Algorithmic mechanism design

Fedor Sandomirskiy

October 11, 2017

Higher School of Economics, St.Petersburg
e-mail: fsandomirskiy@hse.ru

Computer boom and mechanism design

Rapid development of computers at the end of 90ies $\Rightarrow$

- an opportunity to implement theoretically developed mechanisms
    - complex auctions, large centralized markets (school choice, organ transplants)
- need for new mechanisms
    - sponsored search auctions, peer-review in MOOCs, online-markets, ranking systems, procedures for sharing computation resources etc

New methodology

The mechanism design became more practically-oriented. The main new features:

- focus is on positive results. Non-existence of an ideal mechanism say nothing for practice.
- importance of algorithmic and complexity issues: How hard it is for agents to communicate the relevant information to a mechanism? How hard is to compute the outcome?

Algorithmic questions are studied by Algorithmic Mechanism Design, Algorithmic Game Theory, and Computational Social Choice

Outline:

- Combinatorial auctions: the role of complexity
- Fair division of indivisible goods: how to overcome negative results?

Combinatorial auctions: the role of complexity

Combinatorial Auctions

CA = Auction with multiple goods

- a set A, $|A| = m$, of different indivisible goods is to be allocated via auction to the set N of agents
- Agents are interested in bundles of goods. Valuation of agent i : $v_i : 2^A \to \mathbb{R}_+$

Question: How to organize such an auction?

Combinatorial Auctions

CA = Auction with multiple goods

- a set A, $|A| = m$, of different indivisible goods is to be allocated via auction to the set N of agents
- Agents are interested in bundles of goods. Valuation of agent i : $v_i : 2^A \to \mathbb{R}_+$

Question: How to organize such an auction?

Combinatorial Auctions

CA = Auction with multiple goods

- a set A, $|A| = m$, of different indivisible goods is to be allocated via auction to the set N of agents
- Agents are interested in bundles of goods. Valuation of agent i : $v_i : 2^A \to \mathbb{R}_+$

Question: How to organize such an auction?

Bad idea: Run m independent auctions (e.g., first price) for every item

Combinatorial Auctions

CA = Auction with multiple goods

- a set A, $|A| = m$, of different indivisible goods is to be allocated via auction to the set N of agents
- Agents are interested in bundles of goods. Valuation of agent i : $v_i : 2^A \to \mathbb{R}_+$

Question: How to organize such an auction?

Bad idea: Run m independent auctions (e.g., first price) for every item

This will not work well, when agents' valuations express complementarity ($v_i(A_1 \cup A_2) > v_i(A_1) + v_i(A_2)$ for some disjoint $A_1, A_2 \subset A$).

Combinatorial Auctions

CA = Auction with multiple goods

- a set A, $|A| = m$, of different indivisible goods is to be allocated via auction to the set N of agents
- Agents are interested in bundles of goods. Valuation of agent i : $v_i : 2^A \to \mathbb{R}_+$

Question: How to organize such an auction?

Bad idea: Run m independent auctions (e.g., first price) for every item

This will not work well, when agents' valuations express complementarity ($v_i(A_1 \cup A_2) > v_i(A_1) + v_i(A_2)$ for some disjoint $A_1, A_2 \subset A$).

Example: $A = \{$red sofa, red chair, green sofa, green chair$\}$
If A' contains $\{rs, rc\}$ or $\{gs, gc\}$, then $v_{Alice}(A') = 100$, otherwise 0.

In independent auctions Alice may end up with a useless bundle but pay for it (the so called exposure problem)

Combinatorial Auctions

CA = Auction with multiple goods

- a set A, $|A| = m$, of different indivisible goods is to be allocated via auction to the set N of agents
- Agents are interested in bundles of goods. Valuation of agent i : $v_i : 2^A \to \mathbb{R}_+$

Question: How to organize such an auction?

Bad idea: Run m independent auctions (e.g., first price) for every item

This will not work well, when agents' valuations express complementarity ($v_i(A_1 \cup A_2) > v_i(A_1) + v_i(A_2)$ for some disjoint $A_1, A_2 \subset A$).

Example: $A = \{$red sofa, red chair, green sofa, green chair$\}$
If A' contains $\{rs, rc\}$ or $\{gs, gc\}$, then $v_{Alice}(A') = 100$, otherwise 0.

In independent auctions Alice may end up with a useless bundle but pay for it (the so called exposure problem)

Corollary: independent auctions may produce unpredicted and inefficient outcomes. Agents take these risks into account and post lower bids decreasing the revenue of the seller.

