Communication issues in Collective Decision-Making

N. Maudet, LIP6, Sorbonne University

May 2018

European Summer School in Multiagent Systems 2018, Maastricht

Example: protocols for allocating one good

Consider the following situation:

> **Problem**: *Two agents (A and B); one object to allocate. Each agent x has a valuation* $v_x \in \{0, 1, 2, 3\}$ *for the object.*
> **Goal**: *assign the object to the agent who values it the most (if same valuation, any agent is fine).*

Can we design efficient protocols to achieve this goal?

Segal. *Communication in Economic Mechanisms.* CES-2006.

Example: protocols for allocating one good

Consider the following situation:

> **Problem**: *Two agents (A and B); one object to allocate. Each agent x has a valuation $v_x \in \{0, 1, 2, 3\}$ for the object.*
> **Goal**: *assign the object to the agent who values it the most (if same valuation, any agent is fine).*

Can we design efficient protocols to achieve this goal?

Protocol π_0: "One-sided Revelation"	bits
A gives her valuation	2
B computes the allocation, and send it	1
	total $\Rightarrow$ 3

Example: protocols for allocating one good

Consider the following situation:

> **Problem**: *Two agents (A and B); one object to allocate. Each agent x has a valuation* $v_x \in \{0, 1, 2, 3\}$ *for the object.*
> **Goal**: *assign the object to the agent who values it the most (if same valuation, any agent is fine).*

Can we design efficient protocols to achieve this goal?

Protocol π_1: "English Auction"	bits
$p \leftarrow 0, X \leftarrow B$	
while continue:	
$p \leftarrow p + 1$	
ask X "continue?"	1
$X \leftarrow \overline{X}$	
allocate to $\overline{X}$	
	total $\Rightarrow$ 1, 2, or 3

Example: protocols for allocating one good

Consider the following situation:

> **Problem**: *Two agents (A and B); one object to allocate. Each agent x has a valuation $v_x \in \{0, 1, 2, 3\}$ for the object.*
> **Goal**: *assign the object to the agent who values it the most (if same valuation, any agent is fine).*

Can we design efficient protocols to achieve this goal?

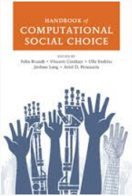
Protocol π_2: "High/Low Bisection"	bits
A says whether her valuation $\{0, 1\}$ (low) or $\{2, 3\}$ (high)	1
B computes the allocation *(if low (if $v_B = 0$ then give to A else give to B))* *(if high (if $v_B = 3$ then give to B else give to A))* and send it	1
	total $\Rightarrow$ 2

Presentation of this tutorial

The course is divided in four parts:

- Intro, background (mini break)
- Case studies I: Voting (Coffee break)
- Case studies II: Resource Allocation (mini-break)
- Case studies III: Sharing Information (World Cup)

Part of the content is based on



Presentation of this tutorial

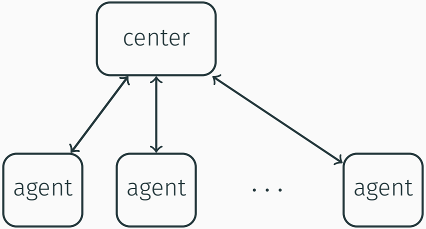
- Book available at:

  `www.cambridge.org/download_file/932961`

Brandt et al. *Handbook of Computational Social Choice*. 2016.

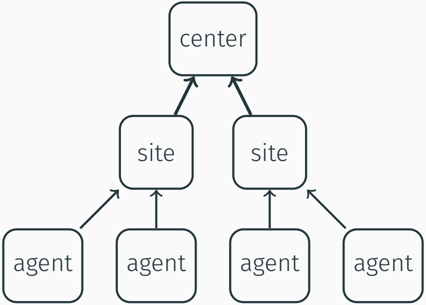
Settings and Research Questions

Different settings



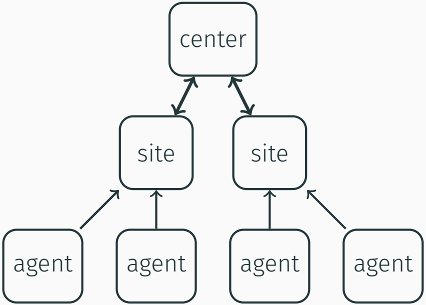
- each agent directly communicates with the center
- the center computes the outcome

Different settings



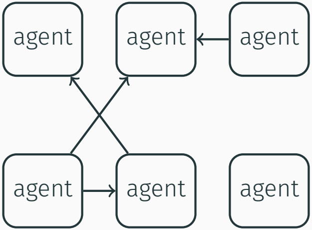
- agents communicate to some site, which only send **one** message to the center
- can be seen as the **compilation complexity**

Different settings



- agents communicate to some site, which may send message and receive messages from the center

Different settings



- each agent directly communicates with all (some of) the other agents
- the outcome is computed in a distributed manner

Objective

According to (Boutilier and Rosenschein):
Elicit partial preference profiles with **just enough** information to determine a winning outcome of **sufficiently high quality**.

- determining the optimal outcome w.r.t. the underlying (complete) preference profile
- determining the optimal outcome (i.e., true winner) with high probability
- determining an outcome that is "close to optimal" (e.g., has low max regret)
- determining an outcome that is "close to optimal" with high probability

Boutilier and Rosenschein. *Incomplete information and communication.* Handbook of Computational Social Choice. 2016.

Possible and necessary winners

Given a partial profile of preferences, an option x is

- a **possible** winner if there exists a completion of the profile such that x is the winner
- a **necessary winner** if x is the winner in any completion of the profile

☞ if an option is a necessary winner, we may safely stop elicitation

Type of messages

We usually talk about

- queries from the center
- messages among agents

Queries can be of different types, e.g:

- pairwise comparison queries
  "Do you prefer x over y?"
- value queries
  "How much do you value x over y?"
- top-k queries
  "What are your k preferred options?"
- etc.

Type of messages

We usually talk about

- queries from the center
- messages among agents

Queries can be of different types, e.g:

- pairwise comparison queries
  “Do you prefer x over y?”
- value queries
  “How much do you value x over y?”
- top-k queries
  “What are your k preferred options?”
- etc.

Communication Complexity

Communication Complexity Setting

Basic communication complexity setting

A set of n agents have to compute a function $f(x^1, \ldots, x^n)$ given that the input is distributed among the agents (x^1 privately known from agent 1, etc.)

- **protocols**: specify a communication action by the agents, given its (private) input and the bits exchanged so far
- useful **tree representation** where each node is labelled by either agent a or agent b (case of two agents), with a function specifying whether to walk left (L) or right (R) depending on its private input.

Kushilevitz & Nisan. *Communication complexity*. Cambridge U. Press, 1997.

