Fair Division **Summer 2018**

Lecture 1: Cake Cutting

Instructor: Rohit Vaish *Scribes: Varad P.*

1.1 Cake Cutting Model

1.1.1 Set of n agents

$$N = \{1, 2, 3, ...n\}$$

1.1.2 Resource

Let us say that the resource is a **Cake** [0,1]. Cake is a heterogenous divisible resource. [0,1] is the simplest model of divisible resource.

1.1.3 Preference of Agents

The preference of agents is given by validation function V_i.
Let's say there is an interval $I = [x, y]$
So, the valuation function will be:

$$V_i(I) = V_i([x, y]) = V_i(x, y) \tag{1.1}$$

1.1.3.1 Assumptions on V_i

1. Normalisation:

   $$\forall i \in N,$$
   $$V_i(0, 1) = 1$$

   That means, "The total value of the cake is the same for everybody"

2. Divisibility:
   For an interval $[x, y] \subseteq [0, 1]$ and for any $\lambda \in [0, 1]$, $\exists z \in [x, y]$ such that $V_i(x, z) = \lambda V_i(x, y)$.
   It is a **"No point mass valuation"**.

3. Non-negativity:

   $$\forall i \subseteq [0, 1]$$

,

$$V_i(I) \geq 0$$

4. Additivity:
   For any disjoint intervals I, $I^{'}$,

$$V_i(I \cup I^{'} = V_i(I) + V_i(I^{'}) \tag{1.2}$$

5. Piece of Cake * It is a finite union of disjoint intervals. The intervals are together referred to as a piece. Additivity is the extended valuation to the piece of cake.

Note 1. Non-negativity and Additivity together imply a monotonicity implementation. (The more is the cake, the better it is).

$$S \leq T \leq [0, 1]$$

$$V_i(S) \leq V_i(T)$$

1.1.4 Examples

Some of the most used examples of valuation function are:

1. Uniform

$$V_i(I) = \int_{x \in I} pi(x)dx \tag{1.3}$$

   - pi(x) is called the Value Density Function.
   - Can represent $V_i()$ as an integral of a value density function.

2. Piecewise Uniform
3. Piecewise constant
4. Polynomial value density

A non-example

5. Point Mass

1.2 Goal

Goal: The goal is to find an allocation

$$A = (A_1, A_2...A_n)$$

of [0,1] that is fair, where A_i is a piece of cake.

1.3 Fairness Notions

Allocation $A = (A_1, A_2...A_n)$ satifies:

- Proportionality: For every $i \in N$,
$$V_i(A_i) \geq \frac{1}{n}.1$$
where "1" is the normalized $V_i[0, 1]$
- Envy-freeness: For every i,j ∈ N,
$$V_i(A_i \geq V_i(A_j$$
That means that valuation of i should be greater than or equal to the valuation of j.

Exercise: Which one is stronger? EF or proportionality?

- For two agents, proportionality is equal to envy-freeness (EF)
- For three agents, think of valuations/allocation where we can have proportionlaity but not envy-freeness.

Two Important Questions:

Q1. Do fair divisions of a cake always exists? **EXISTENCE**

Q2. Can we find a fair division efficiently? **COMPUTATION**

1.4 Existence Model

Theorem (Alon '87) : Let $V_1, V_2, V_3...V_n$ be the valuation functions induced by the continous density functions. Then, it is possible to cut the cake in atmost $n^2 - n$ places and partion the resulting $n^2 - n + 1$ intervals into n pieces of $A_1, A_2, A_3...A_n$ such that $\forall i, j \in N$,
$$V_i(A_j) = \frac{1}{n}$$

Remarks :

1. A very general existence result
2. Requires continuos value density – does not cover piecewise uniform/piecewise constant. But there are easy ways of guaranting fair outcomes in the excluded casts.
3. $V_i(A_j) = \frac{1}{n}$ is called a **Perfect Partition**
4. Proof is non-constructive – does not give an algorithm.

1.5 Computational Model

Valuations are "continuous" objects, but algorithms are "discrete". So there is a need to establish a way/model in which algorithm interacts with valuation.
Robertson Webb Query Model : Any algorithm is allowed to make the following two types of queries to the agents (This is how an algorithm "accesses" the valuations) :

(1) $cut_i(x, \alpha)$: Starting from point x, player i should make a cut "y" such that

$$V_i(x, y) = \alpha$$

Player i returns $y \in [0, 1]$

(2) $eval_i(x, y)$: Output is $V_i(x, y)$. We get to know how much is [x,y] worth to agent i.