Famous real-world examples

- GSM spectrum auctions (beginning of 00s; many countries except Russia :-( ):
  - $A \ni \{$"1100 MHz over North-west region"$\}$, usually $|A| > 1000$
  - bidders = telecommunication companies
  - volume: hundreds of billions of dollars
  - Different frequencies at the same region are substitutes; different regions are complements
- Airport landing slots:
  - $A =$ opportunities to depart or land at a particular airport in a given interval of time
  - bidders = airlines
  - departure opportunity without corresponding landing one has no value

Famous real-world examples

- GSM spectrum auctions (beginning of 00s; many countries except Russia :-( ):
  - $A \ni \{$"1100 MHz over North-west region"$\}$, usually $|A| > 1000$
  - bidders = telecommunication companies
  - volume: hundreds of billions of dollars
  - Different frequencies at the same region are substitutes; different regions are complements
- Airport landing slots:
  - $A =$ opportunities to depart or land at a particular airport in a given interval of time
  - bidders = airlines
  - departure opportunity without corresponding landing one has no value

The main two approaches to combinatorial auctions:

Simultaneous ascending auctions

- ascending auctions for every good are conducted at the same time
- agents learn some information about others' preferences looking at their previous bidding behavior $\Rightarrow$ may estimate their chances of getting the desired bundle and thus adapt the bidding strategy

The main two approaches to combinatorial auctions:

Simultaneous ascending auctions

- ascending auctions for every good are conducted at the same time
- agents learn some information about others' preferences looking at their previous bidding behavior $\Rightarrow$ may estimate their chances of getting the desired bundle and thus adapt the bidding strategy

Pros:

- Partial elimination of exposure problem

The main two approaches to combinatorial auctions:

Simultaneous ascending auctions

- ascending auctions for every good are conducted at the same time
- agents learn some information about others' preferences looking at their previous bidding behavior ⇒ may estimate their chances of getting the desired bundle and thus adapt the bidding strategy

Pros:

- Partial elimination of exposure problem

Cons:

- An ad-hoc approach with many small details to be fixed. Example: incentives to wait until other agents reveal their preferences ⇒ necessity of various activity rules which inspire active bidding.
- Efficiency of the outcome is not guaranteed
- Inspire collusion and decrease competition. If goods are "almost substitutes", it is easy for agents to signal with their first bids what are the bundles they will compete for ⇒ easy to divide the market and thus pay less (this is why spectrum auction in Switzerland failed).

The main two approaches to combinatorial auctions:

Simultaneous ascending auctions

- ascending auctions for every good are conducted at the same time
- agents learn some information about others' preferences looking at their previous bidding behavior ⇒ may estimate their chances of getting the desired bundle and thus adapt the bidding strategy

Pros:

- Partial elimination of exposure problem

Cons:

- An ad-hoc approach with many small details to be fixed. Example: incentives to wait until other agents reveal their preferences ⇒ necessity of various activity rules which inspire active bidding.
- Efficiency of the outcome is not guaranteed
- Inspire collusion and decrease competition. If goods are "almost substitutes", it is easy for agents to signal with their first bids what are the bundles they will compete for ⇒ easy to divide the market and thus pay less (this is why spectrum auction in Switzerland failed).

The main two approaches to combinatorial auctions:

Simultaneous ascending auctions

- ascending auctions for every good are conducted at the same time
- agents learn some information about others' preferences looking at their previous bidding behavior ⇒ may estimate their chances of getting the desired bundle and thus adapt the bidding strategy

Pros:

- Partial elimination of exposure problem

Cons:

- An ad-hoc approach with many small details to be fixed. Example: incentives to wait until other agents reveal their preferences ⇒ necessity of various activity rules which inspire active bidding.
- Efficiency of the outcome is not guaranteed
- Inspire collusion and decrease competition. If goods are "almost substitutes", it is easy for agents to signal with their first bids what are the bundles they will compete for ⇒ easy to divide the market and thus pay less (this is why spectrum auction in Switzerland failed).

The main two approaches to combinatorial auctions:

Direct mechanisms (sealed-bid auctions)

- agents submit the profile of their valuations $(v_i)_{i \in N}$ (their "bids")
- a mechanism computes who gets what and how much pays

The main two approaches to combinatorial auctions:

Direct mechanisms (sealed-bid auctions)

- agents submit the profile of their valuations $(v_i)_{i \in N}$ (their "bids")
- a mechanism computes who gets what and how much pays

Pros:

- Easy to guarantee efficient allocation (theoretically)
- No exposure problem: nobody will pay for useless bundle

The main two approaches to combinatorial auctions:

Direct mechanisms (sealed-bid auctions)

- agents submit the profile of their valuations $(v_i)_{i \in N}$ (their "bids")
- a mechanism computes who gets what and how much pays

Pros:

- Easy to guarantee efficient allocation (theoretically)
- No exposure problem: nobody will pay for useless bundle

Cons:

- Serious algorithmic obstacles (to be discussed)

Examples of direct mechanisms

Extension of the first price auction:

- find a welfare maximizing allocation
$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$
(the so-called winner-determination problem)
- give the bundle A_i to agent i
- his payment is $p_i = v_i(A_i)$

Compute the outcome of FPA:

$A = \{a, b, c, \}, N = \{Alice, Bob, Claire\}$

Alice wants a and b together: $v_{Alice}(a, b) = 100$, $v_{Alice}(a) = v_{Alice}(b) = 0$

Bob needs a only: $v_{Bob}(a) = v_{Bob}(a, b) = 75$, $v_{Bob}(b) = 0$

Claire needs b only: $v_{Claire}(b) = v_{Claire}(a, b) = 40$, $v_{Claire}(b) = 0$

Remark: as in one-good FPA nobody will submit his truthful valuation $\Rightarrow$ mechanism is manipulable and resulting allocation may be inefficient. Also there is no explicit description of equilibrium bidding strategies and no RET.

