$\begin{array}{ll} \times 4: & a \succ b \succ c \succ e \succ d \succ f \\ \times 3: & b \succ c \succ a \succ e \succ d \succ f \\ \times 2: & f \succ e \succ d \succ b \succ c \succ a \end{array}$$

- $k = 1 \mapsto \{a\}$ (maximin winner)
- $k = 2 \mapsto \{a,b\}$
- $k = 3 \mapsto \{a,b,f\}$

Plan

1. Introduction: computational social choice
2. Voting rules, easy and hard
3. Multi-winner rules
4. Voting protocols
5. Fair division
6. Conclusion

Multiple round rules

Plurality with runoff

- let x, y the two candidates with the highest plurality score (use tie-breaking rule if necessary)
- winner: majority winner between x and y

33	a $\succ$b $\succ$c $\succ$d $\succ$e
16	b $\succ$d $\succ$c $\succ$e $\succ$a
3	c $\succ$d $\succ$b $\succ$a $\succ$e
8	c $\succ$e $\succ$b $\succ$d $\succ$a
18	d $\succ$e $\succ$c $\succ$b $\succ$a
22	e $\succ$c $\succ$b $\succ$d $\succ$a

- first step: keep a and e
- winner: e

Multiple round rules

Single transferable vote (STV)

Repeat

$x :=$ candidate ranked first by the fewest voters;
eliminate x from all ballots
{votes for x transferred to the next best remaining candidate}

Until some candidate y is ranked first by more than half of the votes;
Winner: y

- When there are only 3 candidates, STV coincides with plurality with runoff.
- STV is used for political elections in several countries (at least Australia and Ireland)

Single transferable vote (STV)

33	$a \succ b \succ c \succ d \succ e$
16	$b \succ d \succ c \succ e \succ a$
3	$c \succ d \succ b \succ a \succ e$
8	$c \succ e \succ b \succ d \succ a$
18	$d \succ e \succ c \succ b \succ a$
22	$e \succ c \succ b \succ d \succ a$

33	$a \succ b \succ d \succ e$
16	$b \succ d \succ e \succ a$
3	$d \succ b \succ a \succ e$
8	$e \succ b \succ d \succ a$
18	$d \succ e \succ b \succ a$
22	$e \succ b \succ d \succ a$

33	$a \succ d \succ e$
16	$d \succ e \succ a$
3	$d \succ a \succ e$
8	$e \succ d \succ a$
18	$d \succ e \succ a$
22	$e \succ d \succ a$

33	$a \succ d$
16	$d \succ a$
3	$d \succ a$
8	$d \succ a$
18	$d \succ a$
22	$d \succ a$

winner: d

Single transferable vote (STV)

(*) How do we handle ties in STV?

STV^T ties are broken immediately using a tie-breaking priority T: polynomial

STV^{PU} exploring all possibilities and possible use tie-breaking at the very last moment: NP-complete

4	$a \succ d \succ b \succ c$
3	$b \succ c \succ d \succ a$
2	$c \succ d \succ a \succ b$
2	$d \succ b \succ c \succ a$

Tie-breaking : $a > b > d > c$

- break ties immediately: c eliminated, then b, winner: d
- parallel universes:
  - branch 1 (above): winner: d
  - branch 2: d eliminated, then c, winner: a
  - cowinners $\{a, d\}$, winner: a.

Computing voting rules

Three classes of rules:

- winner determination in P: easy to compute
  - positional scoring rule, Copeland, maximin, plurality with runoff, STV^{T}, and others
- winner determination is NP-complete: not easy to compute but easy to verify a solution using a succinct certificate
  - Banks, STV^{PU}, and others
- winner determination is beyond NP: not even easy to verify.
  - Kemeny, probably Slater (?), and others

Communication complexity of voting rules

- *Voting rule*
  - profile $(V_1, \ldots, V_n) \mapsto$ winner(s) $r(V_1, \ldots, V_n)$
  - does not specify how the votes V_i are elicited from the voters by the central authority.
- *Protocol for a voting rule r*
  - informally: similar to an algorithm, except that instructions are replaced by communication actions, and such that communication actions are based on the *private information* of the agents.
  - V_i is the private information of agent (voter) i.
- *Communication complexity of a voting rule r*:
  - minimum cost of a protocol for r.