Complexity of an algorithm = No. of Robertson-Webb (RW) queries.

Note 2. This is a query complexity, not computational complexity. "Hardness" here means "a lot" of information about the valuations is required.

1.6 Cake Cutting Algorithms

1.6.1 "I Cut, you Choose" Algorithm

1.6.1.1 For 2 agents

Step 1: Agent 1 cuts the cake into two pieces, which are considered equal by A

$$cut_1(0, 1) = y$$

Step 2: Agent 2 chooses a piece of cake that it like. Allocation depends on whether "<1/2" or ">1/2"

$$eval_2(0, y)$$

Theorem: For n=2, an EF/Proportionality allocation can be computed with two queries in the RW model.

1.6.1.2 Proportionality for n >2 agents

Thought Experiment: Moving Knife
A reference slides a moving knife from 0 towards 1. As soon as the value of some agents for the [0,x] piece reaches $\frac{1}{n}$, it shouts "stop". The piece is assigned to that agent, and it is kicked out.The

game now resumes with the remaining n–1 agents and the cake [x,1]. We now know that the agent leaves gets its Proportionality. What do we know about the agents that didn't cut shout?
They Value:

$$V_i[0, x] \leq \frac{1}{n}$$

that is,

$$V_i[x, 1] \geq \frac{n-1}{n}$$

Dubins-Spanier algorithm

Step 1 : Each agent i makes a point X_i such that

$$V_i(0, x_i) = \frac{1}{n}$$

Step 2 : The agent i^* with the leftmost mark leaves with the piece $[0, x_{i^*}]$

Step 3 : Repeat the process with the remaining agents and remaining cake.

No of queries $= n + (n-1) + ... + 2 = O(n^2)$

Theorem: For any n, a proportionlaity algorithm can be computed with $O(n^2)$ queries in RW Model.

Possible to do better via divide and conquer. Even-paz algorithm gives a proportionality allocation with $O(nlogn)$ queries. And this is the best we can hope for.

Theorem (Edmunds and Pruhs, 2006: Any proportional cake cutting algorithm requires $\Omega(nlogn)$ queries in RW model.

Note 3. Envy-freeness works for n=2 in Cut and Choose algorithm. next will we discuss about n=3.

1.6.1.3 Envy-Freeness for n=3 agents

Selfridge-Conway algorithm
This algorithm is only for 3 agents and let the agents be A, B, C.

Phase-I

Step 1 : Agent A divides [0,1] into three equal parts (according to A)

Step 2 : Agent B creates a two way tie(valuation-wise) for the largest pieces possibly by trimming the strictly largest piece. Call the trimming T and the main cake M.

Step 3 : Agents C >B >A pick a piece from M each in that order, with the condition that B picks the trimmed piece if C doesn't

So far:

(1) The partial allocation is EF

(2) Agent A has an irrevocable advantage over whoever gets the trimmed piece.

(3) Combining the two partial EF alloccations gives EF (as they are disjoint) Only need an EF division of the trimming T

Key: Phase I as is because of trimming and infinitism. Use the one thing about T that we didn't know about [0,1] which is an irrevocable advantage.

Phase-II

Step 1 : Whoever of B or C got the untrimmed piece is the new cutter, the other one is the non-cutter, The cutter divides T into three equal pieces.

Step 2 : Agents non-cutter >A >cutter pick in that order.

Theorem: The allocation returned by Selfridge-Conway algorithm is Envy-Free. **Exercise:** How many queries are there in the above algorithm?

Note 4. Agents might get "disconnected" pieces under Selfridge-Conway Algorithm

Question: Does there exists a connected EF division? **Answer:** Yes! There do exists a connected EF division, we will see about it in the next lecture.
Question: Can we compute such a connected EF division? **Answer:** Provably no! No finite protocol can. (Stromquit, 2008)

Note 5. If we relax both connectedness and EF, then a very simple algorithm does the trick. Round Robin.