Examples of direct mechanisms

Extension of the first price auction:

- find a welfare maximizing allocation
$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$
(the so-called winner-determination problem)
- give the bundle A_i to agent i
- his payment is $p_i = v_i(A_i)$

Compute the outcome of FPA:

$A = \{a, b, c, \}, N = \{Alice, Bob, Claire\}$

Alice wants a and b together: $v_{Alice}(a, b) = 100$, $v_{Alice}(a) = v_{Alice}(b) = 0$

Bob needs a only: $v_{Bob}(a) = v_{Bob}(a, b) = 75$, $v_{Bob}(b) = 0$

Claire needs b only: $v_{Claire}(b) = v_{Claire}(a, b) = 40$, $v_{Claire}(b) = 0$

Remark: as in one-good FPA nobody will submit his truthful valuation $\Rightarrow$ mechanism is manipulable and resulting allocation may be inefficient. Also there is no explicit description of equilibrium bidding strategies and no RET.

Examples of direct mechanisms

Extension of the first price auction:

- find a welfare maximizing allocation
$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$
(the so-called winner-determination problem)
- give the bundle A_i to agent i
- his payment is $p_i = v_i(A_i)$

Compute the outcome of FPA:

$A = \{a, b, c, \}, N = \{Alice, Bob, Claire\}$

Alice wants a and b together: $v_{Alice}(a, b) = 100$, $v_{Alice}(a) = v_{Alice}(b) = 0$

Bob needs a only: $v_{Bob}(a) = v_{Bob}(a, b) = 75$, $v_{Bob}(b) = 0$

Claire needs b only: $v_{Claire}(b) = v_{Claire}(a, b) = 40$, $v_{Claire}(b) = 0$

Remark: as in one-good FPA nobody will submit his truthful valuation $\Rightarrow$ mechanism is manipulable and resulting allocation may be inefficient. Also there is no explicit description of equilibrium bidding strategies and no RET.

Examples of direct mechanisms

Extension of the second-price auction (VCG mechanism):

- find a welfare maximizing allocation

$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$

- give the bundle A_i to agent i
- his payment is $p_i = SW_{-i}(\mathcal{A}) - SW^*_{-i}$, where $SW_{-i}(\mathcal{A}) = \sum_{i \in N \setminus \{i\}} v_i(A_i)$ and SW^*_{-i} is the maximal value of SW_{-i} over all allocations.

Compute the outcome of VCG:

$A = \{a, b, c, \}, N = \{Alice, Bob, Claire\}$

Alice wants a and b together: $v_{Alice}(a, b) = 100$, $v_{Alice}(a) = v_{Alice}(b) = 0$

Bob needs a only: $v_{Bob}(a) = v_{Bob}(a, b) = 75$, $v_{Bob}(b) = 0$

Claire needs b only: $v_{Claire}(b) = v_{Claire}(a, b) = 40$, $v_{Claire}(b) = 0$

Remark: truthful report is the dominant strategy $\Rightarrow$ always get efficient allocation.

Examples of direct mechanisms

Extension of the second-price auction (VCG mechanism):

- find a welfare maximizing allocation

$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$

- give the bundle A_i to agent i
- his payment is $p_i = SW_{-i}(\mathcal{A}) - SW^*_{-i}$, where $SW_{-i}(\mathcal{A}) = \sum_{i \in N \setminus \{i\}} v_i(A_i)$ and SW^*_{-i} is the maximal value of SW_{-i} over all allocations.

Compute the outcome of VCG:

$A = \{a, b, c, \}, N = \{Alice, Bob, Claire\}$
Alice wants a and b together: $v_{Alice}(a, b) = 100$, $v_{Alice}(a) = v_{Alice}(b) = 0$
Bob needs a only: $v_{Bob}(a) = v_{Bob}(a, b) = 75$, $v_{Bob}(b) = 0$
Claire needs b only: $v_{Claire}(b) = v_{Claire}(a, b) = 40$, $v_{Claire}(b) = 0$

Remark: truthful report is the dominant strategy $\Rightarrow$ always get efficient allocation.

Examples of direct mechanisms

Extension of the second-price auction (VCG mechanism):

- find a welfare maximizing allocation

$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$

- give the bundle A_i to agent i
- his payment is $p_i = SW_{-i}(\mathcal{A}) - SW^*_{-i}$, where $SW_{-i}(\mathcal{A}) = \sum_{i \in N \setminus \{i\}} v_i(A_i)$ and SW^*_{-i} is the maximal value of SW_{-i} over all allocations.