Protocols illustrated

	y_0	y_1	y_2	y_3
x_0	0	0	0	1
x_1	0	0	0	0
x_2	0	0	0	0
x_3	1	1	1	0

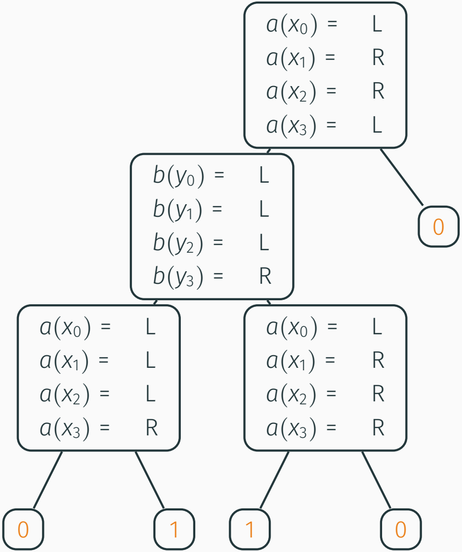
Protocols illustrated

$$\begin{aligned} a(x_0) &= \mathrm{L} \\ a(x_1) &= \mathrm{R} \\ a(x_2) &= \mathrm{R} \\ a(x_3) &= \mathrm{L} \end{aligned}$$

0

	y_0	y_1	y_2	y_3
x_0	0	0	0	1
x_1	0	0	0	0
x_2	0	0	0	0
x_3	1	1	1	0

Protocols illustrated



	y_0	y_1	y_2	y_3
x_0	0	0	0	1
x_1	0	0	0	0
x_2	0	0	0	0
x_3	1	1	1	0

Protocols illustrated

	y_0	y_1	y_2	y_3
x_0	0	0	0	1
x_1	0	0	0	0
x_2	0	0	0	0
x_3	1	1	1	0

$b(y_0) = \mathrm{L}$
$b(y_1) = \mathrm{L}$
$b(y_2) = \mathrm{L}$
$b(y_3) = \mathrm{R}$

$a(x_0) = \mathrm{L}$	$a(x_0) = \mathrm{L}$
$a(x_1) = \mathrm{L}$	$a(x_1) = \mathrm{R}$
$a(x_2) = \mathrm{L}$	$a(x_2) = \mathrm{R}$
$a(x_3) = \mathrm{R}$	$a(x_3) = \mathrm{R}$

0 1 1 0

Protocols illustrated

$b(y_0) =$ L
$b(y_1) =$ L
$b(y_2) =$ L
$b(y_3) =$ R

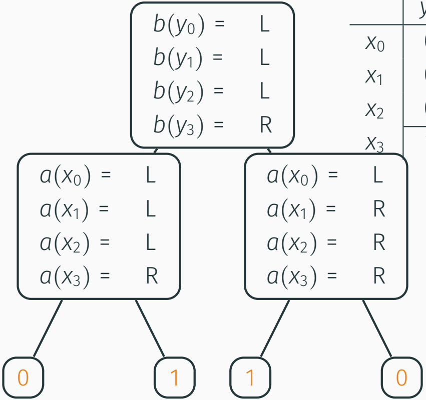
	y_0	y_1	y_2	y_3
x_0	0	0	0	1
x_1	0	0	0	0
x_2	0	0	0	0
x_3	1	1	1	0

$a(x_0) =$ L
$a(x_1) =$ L
$a(x_2) =$ L
$a(x_3) =$ R

$a(x_0) =$ L
$a(x_1) =$ R
$a(x_2) =$ R
$a(x_3) =$ R

0 1 1 0

Protocols illustrated



	y_0	y_1	y_2	y_3
x_0	0	0	0	1
x_1	0	0	0	0
x_2	0	0	0	0
x_3	1	1	1	0

Cost of protocols

- the cost of a protocol is the number of bits exchanged (in the worst case), *i.e.* the height of the tree.
  ☞ on our example, the "best" cost is the second one (cost 2 vs. 3 for the first one)
- other models (*e.g.* average) are of course possible
- the communication complexity of a function f is the minimum cost of $\mathcal{P}$ among all protocols $\mathcal{P}$ that compute f.

Protocols

Observe that the protocols, as described, in fact partition the matrix of inputs into **monochromatic** (same output) rectangles

	y_0	y_1	y_2	y_3
x_0	0	0	0	1
x_1	0	0	0	0
x_2	0	0	0	0
x_3	1	1	1	0

$\Rightarrow$ 5 monochromatic rectangles

- the number of leaves is the number of rectangles in the partition
- the cost of any protocol for a function is at least log of the minimum number of rectangles

Protocols

Observe that the protocols, as described, in fact partition the matrix of inputs into **monochromatic** (same output) rectangles

$$\begin{array}{c|ccc|c} & y_0 & y_1 & y_2 & y_3 \\ \hline x_0 & 0 & 0 & 0 & 1 \\ \cline{2-5} x_1 & 0 & 0 & 0 & 0 \\ x_2 & 0 & 0 & 0 & 0 \\ \cline{2-5} x_3 & 1 & 1 & 1 & 0 \end{array} \Rightarrow \text{5 monochromatic rectangles}$$

- the number of leaves is the number of rectangles in the partition
- the cost of any protocol for a function is at least log of the minimum number of rectangles

Back to our first example...

	0	1	2	3
0	B	B	B	B
1	A	B	B	B
2	A	A	B	B
3	A	A	A	B

	0	1	2	3
0	A	B	B	B
1	A	B	B	B
2	A	A	A	B
3	A	A	A	B

	0	1	2	3
0	A	B	B	B
1	A	B	B	B
2	A	A	A	B
3	A	A	A	B

Here it is easy to just “see” how many rectangles there are…
But in general how can we bound the number of monochromatic rectangles?

Lower bound techniques

How can we find lower bounds on the communication complexity?

- one of them is the **fooling set** technique (from TCS)
- another one is the **budget protocol** technique (from economics)

Note: These techniques actually yields lower bounds on non-deterministic protocols

The fooling set technique

- if we find a large number of inputs such that no two of them can be in the same rectangle, the number of rectangles must be large as well.
- when two input pairs (x_1, y_1) and (x_2, y_2) are in the same monochromatic rectangle, so do (x_1, y_2) and (x_2, y_1)

$$\begin{array}{c|c} 0 & ? \\ \hline ? & 0 \end{array}$$

fooling set— a collection of inputs such that no pair of them can be in the same monochromatic rectangle

The fooling set technique

Key result (Yao, 1979): CC is at least log(#fooling set)

	0	1	2	3
0	B	B	B	B
1	A	B	B	B
2	A	A	B	B
3	A	A	A	B

Note that this may sometimes provide weak bounds.

The fooling set technique

Key result (Yao, 1979): CC is at least log(#fooling set)

	0	1	2	3
0	$\boxed{\text{B}}$	B	B	B
1	A	$\boxed{\text{B}}$	B	B
2	A	A	$\boxed{\text{B}}$	B
3	A	A	A	$\boxed{\text{B}}$

Note that this may sometimes provide weak bounds.

We exhibit a fooling set of size 4. Hence CC is at least 2.