- Each agent makes $\frac{1}{\varepsilon}$ marks, with each piece between consecutive marks being worth ε. That is, total $\frac{n}{\varepsilon}$ cut queries,
- Treat each piece between consecutive marks as an indivisble good.
- Move through agents 1 >2 >... >n, where each agent picks its favorite "good".

Since no agent valuesany good for more then ε, we get ε–EF allocation.

Relax Connectedness:

n=3 Selfbridge-Conway (small # queries)

n>3 Brans amd Taylor(finite but unbounded
Aziz and Mackenzic(boiunded protocol, doesn't depend on valuations).
Bound is $n^{n^{n^{n^n}}}$

Relax EF to approximate EF:

- A connected approximate EF allocation via Spuner's lemma.
- Bounded query complexity(possibly exponential) is n.

Fair Division **Summer 2018**

Lecture 2: Fair Division of Indivisible Goods

Instructor: Rohit Vaish *Scribes: Varad P.*

2.1 The Model

2.1.1 Set of n agents

$$N = \{1, 2, 3, ...n\}$$

2.1.2 Indivisble Goods

Let there be 'm' indivisble goods. So, let the set be represented by:

$$M = \{g_1, g_2, g_3...g_m\}$$

2.1.3 Preference of Agents

The preference of agents is also given by the validation function V_i like in the previous lecture, So, the valuation function is:

$$V_i = 2^m : \mathbb{R}$$

where, 2^m is a space of all the subsets off the set of goods, and then it is mapped to some number $\mathbb{R}$

2.1.4 Assumptions on V_i

1. Additivity: For any S $\subseteq$ M, and any i $\in$ N,

$$V_i(S) = \sum_{g_j \in S} V_i(\{g_j\}) = \sum_{g_j \in S} V_{ij}$$

2. Non-negativity: For any g $\in$ M and any i $\in$ N,

$$V_i(\{g_j\}) \geq 0$$

Note 1. Non-negativity and Additivity together imply a monotonicity implementation. (The more goods, the better).

$$S \subseteq T$$
$$V_i(S) \leq V_i(T)$$

Exercise: To prove the above monotonicity implementation.

2.1.5 Input/Output

Input: The input to any fair division algorithm is going to be n numbers per agent, that is, it will be a valuation matrix with agents being the rows and goods being the columns.

$$\{V_{ij}\}_{i \in N, g_j \in M}$$

Output: The output to a fair division algorithm will be an allocation

$$A = (A_1, A_2, A_3 ... A_n)$$

which is simply a partition of set M. It is to be noted that each $A_i \subseteq$ M, for all i $\in$ M and $A_i \cap A_k = \{\Phi\}$

2.2 Fairness Notions

Allocation $A = (A_1, A_2 ... A_n)$ satifies:

- Proportionality: For every $i \in N$,

  $$V_i(A_i) \geq \frac{1}{n}.V_i(M)$$

  where "1" is the normalized $V_i[0, 1]$

- Envy-freeness: For every i,j $\in$ N,

  $$V_i(A_i \geq V_i(A_j$$

  That means that valuation of i should be greater than or equal to the valuation of j.

Note 2. Proportionality and Envy-freeness fail to exist for 2 agents, 1 good. So, in this setting there is no proportional allocation nor an envy-free allocation. Thus, we will have to redefine the problem.

2.2.1 Envy-Freeness upto one good (EF1)

EF1 is a relaxation of EF. It is defined so as to set up a benchmark for solving algorithms. It means $\forall i, j \in N \ \exists g \in A_j$ such that

$$V_i(A_i) \geq V_i(A_j \backslash \{g\})$$

So, the term on the left is the valuation of i on its own bundle while the term on the right is also the valuation of i except that we are throwing away one of the goods so that it becomes EF.

Note 3. If for any pair of agents, if we kill the envy by removal of some good, then the allocation is approximately envy-free in this case.

Exercise: Does EF1 implies Proportionality? What about vice-versa?

2.2.2 Maximin Share (MMS)

We say that an allocation is MMS Fair or simply MMS if the utility of every agent is atleast some threshold. That is,

$$\mu_i := max_{A \in \prod_n(M)}\{min_{j \in M}\{V_i(A_j)\}\}$$

where, $\prod_n(M)$ is the space of all partitions of goods in M among n agents.