Compute the outcome of VCG:

$A = \{a, b, c, \}, N = \{Alice, Bob, Claire\}$
Alice wants a and b together: $v_{Alice}(a, b) = 100$, $v_{Alice}(a) = v_{Alice}(b) = 0$
Bob needs a only: $v_{Bob}(a) = v_{Bob}(a, b) = 75$, $v_{Bob}(b) = 0$
Claire needs b only: $v_{Claire}(b) = v_{Claire}(a, b) = 40$, $v_{Claire}(b) = 0$

Remark: truthful report is the dominant strategy $\Rightarrow$ always get efficient allocation.

Algorithmic issues with direct mechanisms

Difficulty 1: complexity of preferences

For general valuation functions, to report v_i agent i should specify $2^{|A|}$ numbers ($v_i(A')$ for any $A' \subset A$), i.e., the report has exponential size.
Example: For 20 goods, there are more than one million numbers.
Corollary: For practice the class of possible reports should be restricted.
This is a problem of choosing an appropriate bidding language, the class of reports that are

- expressive: rich enough to express the relevant complementarity/substitutability
- concise: the report is not too long
- easy to handle: both by humans and machines

Difficulty 1: complexity of preferences

For general valuation functions, to report v_i agent i should specify $2^{|A|}$ numbers ($v_i(A')$ for any $A' \subset A$), i.e., the report has exponential size.
Example: For 20 goods, there are more than one million numbers.
Corollary: For practice the class of possible reports should be restricted.
This is a problem of choosing an appropriate bidding language, the class of reports that are

- expressive: rich enough to express the relevant complementarity/substitutability
- concise: the report is not too long
- easy to handle: both by humans and machines

Difficulty 1: complexity of preferences

For general valuation functions, to report v_i agent i should specify $2^{|A|}$ numbers ($v_i(A')$ for any $A' \subset A$), i.e., the report has exponential size.

Example: For 20 goods, there are more than one million numbers.

Corollary: For practice the class of possible reports should be restricted. This is a problem of choosing an appropriate bidding language, the class of reports that are

- expressive: rich enough to express the relevant complementarity/substitutability
- concise: the report is not too long
- easy to handle: both by humans and machines

Examples of bidding languages

use the language of propositional logic.

- **Atomic language** (for single-minded agents):
  {laptop, mouse} : 100 means $v_i(A') = 100$ if A' contains laptop and mouse and 0, otherwise
- **OR language** (non-exclusive disjunction of atomic bids)
  {laptop, mouse} : 100 OR {smartphone} : 50 OR {smartphone, headphones} : 60 means:
  $v_i(laptop, mouse) = 100$
  $v_i(laptop, mouse, smartphone) = 150$
  $v_i(laptop, mouse, smartphone, headphones) = 160$ etc

Examples of bidding languages

use the language of propositional logic.

- **Atomic language** (for single-minded agents):
  {laptop, mouse} : 100 means $v_i(A') = 100$ if A' contains laptop and mouse and 0, otherwise
- **OR language** (non-exclusive disjunction of atomic bids)
  {laptop, mouse} : 100 OR {smartphone} : 50 OR {smartphone, headphones} : 60 means:
  $v_i(laptop, mouse) = 100$
  $v_i(laptop, mouse, smartphone) = 150$
  $v_i(laptop, mouse, smartphone, headphones) = 160$ etc

Examples of bidding languages

use the language of propositional logic.

- **Atomic language** (for single-minded agents):
  $\{$laptop, mouse$\}$: 100 means $v_i(A') = 100$ if A' contains laptop and mouse and 0, otherwise
- **OR language** (non-exclusive disjunction of atomic bids)
  $\{$laptop, mouse$\}$: 100 OR $\{$smartphone$\}$: 50 OR $\{$smartphone, headphones$\}$: 60 means:
  $v_i(laptop, mouse) = 100$
  $v_i(laptop, mouse, smartphone) = 150$
  $v_i(laptop, mouse, smartphone, headphones) = 160$ etc

Examples of bidding languages

use the language of propositional logic.

- **Atomic language** (for single-minded agents):
  $\{$laptop, mouse$\}$: 100 means $v_i(A') = 100$ if A' contains laptop and mouse and 0, otherwise
- **OR language** (non-exclusive disjunction of atomic bids)
  $\{$laptop, mouse$\}$: 100 OR $\{$smartphone$\}$: 50 OR $\{$smartphone, headphones$\}$: 60 means:
  $v_i(laptop, mouse) = 100$
  $v_i(laptop, mouse, smartphone) = 150$
  $v_i(laptop, mouse, smartphone, headphones) = 160$ etc
  Theorem: OR language can express any valuation such that $v_i(A_1 \cup A_2) > v_i(A_1) + V_i(A_2)$ for all disjoint $A_1, A_2 \subset A$ (i.e., without substitutability)

Examples of bidding languages

use the language of propositional logic.