Case studies I: Voting

Settings and Research Questions

Basics of communication complexity

Case studies I: Voting

- Two examples of voting rules
- Practical Elicitation Methods
- Determining Condorcet Winner
- Distributed Monitoring of Elections
- Distributed Voting

Case studies II: Multiagent Resource Allocation

- Envy-free allocations
- Distributed Resource Allocation

Case studies III: Spreading and sharing information

- The Gossip Problem
- Russian Card Problem

Case studies I: Voting

Two examples of voting rules

Example: Borda voting

Consider the following situation:

> Problem: *There are n agents and p candidates. Each agent x has a ranking $\succ_x$ of the candidates.*
> Goal: *select the candidate who maximizes the number of points. Under the Borda scoring rule, we give p points to the first candidate, $p - 1$ for the second, and so on.*

Example: Borda voting

A first simple protocol:

- each agent reports his own vote to the center ($n \log p!$ bits)
- the center sends back the result (name of the winner) ($n \log p$ bits)

Observe that:

- this is actually a universal protocol for any voting rule!
- for specific rules we may design more clever protocols

Conitzer & Sandholm. *Communication Complexity of Common Voting Rules*. EC-05.

Example: Simple transferable vote (STV)

if there exists a candidate *c* ranked first by a majority of votes
then *c* wins
else Repeat
    let *d* be the candidate ranked first by the fewest voters;
    eliminate *d* from all ballots
    {votes for *d* transferred to the next best remaining candidate};
Until there exists a candidate *c* ranked first by a majority of votes

3	4	3	2
a	*b*	*c*	*d*
d	*d*	*d*	*c*
b	*a*	*a*	*b*
c	*c*	*b*	*a*

Example: Simple transferable vote (STV)

if there exists a candidate c ranked first by a majority of votes
then c wins
else Repeat
  let d be the candidate ranked first by the fewest voters;
  eliminate d from all ballots
  {votes for d transferred to the next best remaining candidate};
Until there exists a candidate c ranked first by a majority of votes

3	4	3	2
a	b	c	d
d	d	d	c
b	a	a	b
c	c	b	a

3	4	3	2
a	b	c	c
b	a	a	b
c	c	b	a

Example: Simple transferable vote (STV)

if there exists a candidate c ranked first by a majority of votes
then c wins
else Repeat

let d be the candidate ranked first by the fewest voters;
eliminate d from all ballots
{votes for d transferred to the next best remaining candidate};

Until there exists a candidate c ranked first by a majority of votes

3	4	3	2	3	4	3	2	7	5
a	b	c	d	a	b	c	c	b	c
d	d	d	c	b	a	a	b	c	b
b	a	a	b	c	c	b	a		
c	c	b	a						

Winner: b

- with only 3 candidates, coincides with plurality with runoff.

Example: Single Transferable Vote (STV)

A slightly more involved protocol...

step 1 voters send their most preferred candidate to the center (C) $\Rightarrow n \log p$ bits

step 2 let x be the candidate to be eliminated. All voters who had x ranked first receive a message from C asking them to send the name of their next preferred candidate. There were at most $\frac{n}{p}$ such voters $\Rightarrow \frac{n}{p} \log p$ bits

step 3 similarly with the new candidate y to be eliminated. At most $\frac{n}{p-1}$ voters voted for y $\Rightarrow \frac{n}{p-1} \log p$ bits

etc.

total $\leqslant n \log p (1 + \frac{1}{p} + \frac{1}{p-1} + \ldots + \frac{1}{2}) = \mathcal{O}(n.(\log p)^2)$

Communication complexity of voting rules

Can we apply the lower bound techniques? In our context:

- f is the voting rule
- x_i is the ballot of voter i
- we are interested in a distinguished candidate a, so f returns 1 if a wins, ad 0 otherwise

A fooling set is then a set of profiles P_i such that :

1. there exists a candidate c such that $r(P^i) = c$
2. for any pair (i, j) $(i \neq j)$, there exists $(m_1, m_2, \ldots, m_n) \in \{i, j\}^n$ such that $r(v_1^{m_1}, v_2^{m_2}, \ldots, v_n^{m_n}) \neq c$

☞ we can "mix" the profiles by picking votes either in P^i or P^j and fool the function

Conitzer & Sandholm. *Communication complexity of common voting rules.* EC-05.

Example: Lower bound for the Borda rule

π an arbitrary permutation of $\mathcal{X} \setminus \{a, b\}$ and $\overline{\pi}$ the "mirror" of π.

$$
\begin{array}{ccccccc}
1 & 2 & 3 & 4 & \cdots & n-1 & n \\
\hline
a & a & \overline{\pi} & \overline{\pi} & \cdots & a & \overline{\pi} \\
b & b & \vdots & \vdots & & b & \vdots \\
\pi & \pi & \vdots & \vdots & & \pi & \vdots \\
\vdots & \vdots & \overline{\pi} & \overline{\pi} & & \vdots & \overline{\pi} \\
\vdots & \vdots & b & b & & \vdots & a \\
\pi & \pi & a & a & \cdots & \pi & b
\end{array}
\Rightarrow (p'!)^{n'} \text{ such profiles, with } p' = p-2 \text{ and } n' = (n-2)/4
$$

1. Does a wins in any such profile?

Observe that a is 1 point ahead of any other candidate (thanks to n)

Example: Lower bound for the Borda rule

π an arbitrary permutation of $\mathcal{X}\backslash\{a, b\}$ and $\overline{\pi}$ the "mirror" of π.

$$
\begin{array}{ccccccc}
1 & 2 & 3 & 4 & \cdots & n-1 & n \\
\hline
a & a & \overline{\pi} & \overline{\pi} & \cdots & a & \overline{\pi} \\
b & b & \vdots & \vdots & & b & \vdots \\
\pi & \pi & \vdots & \vdots & & \pi & \vdots \\
\vdots & \vdots & \overline{\pi} & \overline{\pi} & & \vdots & \overline{\pi} \\
\vdots & \vdots & b & b & & \vdots & a \\
\pi & \pi & a & a & \cdots & \pi & b
\end{array}
\quad \Rightarrow (p'!)^{n'} \text{ such profiles, with } p' = p-2 \text{ and } n' = (n-2)/4
$$

2. Is it fooling?

Take two profiles P_1 and P_2, for at least one voter $i \in \{1, \ldots, n'\}$ the vote differs. Thus at least one candidate $c \notin \{a, b\}$ must be ranked higher in P_1 than P_2. Mix profiles by picking votes 4i-3 and 4i -2 from P_1 and the rest from P_2. Now c get 2 aditional points and wins.

More about this...

Service, Adams. *Communication Complexity of Approximating Voting Rules.* AAMAS-12.

Case studies I: Voting

Practical Elicitation Methods

Incremental elicitation

Suppose a partial profile has been elicited so far.

Is a a necessary winner?

Take all the “adversary” and try to their score against a.

$a \succ b \succ c \succ d$

$a \succ b$

$b \succ a, c \succ d$

$a \succ c, a \succ d, c \succ b, d \succ b$

Boutilier and Rosenschein. *Incomplete information and communication.* Handbook of Computational Social Choice. 2016.

Incremental elicitation

Suppose a partial profile has been elicited so far.
Is a a necessary winner?
Take all the “adversary” and try to their score against a.

For instance for c: maximize $s(c) - s(a)$

$a \succ b \succ c \succ d$	
$a \succ b$	$c \succ d \succ a \succ b$
$b \succ a$, $c \succ d$	$c \succ d \succ b \succ a$
$a \succ c$, $a \succ d$, $c \succ b$, $d \succ b$	$a \succ c \succ d \succ b$

Incremental elicitation

Suppose a partial profile has been elicited so far.