Communication complexity of voting rules

- An obvious protocol that works for *any* voting rule r:
  1. every voter i sends her vote V_i to the central authority
  2. the central authority sends back the name of the winner to all voters

  - step 1: $n \log(m!) = O(nm \log m)$ bits
  - step 2: ignored (or else: $n \log m$ bits)
    from now on, we shall ignore step the cost of information flow from the central authority to the voters.
- The communication complexity of an arbitrary voting rule r is in $O(nm \log m)$

Communication complexity: plurality with runoff

- An easy protocol for plurality with runoff:
    1. voters send the name of their most preferred candidate to the central authority
    2. the central authority sends the names of the two finalists to the voters
    3. voters send the name of their preferred finalist to the central authority

    - step 1: $n \log m$ bits
    - step 2: ignored (or else: $2n \log m$ bits)
    - step 3: n bits
    - total: $O(n(\log m))$
    - lower bound matches (Conitzer & Sandholm, 05)
- the communication complexity of plurality with runoff is in $\Theta(n. \log m)$

Communication complexity: STV

- A protocol for STV (Conitzer & Sandholm, 05)
  1. voters send their most preferred candidate to the central authority (C)
  2. let x be the candidate ranked first in the smallest number of votes. All voters who had x ranked first receive a message from C asking them to send the name of their next preferred candidate.
  3. repeat step 2 until there is a candidate ranked first in a majority of votes

  - after doing t times step 2: x ranked first in at most $\frac{n}{m-t}$ votes
  - cost of protocol

$$\leq n \log m(1 + 1/m + 1/m-1 + \ldots + 1/2) = O(n(\log m)^2)$$

  - lower bound $\Omega(n \log m)$
  - gap still open!

Plan

1. Introduction: computational social choice
2. Voting rules, easy and hard
3. Multi-winner rules
4. Voting protocols
5. Fair division
6. Conclusion

Fair division of indivisible objects

- a finite set of objects O, a finite set of agents $N = \{1, \dots, n\}$
- each agent i has a preference relation $\succeq_i$ over subsets of objects
  - Example: *additive preferences*.
  - the value of a set of objects of the sum of the values of its elements
  - $S \succeq_i S'$ if value of $S \geq$ value of S'

	a	b	c	d	e
Ann	1	1	0	1	0
Bob	1	1	0	1	0
Charles	0	0	1	0	1

$\mapsto$ allocation $\pi : N \to 2^O$ with $\pi(i) \neq \pi(j)$ for $i \neq j$

- notation: $[a|bc|de]$ is the allocation π where $\pi(Ann) = \{a\}$, $\pi(Bob) = \{b, c\}$ and $\pi(Charles) = \{d, e\}$.

Pareto-efficiency

- allocation π *Pareto-dominates* allocation π' if π' is at least as good as π for all agents and strictly better for some agent:
- for all i, $\pi(i) \succeq_i \pi'(i)$, and for some i, $\pi(i) \succ_i \pi'(i)$
- π is *Pareto-efficient* if it is not Pareto-dominated

	a	*b*	*c*	*d*	*e*
Ann	1	1	0	1	0
Bob	1	1	0	1	0
Charles	0	0	1	0	1

$[a|bc|de]$ not Pareto-efficient

Pareto-efficiency

- allocation π *Pareto-dominates* allocation π' if π' is at least as good as π for all agents and strictly better for some agent:
- for all i, $\pi(i) \succeq_i \pi'(i)$, and for some i, $\pi(i) \succ_i \pi'(i)$
- π is *Pareto-efficient* if it is not Pareto-dominated

	a	*b*	*c*	*d*	*e*
Ann	1	1	0	1	0
Bob	1	1	0	1	0
Charles	0	0	1	0	1

$[a|bc|de]$ not Pareto-efficient

	a	*b*	*c*	*d*	*e*
Ann	1	1	0	1	0
Bob	1	1	0	1	0
Charles	0	0	1	0	1

$[a|bd|ce]$ Pareto-efficient

Envy-freeness

- π is envy-free if no agent prefers the share of another agent to her own: for all i, j, $u_i(\pi(i)) \geq u_i(\pi(j))$

	a	*b*	*c*	*d*	*e*
Ann	1	1	0	1	0
Bob	1	1	0	1	0
Charles	0	0	1	0	1

$[a|bc|de]$ envy-free
but not Pareto-efficient

Envy-freeness

- π is envy-free if no agent prefers the share of another agent to her own: for all i, j, $u_i(\pi(i)) \geq u_i(\pi(j))$

	a	b	c	d	e
Ann	1	1	0	1	0
Bob	1	1	0	1	0
Charles	0	0	1	0	1

$[a|bc|de]$ envy-free
but not Pareto-efficient

	a	b	c	d	e
Ann	1	1	0	1	0
Bob	1	1	0	1	0
Charles	0	0	1	0	1

$[a|bd|ce]$ Pareto-efficient
but not envy-free

Pareto-efficiency and envy-freeness

	a	b	c	d	e
Ann	1	1	0	1	0
Bob	1	1	0	1	0
Charles	0	0	1	0	1

- $[a|bc|de]$ envy-free but not Pareto-efficient
- $[a|bd|ce]$ Pareto-efficient but not envy-free
- $[ab|d|ce]$ Pareto-efficient but not envy-free
- no allocation is both Pareto-efficient and envy-free