μ_i is defined in the discre analogue of the Cut and Choose protocol. Thus, MMS is a threshold that agent i can guarantee itself by being the cutter.

Note 4. By definition, for each agent, $\exists$ some allocation that realizes μ_i. But, the same allocation must not realize μ_k for k $\neq$ i.

However, if it does, then we call it MMS-fair. Allocation A is MMS if $V_i(A_i) \geq \mu_i$ for all i $\in$ N. **Approximation of MMS** is given by $\alpha - MMSifV_i(A_i) \geq \alpha.\mu_i$ for all i $\in$ N.

Exercise: Does Proportionality implies MMS?
Answer: Yes! MMS is a subset of Prop. so if our result is Prop. then it is definitely MMS.

Exercise: Does EF1 implies MMS? What about vice-versa?
Hint: We have seen counter examples already.

2.3 Existence Results

* EF, Proportionality fails to exists for two agents, one good.
* MMS allocations might even fail for three agents, ffour goods.
* EF1 always exists.

Suppose, we want to write an inline comment...

2.4 Computation

Theorem: Even with two agents and identical valuations, deciding whether an envy-free or Proportional allocation exists, is **NP-Complete**.

Exercise: Prove the above theorem. (**Hint:** Shown that it reduces from the Partitiom problem.)

Theorem: MMS allocation always exists for two agents, even with non-identical valuations.
Proof: Let A be an allocation that realizes MMS value of agent 1. So, $V_1(A_1) \geq \mu_1$. We claim

that the same allocation also realize μ_2. Imagine that the agent 1 actually "cuts" M into A_1 and A_2 and agent 2 "picks" A_2. Therefore,

$$V_2(A_2) \geq V_2(A_1)$$

. Trivially,

$$V_2(A_2) \geq V_2(A_2)$$

. Thus, adding the above two equations, we get,

$$V_2(A_2) \geq \frac{1}{2}.V_2(M) \geq V_2(B_2) = \mu_2$$

$\frac{1}{2}.V_2(M)$ is the Proportionality threshold and B is the allocation that realizes μ_2. So, $V_2(B_2) \leq V_2(B_1)$. That is, $V_2(B_2) \leq \frac{1}{2}.V_2(M)$.

Theorem: Even with two agents and identical valuations, deciding if $\mu_i \geq \theta$ is NP-complete.
Proof: Use $\theta = \frac{1}{2}.\sum a_i$ in the Partition reduction.

This is a comment

2.5 Algorithms

2.5.1 Greedy Round-Robin Algorithm

- Fix an ordering over the agents $1 > 2 > ... > n$
- Starting from agent 1, move through thte agents in the round-robin fashion. At each step, i picks its favorite good from the unallocated set.
- Stop when all goods have been allocated.

Claim: The above algorithm returns EF1 allocation and runs in polynomial time.

Proof: To show EF1, consider any two agents i,j ∈ N.

Case I:

i>j: Then, good-by-good, i will prefer A_i over A_j. Therefore, i won't envy j.

Case II:

j>i: Now, suppose j picks up good g_1, i picks good $g_1^{'}$ in round 1, then j picks up good g_2, i picks good $g_2^{'}$ in round 2 and so on... If we remove the first bundle g_1, then we observe that,

$$V_i(g_1^{'}) \geq V_i(g_2)$$

$$V_i(g_2^{'}) \geq V_i(g_3)$$

and so on

$$V_i(g_{k-1}^{'}) \geq V_i(g_k)$$

Thus,

$$\sum V_i(A_i) = \sum V_i(A_i \backslash g_1)$$

LHS represents how much i values its own bundle while RHS means how much i values the bundle of A_j with one good removed. Since, $g_1 \in A_j$, A is EF1.

2.5.2 Envy Graph Algorithm

Proposed by **Lipton et al. EC'04**

- Fix an order over the goods $g_1, g_2, g_3 ... g_m$
- Initialize $A^{(0)} := \{\Phi, \Phi, \Phi \ ... \ \Phi\}$
- For k = 1,2, ... m
  - Assign g_k to a source of $G_{A^{(k-1)}}$.
    Call the new allocation $B^{(k)}$ (Must exist due to acyclicity invariant)
  - $A^{(k)} \leftarrow$ ACYCLIZE($B^{(k)}$)
    If $B^{(k)}$ is EF1, then $A^{(k)}$ is EF1.
- Return $A^{(m)}$ as the output.