Is a a necessary winner?

Take all the “adversary” and try to their score against a.

For instance for c: maximize $s(c) - s(a)$

$a \succ b \succ c \succ d$		-2
$a \succ b$	$c \succ d \succ a \succ b$	2
$b \succ a, c \succ d$	$c \succ d \succ b \succ a$	3
$a \succ c, a \succ d, c \succ b, d \succ b$	$a \succ c \succ d \succ b$	-1

Incremental elicitation

Suppose a partial profile has been elicited so far.
Is a a necessary winner?
Take all the “adversary” and try to their score against a.

For instance for c:		maximize $s(c) - s(a)$
$a \succ b \succ c \succ d$		-2
$a \succ b$	$c \succ d \succ a \succ b$	2
$b \succ a, c \succ d$	$c \succ d \succ b \succ a$	3
$a \succ c, a \succ d, c \succ b, d \succ b$	$a \succ c \succ d \succ b$	-1
		2

Incremental elicitation

Suppose a partial profile has been elicited so far.
Is a a necessary winner?
Take all the "adversary" and try to their score against a.

$a \succ b \succ c \succ d$	-2
$a \succ b$	2
$b \succ a$, $c \succ d$	3
$a \succ c$, $a \succ d$, $c \succ b$, $d \succ b$	-1
	2

☞ c could win against a so a is not a necessary winner

Incremental vote elicitation: max regret computation

We can systematically compute the pairwise max regret $PMR(a, a', P)$; i.e. the worst-case (over possible completions of P) loss of selecting a instead of a'.

	a	b	c	d
a		6	8	10
b	−2	—	4	6
c	2	6		8
d	0	3	2	
MR	2	6	8	10

max regret of a is then $MR(a, P) = \max_{a'} PMR(a, a', P)$

Incremental vote elicitation

This approach has two advantages:

- by selecting the candidate minimizing *MR*
  "Close to optimal" ☞ bounded regret loss
  MR(a,P) = 0 implies *a* is (co-)necessary winner
- also provides heuristic to select queries!

Example: **current solution heuristic**

- identify a^*, the minimax regret option
- let a' be the option which maximizes regret against a^*
- pick voter with "highest potential" to decrease $PMR(a^*, a')$

Boutilier, Lu. *Robust approximation and incremental elicitation in voting protocols*. IJCAI-11.

More about this...

Other heuristics for incremental voting elicitation (e.g. top-k queries):

Kalech et al. *Practical Voting Rules with Partial Information*. JAAMAS-11.

Naamani-Dery et al. *Reducing preference elicitation in group decision making*. Exp. Syst. Appl. 2016.

Case studies I: Voting

Determining Condorcet Winner

Condorcet winner: query complexity

Consider the following situation:

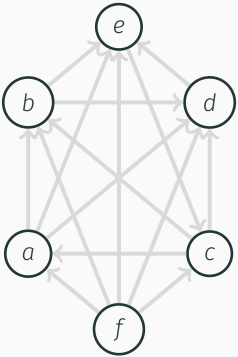
> **Problem**: *n agents with preferences over m options expressed as linear orders, inducing a majority graph.*
> **Goal**: *determine whether one option beats all the other ones in pairwise comparison*

Example: b is a Condorcet winner

$$\begin{array}{ll} 1: & a \succ b \succ c \\ 2: & b \succ c \succ a \\ 3: & c \succ b \succ a \end{array}$$

How many edges of the majority graph do we need to query to answer this question?

Condorcet winner: query complexity

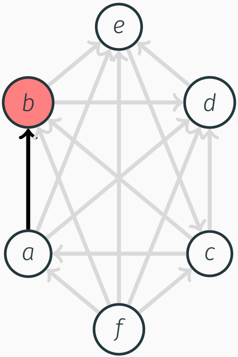


Condorcet winner: query complexity

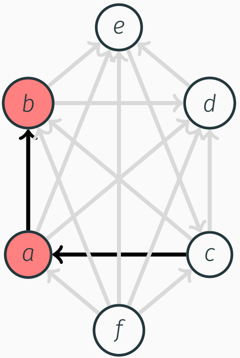
Analyzed under the query complexity model.

- A (di)graph is unknown to start with, and want to check whether some property holds in the graph by probing the fewest possible edges
- Of course $p(p-1)/2$ are sufficient. Can we do better?
- A property is evasive if all edges must be queried (in the worst case)

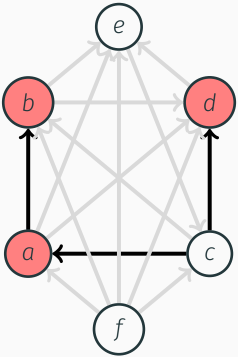
Condorcet winner: query complexity



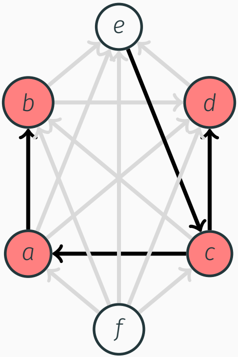
Condorcet winner: query complexity



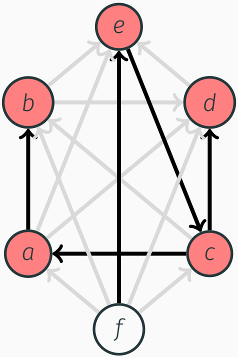
Condorcet winner: query complexity



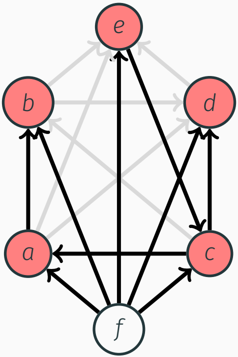
Condorcet winner: query complexity



Condorcet winner: query complexity



Condorcet winner: query complexity



Condorcet winner: query complexity

- start with an arbitrary query between two candidates
- mark the looser as discarded
- repeat $p-2$ times:
  - take the winner of the previous query, query against a non-discarded candidate, mark the loser as discarded
  - note: each pairwise comparison discards exactly 1 new candidate
- after $p-1$ questions we either know that there is no Condorcet winner, or there is a unique potential Condorcet winner
- then we need to check that this candidate beats all the remaining $p-2$ ones
- this protocol requires $2p-3$ queries

Condorcet winner: query complexity

- start with an arbitrary query between two candidates
- mark the looser as discarded
- repeat $p - 2$ times:
  - take the winner of the previous query, query against a non-discarded candidate, mark the loser as discarded
  - note: each pairwise comparison discards exactly 1 new candidate
- after $p - 1$ questions we either know that there is no Condorcet winner, or there is a unique potential Condorcet winner
- then we need to check that this candidate beats all the remaining $p - 2$ ones
- this protocol requires $2p - 3$ queries

Can we do better than this?

Condorcet winner: query complexity

1. build an almost complete binary tree, where leaves are labelled as candidates
2. repeat until the root is labelled
   - query about two leaves
   - label the father with the winner
   - cut the children
3. query about the candidate labelling the root (r) against all candidates not

How many queries?