Pareto-efficiency and envy-freeness

	a	b	c	d	e
Ann	1	1	0	1	0
Bob	1	1	0	1	0
Charles	0	0	1	0	1

- $[a|bc|de]$ envy-free but not Pareto-efficient
- $[a|bd|ce]$ Pareto-efficient but not envy-free
- $[ab|d|ce]$ Pareto-efficient but not envy-free
- no allocation is both Pareto-efficient and envy-free
- relaxing Pareto-efficiency is not considered acceptable
- maybe envy-freeness is too strong and needs to be weakened

Proportional fair share

- n agents
- O set of objects
- agent i gives value $v_i(S)$ to $S \subseteq O$
- the proportional fair share value of i is

$$FS(i) = \frac{u_i(O)}{n}$$

- π satisfies the *proportional fair share* (PFS) property if for all i,

$$u_i(\pi(i)) \geq FS(i)$$

- for additive preferences, envy-freeness implies PFS

Proportional fair share

	a	b	c	d	e
Ann	1	1	0	1	0
Bob	1	1	0	1	0
$Charles$	0	0	1	0	1

- $FS(Ann) = FS(Bob) = 1$; $FS(Charles) = \frac{2}{3}$.
- $[ab|d|ce]$ Pareto-efficient and PFS (but not envy-free)

Proportional fair share

	a	b	c	d
Ann	10	5	7	0
Bob	9	6	7	2

- $FS(Ann) = 11$
- $FS(Bob) = 12$
- no (complete) allocation is fair share proportional
- perhaps PFS is still too strong

Maxmin fair share

- for each agent i, the maximin fair share value of i is her value of the worst share of the best possible partition

$$MaxMinFS(i) := \max_{\pi} \min_{j} u_i(\pi(j))$$

- π satisfies the maxmin fair share property if for all i,

$$u_i(\pi(i)) \geq MaxMinFS(i)$$

- for additive preferences:

$$\text{envy-freeness} \Rightarrow \text{PFS} \Rightarrow \text{MaxMinFS}$$

Maxmin fair share

	a	*b*	*c*	*d*
Ann	10	5	7	0
Bob	9	6	7	2

- $MaxMinFS(Ann) = \max_{\pi} \min_j u_{Ann}(\pi(j)) = 10$

Maxmin fair share

	a	b	c	d
Ann	10	5	7	0
Bob	9	6	7	2

- $MaxMinFS(Ann) = \max_{\pi} \min_j u_{Ann}(\pi(j)) = 10$
- $MaxMinFS(Bob) = \max_{\pi} \min_j u_{Bob}(\pi(j)) = 11.$

Maxmin fair share

	a	b	c	d
Ann	10	5	7	0
Bob	9	6	7	2

- $MaxMinFS(Ann) = \max_\pi \min_j u_{Ann}(\pi(j)) = 10$
- $MaxMinFS(Bob) = \max_\pi \min_j u_{Bob}(\pi(j)) = 11.$
- $[bc|ad]$ is MaxMinFS and Pareto-efficient (but not PEF)

Maxmin fair share

- with two agents and additive preferences, a maxmin fair share allocation always exists
- with non-additive preferences, MaxMinFS is not guaranteed, even for two agents

	≤ 1 item	*ab*	*ac*	*ad*	*bc*	*bd*	*cd*	≥ 3 items
Ann	0	1	0	0	0	0	1	1
Bob	0	0	1	0	0	1	0	1

  - $MaxMinFS(Ann) = MaxMinFS(Ann) = 1$
  - no allocation is MaxMinFS.
- what about n agents with additive preferences?

Maxmin fair share

- with two agents and additive preferences, a maxmin fair share allocation always exists
- with non-additive preferences, MaxMinFS is not guaranteed, even for two agents

	≤ 1 item	ab	ac	ad	bc	bd	cd	≥ 3 items
Ann	0	1	0	0	0	0	1	1
Bob	0	0	1	0	0	1	0	1

  - $MaxMinFS(Ann) = MaxMinFS(Ann) = 1$
  - no allocation is MaxMinFS.
- what about n agents with additive preferences?
  - existence of a MaxMinFS allocation not guaranteed to exist (Procaccia and Wang 2014)
  - but counterexamples are difficult to find.

Plan

1. Introduction: computational social choice
2. Voting rules, easy and hard
3. Multi-winner rules
4. Voting protocols
5. Fair division
6. Conclusion

Computational social choice and theoretical CS

- graph theory and more generally discrete maths
- complexity, parameterized complexity, approximation
- distributed CS, communication complexity
- online computation

Advertising

- *Handbook of Computational Social Choice* (F. Brandt, V. Conitzer, U. Endriss, J. Lang, A. Procaccia, eds.). Cambridge University Press, 2016. *Downloadable for free.*
- *Trends on Computational Computational Social Choice* (U. Endriss, ed.), 2017. *Downloadable for free.*
- An experimental voting platform: *Whale* (developed by Sylvain Bouveret, LIG): `http://whale3.noiraudes.net/`