This time, instead of n agents, we will consider the goods in a specific order. Before that. define a useful visualisation/analysis tool called an "Envy Graph". Given any allocation A, the envy graph G_A associated with A is defined on the vertex set N. It is a possibly bidirected graph such that, if and only if

$$V_i(A_i) < V_i(A_j)$$

that is, i envies j. Note that, G_A can have cycles (envy relation is not transitive).
Observation: Can make G_A acyclic without increasing envy.

Formally,
Lemma: Let A be on EF1 allocation (possibly partial) such that G_A is cyclic. Then, we can compute, in polynomial time. an allocation B such that G_B is acyclic and B is EF1.

Proof: (via cycle elimination algorithm)
Let $C = 1 \rightarrow 2 \rightarrow 3 \rightarrow ...k-1 \rightarrow k \rightarrow 1$ be a cycle in G_A. If G_A is cyclic, then a cycle couldn't be completed in traversal time. Do a "cyclic swap" to obtain $A^{'}$

$$A_1^{'} = A_2, A_2^{'} = A_3 ... A_k^{'} = A_1;$$

$$A_l^{'} = A_l \ \forall l \in N \backslash [k]$$

Two Key Observations:

(1) A' is EF1: Consider i,j $\in$ N such that:

- i $\in$ N$\setminus$ C, j $\in$ N$\setminus$ C : **No Change**
- i $\in$ N$\setminus$ C, j $\in$ C : **Edge gets shifted**
- i $\in$ N, j $\in$ N$\setminus$ C : **i only becomes happier**
- i $\in$ N, j $\in$ C : **Requires A being EF1** but not additivity (only monotonicity)

(2) No. of edges in $G_{A'}$ <No. of edges in G_A
Cyclic swap kills atleast one edge in C and does not add any new edge anywhere.

If $G_{A'}$ is cyclic, repeat the process. Guaranteed to terminate in polynomial time with A' that is EF1 such that $G_{A'}$ is cyclic.

Q. Why care about acyclicity? Beacuse a DAG has a source (an agent that no one envies)

* So, if A is EF1, G_A is acyclic, and we give a good to the source agent, the neew allocation is still EF1.
* We can "acyclize" the new allocation without losing EF1.

Invariant: $A^{(k)}$ is EF1 and $G_{A^{(k)}}$ is acyclic (Note: Proof by induction)

Theorem: The Envy Graph Alogorithm is EF1 in polynomial time even for non-additive valuation.

Corollary: The envy graph algorithm gives a polynomial time algorithm for ε–Ef cake division.

Proof Idea: Each player makes $\frac{1}{\varepsilon}$ marks for ε valued subintervals $\left(\text{Total: } \frac{n}{\varepsilon} \text{ cut queries}\right)$. Each interval between consecutive marks is an indivisible good of value at most ε. Do Round-Robin or Envy-Graph

Fair Division **Summer 2018**

Lecture 3: Approximately Envy-Free cake cutting and Rent Division

Instructor: Rohit Vaish *Scribes: Varad P.*

Recall:

- We discussed about divisible resources, for example, cake [0,1].
- There is a set of n agents: N = $\{1, 2, ...n\}$
- Preferneces of agents given by valuation function. We also made some assumptions like Non-negativity, Additivity, Divisibility.
- We also defined fairness notions like Proportionality, Envy-Freeness. We discussed about Prop. and EF allocations
- Some of the algorithms discussed for EF are: "I cut, you choose: for n=2", "Selfridge-Conway: for n=3", "Brands-Taylor Protocol: for any n (A finite but unbounded protocol)" and "Aziz and McKinsey: for any n(bounded protocol)

Lecture Overview:

- existence of a connected EF allocation of cake.
  It always exists. The proof was given by Stromquist '80. There is no finite protocol or algorithm for this.
- A bounded protocol for the "approximate envy-free" division.
- A fair division problem called the "Rent Division".

3.1 Sperner's Lemma

Before getting into the algorithm, thrre's is a very useful combinatorial result called the "Sperner's Lemma".

Consider there's a triangle T that has been divided into several smaller triangles called the "elementary triangle" or "baby triangles". Basically, we are working with Simplex and the division is called "Traingulation".