Condorcet winner: query complexity

1. build an almost complete **binary tree**, where leaves are labelled as candidates
2. repeat until the root is labelled
   - query about two leaves
   - label the father with the winner
   - cut the children
3. query about the candidate labelling the root (r) against all candidates not

How many queries?

Step 2 takes $p - 1$ queries.

Furthermore, r must have beaten at least $\lfloor \log_2(p) \rfloor$ during step 2.

Therefore there are $p - 1 - \lfloor \log_2(p) \rfloor$ during step 3.

The protocol requires at most $2p - \log_2(p) - 2$ queries.

More about this...

Balasubramanian et al.. *Finding scores in tournaments.* J. of Algorithms, 1997.

Procaccia. *A note on the query complexity of the Condorcet winner problem.* Information Processing Letters 108(6), 2008.

Dey. *Query Complexity of Tournament Solutions.* ArXiv, 2018.

Case studies I: Voting

Distributed Monitoring of Elections

Distributed Monitoring of Elections

Consider the following situation:

> Problem: *k sites. n agents arriving continuously (as a stream) and casting votes to a site; each agent x has linear preferences over m options.*
> Goal: *Maintain the winning outcome for some voting rule*

Can we minimize communication between the k sites and the center?

Distributed Monitoring of Elections

"Close to optimal" ☞ ϵ-winner: in an election with n voters, a candidate who may become a winner by adding at most ϵn voters

Various techniques deployed: one is to design protocols based on **checkpoints**

Idea: only update the winner when necessary (ie. no longer possible to guarantee that announced winner is ϵ-winner). Requires to count the number of voters arriving to determine these checkpoints.

Filtser and Talmon. *Distributed Monitoring of election winners.* ArXiv-2018.

Consider then the **count tracking** problem:

- there are k sites, which make (non-overlapping) observations (in our case: sites receives votes)
- we wish to trigger an action when the overall number of votes reaches a threshold (S)

Consider then the count tracking problem:

- there are k sites, which make (non-overlapping) observations (in our case: sites receives votes)
- we wish to trigger an action when the overall number of votes reaches a threshold (S)

Naive solution each local site sends a new message each time a new voter appears

A simple protocol for count tracking

Idea it requires a number of observations on each local center before being required to trigger More specifically, at least one of the local center must have made S/k observations

A simple protocol for count tracking

Idea it requires a number of observations on each local center before being required to trigger More specifically, at least one of the local center must have made S/k observations

Algorithm 2: Count tracking: basic version

Each agent starts with an individual threshold $t \leftarrow S/k$
repeat
    repeat
        At each new obervation by x, $n_x \leftarrow n_x + 1$
    until *an agent x has made t observations*;
    agent x sends a message to the center
    the center collects the n_x of each agent
    $S \leftarrow S - \sum n_x$ (update # missing observations)
    $t \leftarrow S/k$ (update threshold)
until S=k;
repeat
    send any observation to the center $S \leftarrow S - 1$
until S=0;

More about this...

In the related setting of compilation complexity the sites may only send one single message to the center.

Chevaleyre et al. *Compiling the votes of a subelectorate.* IJCAI-09.

Xia, Conitzer. *Compilation complexity of common voting rules.* AAAI-10.

Case studies I: Voting

Distributed Voting

Envy-free allocation of items

Consider the following situation:

> Problem: *n agents; each agent x has linear preferences over m options. No center available.*
> Goal: *Decide the winning outcome of a scoring-based voting rule*

☞ boils down to compute sums in a distributed way

The Push-Sum protocol

At each turn t, each agent maintains

- a sum $s_{t,i}$, initialized to $s_{0,i} \leftarrow x_i$, and
- a weight $w_{t,i}$, intialized to $w_{0,i} \leftarrow 1$.

Now at each turn t:

1. let $\{(\hat{s}_r, \hat{w}_r)\}$ the set of messages received by i during the previous turn
2. let $s_{t,i} \leftarrow \sum \hat{s}_r$, et $w_{t,i} \leftarrow \sum \hat{w}_r$
3. agent i picks uniformly at random one of the other agents (or his neighbours) $f_t(i)$
4. agent i sends message $(\frac{1}{2}s_{t,i}, \frac{1}{2}w_{t,i})$ to $f_t(i)$ and to himself
5. ratio $\frac{s_{t,i}}{w_{t,i}}$ is the estimate of the mean at time t

The Push-Sum protocol

Convergence guarantees are very good:

> Push-sum converges to a "very close" estimate of the mean in $\mathcal{O}(\log n)$ turns. As each turn requires n messages, this gives $\mathcal{O}(n \log n)$ messages overall.

Note that the protocol is presented in a synchronous way, but can easily be adapted to an asynchronous setting (in that case convergence speed in only conjectured by the authors though).

Kempe, Dobra, Gehrke. *Gossip-Based Computation of Aggregate Information*. FOCS-03.

Case studies II: Multiagent Resource Allocation

Case studies II: Multiagent Resource Allocation

Envy-free allocations

Consider the following situation:

> Problem: *n agents; several object to allocate. Each agent x has a valuation v_x over bundles of items*
> Goal: *assign the objects to the agents so that no agent envies the bundle of the other agents*

	o_1	o_2	o_3	o_4	o_5	o_6
1 :	5	2	1	3	7	4
2 :	2	2	4	9	4	4
3 :	2	2	4	9	4	4

Can you find an envy-free allocation?

Envy-free allocation of items

- difficult problem to decide whether en EF allocation exists (as soon as required to allocate all objects), even in very restricted (e.g. additive) domains
- for general valuations, can be shown to require an exponential number of queries in the worst case, assuming **bundle value queries**

Lipton et al. *On approximately fair allocations of divisible goods.* EC-04.

Envy-free allocation of items

- difficult problem to decide whether en EF allocation exists (as soon as required to allocate all objects), even in very restricted (e.g. additive) domains
- for general valuations, can be shown to require an exponential number of queries in the worst case, assuming **bundle value queries**

Lipton et al. *On approximately fair allocations of divisible goods.* EC-04.

Can we do better by only requiring close to optimal?

Envy-free allocation of items

- difficult problem to decide whether en EF allocation exists (as soon as required to allocate all objects), even in very restricted (e.g. additive) domains
- for general valuations, can be shown to require an exponential number of queries in the worst case, assuming **bundle value queries**

Lipton et al. *On approximately fair allocations of divisible goods.* EC-04.

Can we do better by only requiring close to optimal?

Close to optimal ☞ envy "up to one good"

$$\forall i, j \in N \; \exists r \in \pi(j) : u_i(\pi(i)) \geqslant u_i(\pi(j) \setminus \{r\})$$

Lipton *et al.*

We first present informally the approach, based on a simple sequential allocation of resources.
For each resource r_k to be allocated:

- build the envy graph $G = (\mathcal{N}, E)$, where $(i, j) \in E \times E$ if agent i envies agent j
- while the graph has cycles, pick one $C = (c_1, c_2, \ldots c_q)$, and reallocates the bundle of c_i to c_{i-1} (and of c_1 to c_q).
- allocate r_k to an agent that no one envies.