- **Atomic language** (for single-minded agents):
  $\{$laptop, mouse$\}$: 100 means $v_i(A') = 100$ if A' contains laptop and mouse and 0, otherwise
- **OR language** (non-exclusive disjunction of atomic bids)
  $\{$laptop, mouse$\}$: 100 OR $\{$smartphone$\}$: 50 OR $\{$smartphone, headphones$\}$: 60 means:
  $v_i(laptop, mouse) = 100$
  $v_i(laptop, mouse, smartphone) = 150$
  $v_i(laptop, mouse, smartphone, headphones) = 160$ etc
  Theorem: OR language can express any valuation such that $v_i(A_1 \cup A_2) > v_i(A_1) + V_i(A_2)$ for all disjoint $A_1, A_2 \subset A$ (i.e., without substitutability)
  Remark: to handle substitutability add XOR (exclusive disjunction), which allows to express that agent i is ready to buy bundle B or bundle C but not both.

Difficulty 2: complexity of finding an efficient allocation

Bad news

Even for restricted classes of valuations (like OR) the winner determination problem

$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$

is NP-hard.

Remark: For practice this means that there is no algorithm for computing the Pareto-optimal allocation $\mathcal{A}$ that is much more efficient than comparing all possible partitions of A (there are exponentially many of them).

Corollary: Hence for $|A| = 25$ even modern supercomputers will fail to find $\mathcal{A} \Rightarrow$ efficient algorithms for computing approximately Pareto-optimal allocations are used.

Side remark/exercise: for n men and m women the Deferred acceptance algorithm allows to compute a stable matching in polynomial number of operations (check!). This allows to use this algorithm for large problems (school choice, job markets) with many agents.

Difficulty 2: complexity of finding an efficient allocation

Bad news

Even for restricted classes of valuations (like OR) the winner determination problem

$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$

is NP-hard.

Remark: For practice this means that there is no algorithm for computing the Pareto-optimal allocation $\mathcal{A}$ that is much more efficient than comparing all possible partitions of A (there are exponentially many of them).

Corollary: Hence for $|A| = 25$ even modern supercomputers will fail to find $\mathcal{A} \Rightarrow$ efficient algorithms for computing approximately Pareto-optimal allocations are used.

Side remark/exercise: for n men and m women the Deferred acceptance algorithm allows to compute a stable matching in polynomial number of operations (check!). This allows to use this algorithm for large problems (school choice, job markets) with many agents.

Difficulty 2: complexity of finding an efficient allocation

Bad news

Even for restricted classes of valuations (like OR) the winner determination problem

$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$

is NP-hard.

Remark: For practice this means that there is no algorithm for computing the Pareto-optimal allocation $\mathcal{A}$ that is much more efficient than comparing all possible partitions of A (there are exponentially many of them).

Corollary: Hence for $|A| = 25$ even modern supercomputers will fail to find $\mathcal{A} \Rightarrow$ efficient algorithms for computing approximately Pareto-optimal allocations are used.

Side remark/exercise: for n men and m women the Deferred acceptance algorithm allows to compute a stable matching in polynomial number of operations (check!). This allows to use this algorithm for large problems (school choice, job markets) with many agents.

Difficulty 2: complexity of finding an efficient allocation

Bad news

Even for restricted classes of valuations (like OR) the winner determination problem

$$\mathcal{A} = (A_i)_{i \in N} : SW = \sum_{i \in N} v_i(A_i) \to \max$$

is NP-hard.

Remark: For practice this means that there is no algorithm for computing the Pareto-optimal allocation $\mathcal{A}$ that is much more efficient than comparing all possible partitions of A (there are exponentially many of them).

Corollary: Hence for $|A| = 25$ even modern supercomputers will fail to find $\mathcal{A} \Rightarrow$ efficient algorithms for computing approximately Pareto-optimal allocations are used.

Side remark/exercise: for n men and m women the Deferred acceptance algorithm allows to compute a stable matching in polynomial number of operations (check!). This allows to use this algorithm for large problems (school choice, job markets) with many agents.

Fair division of indivisible goods: how to overcome negative results?

Fair division of indivisible private goods

The model

- A set of indivisible goods A is to be allocated to agents, N, without money transfers
- Allocation $\mathcal{A} = (A_i)_{i \in N}$ is a disjoint partition of A
- Utilities are additive: $u_i(A_i) = \sum_{a \in A_i} u_{ia}$

Question: What kind of fairness properties can we guarantee?

Remark: Using a richer bidding language is a good idea but, for now, nothing is known about fairness in such a setup.

Fair division of indivisible private goods

The model

- A set of indivisible goods A is to be allocated to agents, N, without money transfers
- Allocation $\mathcal{A} = (A_i)_{i \in N}$ is a disjoint partition of A
- Utilities are additive: $u_i(A_i) = \sum_{a \in A_i} u_{ia}$

Question: What kind of fairness properties can we guarantee?

Remark: Using a richer bidding language is a good idea but, for now, nothing is known about fairness in such a setup.

Fair division of indivisible private goods

The model

- A set of indivisible goods A is to be allocated to agents, N, without money transfers
- Allocation $\mathcal{A} = (A_i)_{i \in N}$ is a disjoint partition of A
- Utilities are additive: $u_i(A_i) = \sum_{a \in A_i} u_{ia}$

Question: What kind of fairness properties can we guarantee?