The division can be as small as possible

Sperner's Labelling:

- The main vertices should have distinct labels
- The label of any vertex on the boundary is either of the labels of the main vertices adjacent to it.
- The internal vertices can have any labels assigned to them.

Statement: Sperner's Lemma states that any Sperner labelled Traingulation has an odd number of fully labelled elementary triangles. in particular, there is atleast one.

Proof: Think of the traingulated T as a "house", the elementary triangles are "rooms" and each edge (1,2) as a "door". So, we can make the following observations:

- Number of doors on the boundary is odd. (orientation is not important)
- A room has either zero, one or two doors.

So, that means we need to find a room with only door, as that will lead us to a fully-labelled elementary triangle. Here's how to find it:

+ Start walking in from any door on the boundary, with "1" to the left, and "2" on the right. Either we have found a room with one door, or there must be another door that exists.

  This path cannot cycle back because atmost there is 1 entry and 1 exit.

+ This walk either terminates in a colorful/fully labelled elementary triangle or throws us out of the house. But, according to our first observation, we have another door remaining to enter T.
+ Since the number of rooms is finite, the walk is bound to terminate.

Note 1. Note that we cannot leave from other edges as according to Sperner's labelling, there are no doors present on those edges.

We also cannot revist a room because of the property that no room has 3 doors, that is, no path double-barks on itself.

Thus, there is atleast one fully labelled elementary triangle. Furthermore, any fully labelled elementary triangle "not reachable" from the boundary must be paired with another such triangle.

3.2 Cake Cutting using Sperner's Lemma

The proof was given by **Forest Simmons**.
Divisible resource : Cake [0,1] and $N = \{1, 2, 3...n\}$ agents. An EF division is such that for all i $\in$ N and j $\neq$ i, $V_i(A_i) \geq V_i(A_j)$. We wish to make the fewest number of cuts (n–1) cuts for n agents. Any feasible cut is given by $x_1, x_2...x_n$ such that:

It is to be noted that Sperner's lemma is valid for higher dimensions as well, thus we can use it for any number of agents

(1) $x_i \geq 0, \forall i \in [n]$

(2) $\sum_{i \in N} x_i = 1$

In particular, when n=3, then $x_1, x_2, x_3 \geq 0$ and $\sum_{i=1}^{3} x_i = 1$. So, although the figure is a 3–D object, the space of cuts is 2–D object, and the object is an **Equilateral Traingle**.

3.2.1 Assumptions on preferences of agents

Assumption 1: Given any cut (x_1, x_2, x_3), each player always prefers atleast one piece.

Assumption 2: Hungry players: Non-empty piece>Empty Piece

Triangulate the simplex and assign "ownership" of each vertex to a player such that each elementary traingle has all the three owners. Say the players are A, B, C. Then, each elementary triangle is an ABC Triangle. We will now assign labels to the vertices based on the preferences of the agents. For each vetex(cut), ask the owner "Which piece do you like the best?" **Tie-break arbitarily**

Note 2. 1 By "hungry" agents assumption, each owner of the main vertex answer the only piece with the non-zero length.

2 For the edges, the label is one of the pieces with non-zero lengths.

By Sperner's lemma, there is atleast one elementary traingle with different labels (1,2,3).

Approximate Envy-Free Connected Division: By the ownership of elementary triangles, we have a geometrically similar set of cuts where different agents prefer different pieces.

If the valuations are closed topologically, then the limiting cut has to be envy free.

Assumption 3: Closedness:
Let X = (x_1, x_2, x_3), where X is a point in the simplex. Let $X^{(1)}, X^{(2)}, X^{(3)}$... be an infinite bounded sequence of cards. Valuations are topologically closed. For any sequence $\{X^{(n)}\}$, if agent i prefers k at each of $X^{(1)}, X^{(2)}, X^{(3)}$... then piece i prefers piece k even in the limit.

3.3 Rent Division

Let there be three people who want to divide three rooms in a house between them such the total rent paid is 1. We have to divide the rent amongnst the three in an envy-free manner.

Assign rent to the rooms so that at those proces, each person prefers a differnt room

3.3.1 Preferences of agents:

(1) Agents are able to answer the question: "If room 1 is priced at p_1, room 2 at p_2, room 3 at p_3, which room would you prefer?"