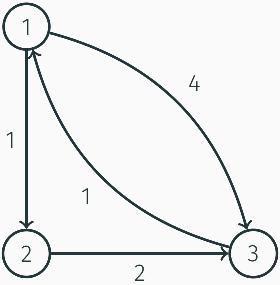
Lipton et al. *On approximately fair allocations of divisible goods.* EC-04.

	r_0	r_1	r_2	r_3	r_4	r_5
agent 1	1	2	5	3	7	2
agent 2	2	6	8	1	1	2
agent 3	5	4	4	3	2	2

	r_0	r_1	r_2	r_3	r_4	r_5
agent 1	1	2	5	3	7	2
agent 2	2	6	8	1	1	2
agent 3	5	4	4	3	2	2

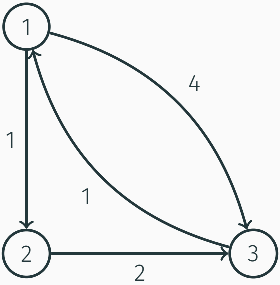
No object is allocated yet.

Lipton *et al.*



	r_0	r_1	r_2	r_3	r_4	r_5
agent 1	$\boxed{1}$	2	5	3	7	2
agent 2	2	$\boxed{6}$	8	1	1	2
agent 3	5	4	$\boxed{4}$	3	2	2

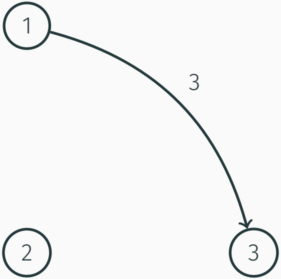
Lipton *et al.*



	r_0	r_1	r_2	r_3	r_4	r_5
agent 1	$\boxed{1}$	2	5	3	7	2
agent 2	2	$\boxed{6}$	8	1	1	2
agent 3	5	4	$\boxed{4}$	3	2	2

There are two cycles: (1,3) or (1,2,3)

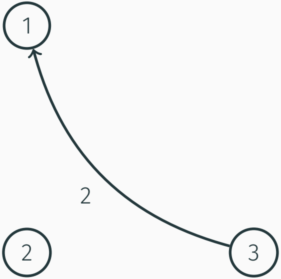
Lipton *et al.*



	r_0	r_1	r_2	r_3	r_4	r_5
agent 1	1	2	5	3	7	2
agent 2	2	6	8	1	1	2
agent 3	5	4	4	3	2	2

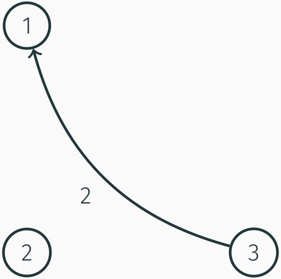
Suppose we chose cycle (1,2,3). After a single rotation, agent 1 and agent 2 are not envied any longer.

Lipton *et al.*



	r_0	r_1	r_2	r_3	r_4	r_5
agent 1	1	2	5	3	7	2
agent 2	2	6	8	1	1	2
agent 3	5	4	4	3	2	2

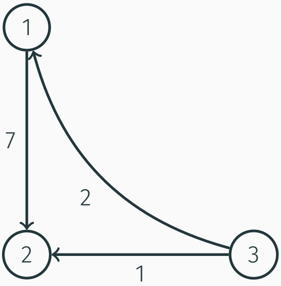
Lipton *et al.*



	r_0	r_1	r_2	r_3	r_4	r_5
agent 1	1	$\boxed{2}$	5	$\boxed{3}$	7	2
agent 2	2	6	$\boxed{8}$	1	1	2
agent 3	$\boxed{5}$	4	4	3	2	2

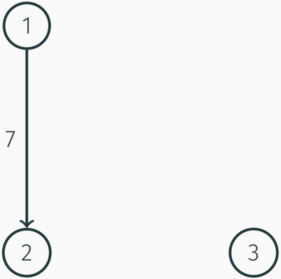
We can give r_3 to agent 1. There are no cycle, agent 2 and agent 3 are not envied.

Lipton *et al.*



	r_0	r_1	r_2	r_3	r_4	r_5
agent 1	1	$\boxed{2}$	5	$\boxed{3}$	7	2
agent 2	2	6	$\boxed{8}$	1	$\boxed{1}$	2
agent 3	$\boxed{5}$	4	4	3	2	2

We can give r_4 to agent 2. There are no cycles but only agent 3 is not envied.



	r_0	r_1	r_2	r_3	r_4	r_5
agent 1	1	$\boxed{2}$	5	$\boxed{3}$	7	2
agent 2	2	6	$\boxed{8}$	1	$\boxed{1}$	2
agent 3	$\boxed{5}$	4	4	3	2	$\boxed{2}$

We finally give r_5 to agent 3. The final allocation is not envy-free, as agent 1 envies agent 2.

Lipton *et al.*: analysis

Cycle reallocation step: $C = (c_1, c_2, \ldots, c_q)$

☞ Envy must have decreased.

- any agent in the cycle has increased its utility.
- bundles are unaffected

Lipton *et al.*: analysis

Cycle reallocation step: $C = (c_1, c_2, \ldots, c_q)$

☞ Envy must have decreased.

- any agent in the cycle has increased its utility.
- bundles are unaffected

☞ The number of edges in the envy graph has decreased.

- edges between agents $\notin C$ are not affected
- edges from agents $\notin C$ to C now point to previous agent in C
- edges from agents $\in C$ to agents $\notin C$ may only decrease
- (original) edges between agents $\in C$ are deleted

Lipton et al. *On approximately fair allocations of divisible goods.* EC-04.

Lipton *et al.*: envy is bounded

Let α be the max value that any agent gives to a good.

- ☞ The max envy between pair of agents is bounded by α
- ☞ The protocol guarantees envy up to one good

Lipton *et al.*: envy is bounded

Let α be the max value that any agent gives to a good.

☞ The max envy between pair of agents is bounded by α
☞ The protocol guarantees envy up to one good

Base case:

A_0: allocate first resource randomly. Clearly $e(A_0) \leqslant \alpha$.

Induction step:

Suppose A with $\{r_1, \ldots, r_k\}$ allocated, and $e(A) \leqslant \alpha$.
By repeatedly applying cycle reallocation in the envy graph, we must get an acyclic graph.
Hence at least an agent j is not envied: she gets r_{k+1}.
Envy among agents $\neq j$ is not affected.
Envy of agents $i \neq j$ towards j is $\leqslant \alpha$, since j was not envied.

Lipton et al. *On approximately fair allocations of divisible goods.* EC-04.

Lipton *et al.*: analysis of the protocol

The communication requirement of the protocol is

- for each agent, to indicate who she envies (n^2),
- his may be repeated at each edge removal, and there may be n^2 edges at most to remove,
- this occurs for each resource allocation

giving overall $\mathcal{O}(mn^4)$ bits.

☞ observe that the protocol never requires agents to communicate utilities

Case studies II: Multiagent Resource Allocation

Distributed Resource Allocation

Distributed Resource Allocation

Consider the following situation:

> **Problem**: *n agents; m objects to allocate.*
> *Each agent x has valuation v_x over bundles of objects.*
> *Each agent initially holds a bundle of objects.*
> **Goal**: *reach an efficient/fair allocation by means of local deals*

Sandholm. *Contract types for satisficing task allocation.* IEEE Symposium-1998.