Remark: Using a richer bidding language is a good idea but, for now, nothing is known about fairness in such a setup.

Fairness notions from the divisible case

- Envy-free allocation: $u_i(A_i) \geq u_i(A_j) \ \forall i, j$
- Fair Share Guaranteed allocation: $u_i(A_i) \geq \frac{u_i(A)}{|N|} \ \forall i$

Bad news: such allocations may fail to exist. Guess the example!

Fairness notions from the divisible case

- Envy-free allocation: $u_i(A_i) \geq u_i(A_j) \; \forall i, j$
- Fair Share Guaranteed allocation: $u_i(A_i) \geq \frac{u_i(A)}{|N|} \; \forall i$

Bad news: such allocations may fail to exist. Guess the example!

Fairness notions from the divisible case

- Envy-free allocation: $u_i(A_i) \geq u_i(A_j)\ \forall i, j$
- Fair Share Guaranteed allocation: $u_i(A_i) \geq \frac{u_i(A)}{|N|}\ \forall i$

Bad news: such allocations may fail to exist. Guess the example!

Example: two agents and two goods a, b, where a is more desirable for both agents.

Two ways to escape non-existence results:

- Looking at the properties for a "typical" profile of preferences (either random or generated by real users)
- Finding an appropriate relaxation of fairness notion that guarantees existence.

Two ways to escape non-existence results:

- Looking at the properties for a "typical" profile of preferences (either random or generated by real users)

Theorem (Dickerson et al 2014)[1]

If the number of goods is large and u_{ia} are independent identically distributed random variables, then E-F (and thus FSG) allocations exist with high probability

- Finding an appropriate relaxation of fairness notion that guarantees existence.

[1] The Computational Rise and Fall of Fairness. John P. Dickerson, Jonathan Goldman, Jeremy Karp, Ariel D. Procaccia, and Tuomas Sandholm. AAAI-14: Proc. 28th AAAI Conference on Artificial Intelligence, pp. 1405-1411, Jul 2014. http://procaccia.info/papers/ef_phase.aaai14.pdf

Two ways to escape non-existence results:

- Looking at the properties for a "typical" profile of preferences (either random or generated by real users)

Theorem (Dickerson et al 2014)[1]

If the number of goods is large and u_{ia} are independent identically distributed random variables, then E-F (and thus FSG) allocations exist with high probability

- Finding an appropriate relaxation of fairness notion that guarantees existence.

[1] The Computational Rise and Fall of Fairness. John P. Dickerson, Jonathan Goldman, Jeremy Karp, Ariel D. Procaccia, and Tuomas Sandholm. AAAI-14: Proc. 28th AAAI Conference on Artificial Intelligence, pp. 1405-1411, Jul 2014. `http://procaccia.info/papers/ef_phase.aaai14.pdf`

Two ways to escape non-existence results:

- Looking at the properties for a "typical" profile of preferences (either random or generated by real users)

Theorem (Dickerson et al 2014)[1]

If the number of goods is large and u_{ia} are independent identically distributed random variables, then E-F (and thus FSG) allocations exist with high probability

- **Finding an appropriate relaxation of fairness notion that guarantees existence.** We will look at two examples

---

[1] The Computational Rise and Fall of Fairness. John P. Dickerson, Jonathan Goldman, Jeremy Karp, Ariel D. Procaccia, and Tuomas Sandholm. AAAI-14: Proc. 28th AAAI Conference on Artificial Intelligence, pp. 1405-1411, Jul 2014. `http://procaccia.info/papers/ef_phase.aaai14.pdf`

Maximin share (MMS)

A natural modification of FSG (Budish, 2011)[2]:

- the Maximin share of agent i is

$$MMS_i = \max_{\mathcal{A}} \min_{j} u_i(A_j).$$

- an allocation is MMS if for any i

$$u_i(A_i) \geq MMS_i.$$

Exercise: find MMS_i and an MMS allocation for the following problem

	a	b	c
u_{Alice} :	60	20	20
u_{Bob} :	55	25	20

[2]BUDISH, E. 2011. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. Journal of Political Economy 119, 6, 1061–1103.

Maximin share (MMS)

A natural modification of FSG (Budish, 2011)[2]:

- the Maximin share of agent i is

$$MMS_i = \max_{\mathcal{A}} \min_j u_i(A_j).$$

- an allocation is MMS if for any i

$$u_i(A_i) \geq MMS_i.$$

Exercise: find MMS_i and an MMS allocation for the following problem

	a	b	c
u_{Alice} :	60	20	20
u_{Bob} :	55	25	20

[2] BUDISH, E. 2011. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. Journal of Political Economy 119, 6, 1061–1103.

Existence of MMS allocations

For several years it was conjectured that MMS allocations always exist:

- computerized search for a counterexample on supercomputers failed
- MMS allocations exist for all preference profiles from Spliddit

But. . .