(2) Agents prefer a free room over a non-free room.

3.3.2 Constraints

- $p_1, p_1, p_1 \geq 0$
- $\sum p_i = 1$

3.3.3 Analysis

As before, traingulate and assign ownership and gather labels. Therefore, free room assumotion implies: But, we can see that this is not a Sperner labelling. So what to do?

3.3.3.1 Case analysis

To prove that there exists a fully labelled elementary triangle
In each case, there is only one (1 or 2) door for entry. Cannot escape through the external walls (as only one door) Each room has 0,1 or 2 doors (note that this is a property of traingle not of sperner labelling)

3.3.3.2 Dual Sperner Labelling

3.3.3.3 A Whacky proof

Fair Division **Summer 2018**

Lecture 4: Algorithms for Indivisible Goods

Instructor: Rohit Vaish *Scribes: Varad P.*

Recall:

- We discussed about setup of indivisible goods in lecture 2
- There is a set of n agents: N = $\{1, 2, ...n\}$ There is a set of m goods: M = $\{g_1, g_2...g_m\}$
- Preferneces of agents given by valuation function. We also made some assumptions like Non-negativity, Additivity.
- We also defined fairness notions like Proportionality, Envy-Freeness, EF1 and Maxmin Share.

Lecture Overview:

- We will discuss about a new fairness notion – NSW(Nash Social Welfare)
- NSW is an "objective", in contrast with properties EF, EF1, MMS.

Proportionality, envy-freeness and their relaxations, are all properties, they are either satisfied, or not satisfied. NWS is an "objective" function. An objective function assigns a number to every allocation and alocations with higher number are intuitively fairer. So, objective is a **measure.**

So, given an allocation A,

$$NSW(A) = (\prod_{i \in N} V_i(A_i)^{\frac{1}{n}}$$

Optimal solution of NSW(A) is the Nash Optimal Allocation. It is given by:
A^* = arg max NSW(A).

4.1 Existence

Once we fix a problem instance, every problem is assigned a problem instance and there has to be some allocation with the highest number, thus there has to be some A^*. Therfore, it always exists.

Theorem (Caragiannis et al, EC '16) A Nash optimal allocation is EF1 and PO. PO is the **Pareto Optimality**. This is a notion of efficiency of an allocation. An allocation A is PO if for no other allocation B,

$$V_i(A_i) \leq V_i(B_i) \forall i \in N$$

and

$$V_k(A_k) < V_k(B_k) \forall k \in N$$

"Cannot make some agent happy without making someone else unhappy".

Note 1. - We know that EF1 always exists.

- We also know that PO always exists,

This theorem states that both EF1 and PO coexist!

4.2 Computation

Theorem: Maximum NSW is NP-complete. Infact, it is APX-complete (Lee '17, IPL). This, means that approximating the Nash social way for objective to a certain constant factor that is also an NP-Hard problem.

Exercise: Prove the above theorem. Hint: Reduce from Partition.

Goal: Design exact algorithms for restricted valuation. We are going to discuss a special case of NSW maximization, "Binary Valuation".

$$V_i(\{g\} \in \{0, 1\}$$

This means that an agent either likes a good or doesn't like the good.

Theorem: For binary valuations. a Nash optimal allocation can be computed in polynomial time.

A few simplifications:

(1) How does a Nash optimal for binary valuations look this?
Each good is assigned to an agent that values it at 1.

(2) There must be some allocation with NSW>0.

4.3 Algorithm

Given a suboptimal allocation, whose Nash Welfare is strictly less than its Nash allocation. Objective is to perform operations on the suboptimal allocation to improve its Nash Welfare.

- Pairwise Swap:
Given a suboptimal allocation, what kind of simple improvements can be used to improve NSW?

- Chain Swap:
  Sometimes, a pairwise swap might not be useful, in that case, a somewhat less obvious "Chain Swap" might be useful.

Note that a pairwise swap between A and B or B and C is non-improving. Turns out a chain swap is all we need...

Notation: For any allocation A, define a possibly bidirected graph G(A) as "k" arrows/edges from u to v if there are k goods that are owned by u (in A) and are valued (at 1) by v.