Endriss et al.. *Negotiating socially optimal allocation of resources.* JAIR-2006.

Contract-Based Negotiation

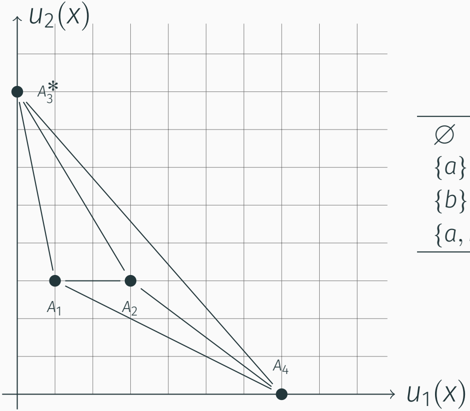
Some known results:

- a deal is IR (with money) iff it increases utilitarian social welfare (i.e, sum of utilities, thus generates a **surplus**).
- allows to show that **any** sequence of IR deals converges to an allocation maximizing utilitarian social welfare
- however, may require **very complex** deals to be implemented during the negotiation (in fact, for any conceivable deal we may construct a scenario requiring exactly that deal).

Sandholm. *Contract types for satisficing task allocation.* IEEE Symposium-1998.

Endriss et al.. *Negotiating socially optimal allocation of resources.* JAIR-2006.

Contract-Based Negotiation

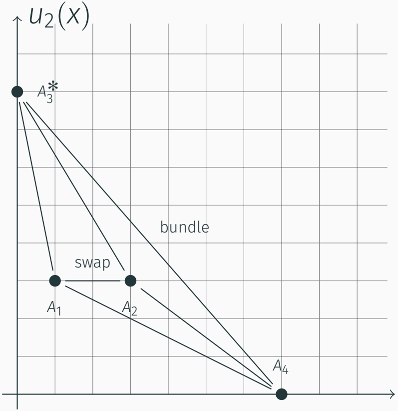


	u_1	u_2
$\emptyset$	0	0
$\{a\}$	1	3
$\{b\}$	3	3
$\{a, b\}$	7	8

with

$A_1 = \langle a, b \rangle$
$A_2 = \langle b, a \rangle$
$A_3 = \langle ab, \emptyset \rangle$
$A_4 = \langle \emptyset, ab \rangle$

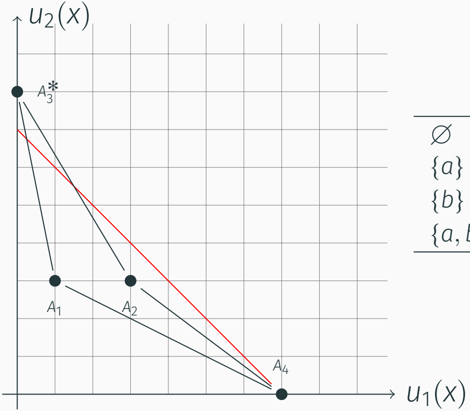
Contract-Based Negotiation



	u_1	u_2
$\emptyset$	0	0
$\{a\}$	1	3
$\{b\}$	3	3
$\{a, b\}$	7	8

with
$A_1 = \langle a, b \rangle$
$A_2 = \langle b, a \rangle$
$A_3 = \langle ab, \emptyset \rangle$
$A_4 = \langle \emptyset, ab \rangle$

Contract-Based Negotiation



	u_1	u_2
$\emptyset$	0	0
$\{a\}$	1	3
$\{b\}$	3	3
$\{a, b\}$	7	8

with

$A_1 = \langle a, b \rangle$
$A_2 = \langle b, a \rangle$
$A_3 = \langle ab, \emptyset \rangle$
$A_4 = \langle \emptyset, ab \rangle$

Length of sequences in distributed resource allocation

We interpret here communication complexity in terms of the number of deals required to reach an (efficient) outcome.

- there are n^m allocations, as it is possible to construct scenarios going through all the allocations, it is a tight upper bound

Length of sequences in distributed resource allocation

We interpret here communication complexity in terms of the number of deals required to reach an (efficient) outcome.

- there are n^m allocations, as it is possible to construct scenarios going through all the allocations, it is a tight upper bound
- without any restriction on the deal complexity, a path of length 1 is always possible

Length of sequences in distributed resource allocation

We interpret here **communication complexity** in terms of the **number of deals** required to reach an (efficient) outcome.

- there are n^m allocations, as it is possible to construct scenarios going through all the allocations, it is a tight upper bound
- without any restriction on the deal complexity, a path of length 1 is always possible
- with **1-deals** in **additive domains**, the path length is between m and $m \times (n-1)$

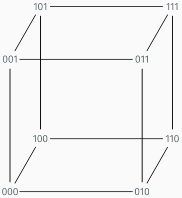
Endriss & Maudet. *Communication Complexity of Multilateral Trading*. JAAMAS05.

Length of sequences with 1-deals

Now consider 1-deals without restriction on utility functions.
Can we find lower bounds on the path length?

Length of sequences with 1-deals

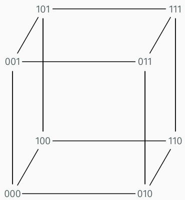
Now consider 1-deals without restriction on utility functions. Can we find lower bounds on the path length?



Related to the problem of finding a sequence of moves in an hypercube such, for any state s_i, any other state $s_{\geqslant i+2}$ in this sequence has a Hamming distance $\geqslant 2$ with s_i (no "shortcuts")

Length of sequences with 1-deals

Now consider 1-deals without restriction on utility functions. Can we find lower bounds on the path length?

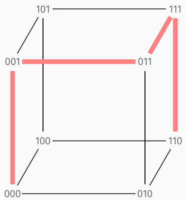


Related to the problem of finding a sequence of moves in an hypercube such, for any state s_i, any other state $s_{\geqslant i+2}$ in this sequence has a Hamming distance $\geqslant 2$ with s_i (no “shortcuts”)

- Corresponds to the snake in the box problem, very well studied. Their maximal length is $\mathcal{O}(2^m)$ (precisely, $\frac{77}{256}2^m - 2$)

Length of sequences with 1-deals

Now consider 1-deals without restriction on utility functions. Can we find lower bounds on the path length?



Related to the problem of finding a sequence of moves in an hypercube such, for any state s_i, any other state $s_{\geqslant i+2}$ in this sequence has a Hamming distance $\geqslant 2$ with s_i (no "shortcuts")

- Corresponds to the snake in the box problem, very well studied. Their maximal length is $\mathcal{O}(2^m)$ (precisely, $\frac{77}{256}2^m - 2$)

Length of sequences with 1-deals

Can we construct a negotiation instance like these snake-in-the-box sequences?

Let $\alpha = \alpha_1, \alpha_2, \ldots \alpha_n$ be such a sequence.