Existence of MMS allocations

For several years it was conjectured that MMS allocations always exist:

- computerized search for a counterexample on supercomputers failed
- MMS allocations exist for all preference profiles from Spliddit

But...

Theorem (Procaccia & Wang, 2014)[3]:

For $|N| \geq 3$ agents MMS allocation may fail to exists (a knife-edge counterexample with 12 goods). But $\frac{2}{3}MMS_i$ can always be guaranteed and there is a polynomial algorithm for computing such an allocation.

[3] Fair Enough: Guaranteeing Approximate Maximin Shares. David Kurokawa, Ariel D. Procaccia, and Junxing Wang. Journal of the ACM (forthcoming). `http://procaccia.info/papers/mms.pdf`

Existence of MMS allocations

For several years it was conjectured that MMS allocations always exist:

- computerized search for a counterexample on supercomputers failed
- MMS allocations exist for all preference profiles from Spliddit

But...

Theorem (Procaccia & Wang, 2014)[3]:

For $|N| \geq 3$ agents MMS allocation may fail to exists (a knife-edge counterexample with 12 goods). But $\frac{2}{3}MMS_i$ can always be guaranteed and there is a polynomial algorithm for computing such an allocation.

Conclusion: Though theoretically MMS allocations may fail to exist, from practical point of view they always exist.

[3] Fair Enough: Guaranteeing Approximate Maximin Shares. David Kurokawa, Ariel D. Procaccia, and Junxing Wang. Journal of the ACM (forthcoming). `http://procaccia.info/papers/mms.pdf`

Existence of MMS allocations

For several years it was conjectured that MMS allocations always exist:

- computerized search for a counterexample on supercomputers failed
- MMS allocations exist for all preference profiles from Spliddit

But...

Theorem (Procaccia & Wang, 2014)[3]:

For $|N| \geq 3$ agents MMS allocation may fail to exists (a knife-edge counterexample with 12 goods). But $\frac{2}{3}MMS_i$ can always be guaranteed and there is a polynomial algorithm for computing such an allocation.

Conclusion: Though theoretically MMS allocations may fail to exist, from practical point of view they always exist.

Remark: Computing MMS (or $\frac{2}{3}$MMS) allocation is not related to maximization of $\min_i u_i(A_i)$, as one might expect. The latter is known as Santa-Claus problem and is NP-hard.

[3] Fair Enough: Guaranteeing Approximate Maximin Shares. David Kurokawa, Ariel D. Procaccia, and Junxing Wang. Journal of the ACM (forthcoming). `http://procaccia.info/papers/mms.pdf`

EF-1

Envy-freeness up to one item[4]

an allocation $\mathcal{A}$ is envy-free up to one item if for all i and j

$$u_i(A_i) \geq u_i(A_j \setminus \{a_{ij}\})$$

for some $a_{ij} \in A_j$.

Easy Proposition:

EF-1 allocations always exist.

Sketch of the proof: Order agents somehow and consider a round-robin mechanism (serial dictatorship with non-unit demand):

- agents $1, ..n$ sequentially come and pick the most desired good
- repeat until all goods are allocated

Check that this procedure leads to EF-1 allocation. □

[4] LIPTON, R. J., MARKAKIS, E., MOSSEL, E., AND SABERI, A. 2004. On approximately fair allocations of indivisible goods. In Proceedings of the 6th ACM Conference on Economics and Computation (EC). 125– 131.

EF-1

Envy-freeness up to one item[4]

an allocation $\mathcal{A}$ is envy-free up to one item if for all i and j

$$u_i(A_i) \geq u_i(A_j \setminus \{a_{ij}\})$$

for some $a_{ij} \in A_j$.

Easy Proposition:

EF-1 allocations always exist.

Sketch of the proof: Order agents somehow and consider a round-robin mechanism (serial dictatorship with non-unit demand):

- agents $1, ..n$ sequentially come and pick the most desired good
- repeat until all goods are allocated

Check that this procedure leads to EF-1 allocation. □

[4] LIPTON, R. J., MARKAKIS, E., MOSSEL, E., AND SABERI, A. 2004. On approximately fair allocations of indivisible goods. In Proceedings of the 6th ACM Conference on Economics and Computation (EC). 125– 131.

EF-1

Envy-freeness up to one item[4]

an allocation $\mathcal{A}$ is envy-free up to one item if for all i and j

$$u_i(A_i) \geq u_i(A_j \setminus \{a_{ij}\})$$

for some $a_{ij} \in A_j$.

Easy Proposition:

EF-1 allocations always exist.

Sketch of the proof: Order agents somehow and consider a round-robin mechanism (serial dictatorship with non-unit demand):

- agents $1, ..n$ sequentially come and pick the most desired good
- repeat until all goods are allocated

Check that this procedure leads to EF-1 allocation. □

[4] LIPTON, R. J., MARKAKIS, E., MOSSEL, E., AND SABERI, A. 2004. On approximately fair allocations of indivisible goods. In Proceedings of the 6th ACM Conference on Economics and Computation (EC). 125– 131.