Now consider two agents, and fix their utilities such that

$$u_1(B) + u_2(\overline{B}) = k \text{ if } B = \alpha_k \text{ (and 0 otherwise)}$$

Hence α is the unique sequence of 1-deals from α_1 to α_n, because:

- no shortcut from α_i to $\alpha_{j>i+1}$
- in $A_i = \alpha_i$, no other allocation is IR except $A_{i+1} = \alpha_{i+1}$

Length of sequences with 1-deals

Hence α is the unique sequence of 1-deals from α_1 to α_n, because:

- no shortcut from α_i to $\alpha_{j>i+1}$
- in $A_i = \alpha_i$, no other allocation is IR except $A_{i+1} = \alpha_{i+1}$

Example $m = 4$, and $\alpha = 0000|1000|1010|1110|0110|0111|0101|1101$

	B	$\overline{B}$	u_1	u_2
α_1	0000	1111	1	0
	0001	1110	0	0
	0010	1101	0	0
	0011	1100	0	0
	0100	1011	0	0
α_7	0101	1010	7	0
α_5	0110	1001	5	0
α_6	0111	1000	6	0
α_2	1000	0111	2	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Case studies III: Spreading and sharing information

Case studies III: Spreading and sharing information

The Gossip Problem

The gossip problem

Consider the following situation:

> Problem: *n agents; each agent x holding a secret X. When two agents communicate, they share their secrets.*
> Goal: *reach a state where all the agents know all the secrets*

How many exchanges are needed to reach the goal?

The gossip problem

Consider the following situation:

> Problem: *n agents; each agent x holding a secret X. When two agents communicate, they share their secrets.*
> Goal: *reach a state where all the agents know all the secrets*

How many exchanges are needed to reach the goal?

Start with 4 agents...

The gossip problem

General case: the busy body solution

- all the agents speak to some designated agent n-1
- who becomes expert and then communicate back to all the agents (except the last one) n-2
- hence summing up to $2n - 3$

The gossip problem

General case: the busy body solution

- all the agents speak to some designated agent n-1
- who becomes expert and then communicate back to all the agents (except the last one) n-2
- hence summing up to $2n - 3$

Can we do better?

General case: the four people solution

- each agent communicates to one of 4 people n-4
- the four people exchange their secrets 4
- they communicate back to the other agents n-4
- hence summing up to $2n - 4$

The gossip problem

But this assumes of course a centralized orchestration.

What about distributed gossip protocols?

Algorithm 3: ANY

repeat
  select to agents who did not call each other
  let a call b
until *all agents are experts*;

Apt et al. *Epistemic protocols for distributed gossiping*. TARK-05.

van Ditmarsch et al. *Reachability and expectation in gossiping*. PRIMA-17.

The gossip problem

But this assumes of course a centralized orchestration.

What about distributed gossip protocols?

Algorithm 4: CO

repeat
  select two agents who did not call each other
  let a call b
until *all agents are experts*;

Apt et al. *Epistemic protocols for distributed gossiping*. TARK-05.

van Ditmarsch et al. *Reachability and expectation in gossiping*. PRIMA-17.

But this assumes of course a centralized orchestration.

What about **distributed gossip protocols**?

Algorithm 5: LNS

repeat
  select two agents a such that a does not know b's secret
  let a call b
until *all agents are experts*;

Apt et al. *Epistemic protocols for distributed gossiping*. TARK-05.

van Ditmarsch et al. *Reachability and expectation in gossiping*. PRIMA-17.

Case studies III: Spreading and sharing information

Russian Card Problem

Consider the following situation:

> Problem: *In the original Russian Card Problem, there are 7 cards* $\{0, 1, \ldots 6\}$. *A and B receive (privately) 3 cards each, and C receives a single card.*
> Goal: *A and B communicate with the aim that they know mutually their hand, while C doesn't know anything*

van Ditmarsch. *The Russian cards problem. The dynamics of knowledge.* Studia Logica, 2003.

Assume messages to be of the form:

“I hold H or H' or …”

where each H is a hand of three cards.

A message from A is said to be :

- safe if, after uttering it, C doesn't know anything (doesn't who holds any card)
- informative for B if, upon receiving the message, B knows the hand of A

The Russian Card Problem: a solution

Assume the true situation to be A:012, B:345, C: 6

Possible worlds for *B*: (012),(016),(026),(126)

Possible worlds for *C*?

The Russian Card Problem: a solution

Assume the true situation to be A:012, B:345, C: 6

Possible worlds for *B*: (012),(016),(026),(126)

Possible worlds for *C*?

Now A sends the message: A: $012 \vee 034 \vee 056 \vee 135 \vee 246$

After the message:

- (012) is the only possible world for *A*
- check that *C* can not locate any card

Can we reach the goal with a shorter message?

☞ Each card must appear at least once in a safe message

The Russian Card Problem: bounding the size of messages

Can we reach the goal with a shorter message?

☞ Each card must appear at least once in a safe message

if x doesn't appear in the message, A doesn't hold x. But A may believe that C doesn't hold x, in which case C would know that B holds : non safe.

The Russian Card Problem: bounding the size of messages

Can we reach the goal with a shorter message?

☞ Each card must appear at least **twice** in a safe message

Can we reach the goal with a shorter message?

☞ Each card must appear at least **twice** in a safe message

Suppose x appears only once. Let xyz the hand where x appears. Suppose A doesn't hold y or z, eg. y. In that case C could hold y, and thus eliminate xyz, and thus that B must have x, etc.

The Russian Card Problem: bounding the size of messages

Can we reach the goal with a shorter message?

☞ 5 hands are needed in any safe message

Can we reach the goal with a shorter message?

☞ 5 hands are needed in any safe message
each card must appear twice, hence 14 occurrences of cards must appear, but with 4 hands we would only get 12 occurrences

The Generalized Russian Card Problem

In the (a, b, c) Card Problem, A receives a cards, B receives b cards, and C receives c cards. Denote by H_X the hand of X.

The message is informative for B iff there are no two hands of A, H_A, H'_A such that $|H_A \cap H'_A| \geqslant a - c$

The Generalized Russian Card Problem

In the (a, b, c) Card Problem, A receives a cards, B receives b cards, and C receives c cards. Denote by H_X the hand of X.

The message is informative for B iff there are no two hands of A, H_A, H'_A such that $|H_A \cap H'_A| \geqslant a - c$

Proof ($\Leftarrow$): Suppose for contradiction H_A and H'_A such that $|H_A \cap H'_A| \geqslant a - c$. Then

$$\begin{aligned} |H_A \cup H'_A| &\leqslant 2a - (a - c) \\ &\leqslant a + c \\ &\leqslant n - b \end{aligned}$$

But then there must exists H_B such that $H_B \cap (H_A \cup H'_A) = \varnothing$! Hence B would hesitate between H_A and H'_A.

The Generalized Russian Card Problem

When $a = c + 1$, no protocol can succeed in two messages.

The Generalized Russian Card Problem

When $a = c + 1$, no protocol can succeed in two messages.

Proof:

For a protocol to proceed in two messages, the first message must be informative (and safe)

- to be informative, all the hands must be disjoint (by the previous result)
- to be safe, all the cards must appear at least twice

Swanson, Stinson. *Combinatorial solutions providing improved security for the generalized Russian cards problem.* Designs, Codes and Cryptography, 2002.

Thank you!