Efficient EF-1 allocations

Theorem (Caragianis et al 2016)[5]:

An allocation maximizing the Nash product $\prod_{i \in N} u_i(A_i)$ is Efficient and EF-1.

Corollary: the Nash rule provides fair and efficient solutions both in divisible and indivisible cases. For indivisibilities, its relation to market-equilibrium is an open question.

Bad news: maximization of the Nash product is NP-hard for indivisible items ⇒ many papers on polynomial approximation algorithms

Good news: if it is known that u_{ia} belong to a fixed lattice (e.g., 1...1000 points), there is a polynomial algorithm to compute the exact solution. It is now used on Spliddit.

[5] The Unreasonable Fairness of Maximum Nash Welfare. Ioannis Caragiannis, David Kurokawa, Herve Moulin, Ariel D. Procaccia, Nisarg Shah, and Junxing Wang. EC-16: Proc. 17th ACM Conference on Economics and Computation, pp. 305-322, Jul 2016. http://procaccia.info/papers/mnw.pdf

Efficient EF-1 allocations

Theorem (Caragianis et al 2016)[5]:

An allocation maximizing the Nash product $\prod_{i \in N} u_i(A_i)$ is Efficient and EF-1.

Corollary: the Nash rule provides fair and efficient solutions both in divisible and indivisible cases. For indivisibilities, its relation to market-equilibrium is an open question.

Bad news: maximization of the Nash product is NP-hard for indivisible items $\Rightarrow$ many papers on polynomial approximation algorithms

Good news: if it is known that u_{ia} belong to a fixed lattice (e.g., 1...1000 points), there is a polynomial algorithm to compute the exact solution. It is now used on Spliddit.

[5] The Unreasonable Fairness of Maximum Nash Welfare. Ioannis Caragiannis, David Kurokawa, Herve Moulin, Ariel D. Procaccia, Nisarg Shah, and Junxing Wang. EC-16: Proc. 17th ACM Conference on Economics and Computation, pp. 305-322, Jul 2016. `http://procaccia.info/papers/mnw.pdf`

Efficient EF-1 allocations

Theorem (Caragianis et al 2016)[5]:

An allocation maximizing the Nash product $\prod_{i \in N} u_i(A_i)$ is Efficient and EF-1.

Corollary: the Nash rule provides fair and efficient solutions both in divisible and indivisible cases. For indivisibilities, its relation to market-equilibrium is an open question.

Bad news: maximization of the Nash product is NP-hard for indivisible items $\Rightarrow$ many papers on polynomial approximation algorithms

Good news: if it is known that u_{ia} belong to a fixed lattice (e.g., 1...1000 points), there is a polynomial algorithm to compute the exact solution. It is now used on Spliddit.

[5] The Unreasonable Fairness of Maximum Nash Welfare. Ioannis Caragiannis, David Kurokawa, Herve Moulin, Ariel D. Procaccia, Nisarg Shah, and Junxing Wang. EC-16: Proc. 17th ACM Conference on Economics and Computation, pp. 305-322, Jul 2016. `http://procaccia.info/papers/mnw.pdf`

Efficient EF-1 allocations

Theorem (Caragianis et al 2016)[5]:

An allocation maximizing the Nash product $\prod_{i \in N} u_i(A_i)$ is Efficient and EF-1.

Corollary: the Nash rule provides fair and efficient solutions both in divisible and indivisible cases. For indivisibilities, its relation to market-equilibrium is an open question.

Bad news: maximization of the Nash product is NP-hard for indivisible items $\Rightarrow$ many papers on polynomial approximation algorithms

Good news: if it is known that u_{ia} belong to a fixed lattice (e.g., 1...1000 points), there is a polynomial algorithm to compute the exact solution. It is now used on Spliddit.

[5] The Unreasonable Fairness of Maximum Nash Welfare. Ioannis Caragiannis, David Kurokawa, Herve Moulin, Ariel D. Procaccia, Nisarg Shah, and Junxing Wang. EC-16: Proc. 17th ACM Conference on Economics and Computation, pp. 305-322, Jul 2016. http://procaccia.info/papers/mnw.pdf

Main take-away points:

Importance of complexity:

- Agents cannot report too much information and the outcome of a mechanism cannot be found without fast algorithm
- If there is no fast algorithm, various approximation methods are used

Ways to avoid non-existence of mechanisms with nice properties

- Mechanisms may behave badly for some knife-edge cases that never occur in practice and have nice properties for all real-life preference profiles
- The definition of “what is nice” may be weakened a bit to guarantee existence

References:

- Shoham, Yoav; Leyton-Brown, Kevin (2009). Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations. New York: Cambridge University Press. http://www.masfoundations.org/download.html
- Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V. V. (Eds.). (2007). Algorithmic game theory (Vol. 1). Cambridge: Cambridge University Press. http://www-cgi.cs.cmu.edu/afs/cs.cmu.edu/Web/People/sandholm/cs15-892F13/algorithmic-game-theory.pdf
- mentioned articles