Note 2. A path P = $(u_1, u_2, u_3...u_k)$ in G(A) is simply a sequence of pairwise swaps. Observe that the utility of u_1 goes down by 1 and that of u_k goes up by 1. Everybody's utility is unchanged. So, as long as the starting and end point remains the same, any path in G(A) between u_1 and u_k has the same effect. So, instead of searching over exponentially many paths, **search over pair of agents**.

4.3.1 Lemma

Let A be a suboptimal allocation. Thus, there exists pair of agents u, v such that:

(a) v is reachable from u in G(A), meaning there is some path from u to v in G(A).

(b) Reallocating along any path (P) from u to v in G(A), leads to an allocation $A^{'}$:= A(P) that satisfies:

$$lnNSW(A^*) - lnNSW(A^{'}) \leq (1 - \frac{1}{m})(lnNSW(A^*) - lnNSW(A)$$

Part (a) of the lemma says there exists an improving path that could lead to an improvement in the NSW. Part (b) of the lemma quantifies that improvement, it means that we are getting closer to the optimal.

4.3.2 Binary Algorithm

i/p: A suboptimal allocation A
o/p: An allocation $A^{'}$

Initialize $A^{(0)} \leftarrow$ A

For i = 1 to 2m(n+1)ln(nm)

- Construct a graph $G(A^{i-1})$
- R = {(u,v): u $\rightarrow$ v in $G(A^{i-1})$}

- If R is Φ, output $A^{(1)} \leftarrow A^{i-1}$
- Otherwise, for each (u,v) $\in$ R
  + $A^{i-1}(u,v) \leftarrow$ allocation obtained by realloacating along some path u $\rightarrow$ v.
- If $max_{(u,v)\in R}$ NSW(A^{i-1}(u,v))>NSW(A^{i-1})
  + update $A^i \leftarrow$ arg max NSW(A^{i-1}(u,v))
    A^{i-1}(u,v): (u,v) $\in$ R
  + Otherwise, return $A^{'} \leftarrow A^{i-1}$

4.3.3 Proof of theorem: $A^{'}$ is Nash Optimal

Using Lemma: Let us analyse iteration i. From lemma, we have that:

$$lnNSW(A^*) - lnNSW(A^i) \leq (1 - \frac{1}{m})(lnNSW(A^*) - lnNSW(A^{i-1})$$

Repeated use of the bound give:

$$lnNSW(A^*) - lnNSW(A^i) \leq (1 - \frac{1}{m})^i(lnNSW(A^*) - lnNSW(A^0)$$

Note that $A^0, A^1, A^2 ... A^{i-1}$ must be suboptimal Since, for loop only execute for 2m(n+1)ln(mn) iteration, we have :

$$lnNSW(A^*) - lnNSW(A^{'}) \leq (1 - \frac{1}{m})^{2m(n+1)ln(mn)}(lnNSW(A^*) - lnNSW(A^0)$$

Since $(1 - \frac{1}{x})^x \leq \frac{1}{e}$for all $x > 0$

$$\leq \frac{1}{e^{2(n+1)ln(mn)}}(lnNSW(A^*) - lnNSW(A^0)$$

Since $NSW(A^0 \geq 1$, ln $NSW(A^0 \geq 0$

$$\leq \frac{1}{e^{2(n+1)ln(mn)}}(lnNSW(A^*)$$

$$\leq \frac{1}{mn^{2(n+1)}}(lnNSW(A^*)$$

Since because of binary valuation, $NSW(A^* \leq m$, we have:

$$\leq \frac{1}{mn^{2(n+1)}}(\text{ln m})$$

Since, ln m $\leq m$and n,m ≤ 1, we have:

$$\leq \frac{1}{n}.\frac{1}{(m^n)^2}$$

Since, ln $(1 + x) < x^2, for x \in (0, 0.5)$

$$< ln\left(1 + \frac{1}{m^n}\right)^{\frac{1}{n}}$$

Therefore, overall:

$$lnNSW(A^*) - lnNSW(A^{'}) < ln\left(1 + \frac{1}{m^n}\right)^{\frac{1}{n}}$$

Substituting NSW in the above equation we get,

$$\prod_{i \in N} V_i(A_i^*) < \left[\prod_{i \in N} V_i(A_i^{'})\right]\left(1 + \frac{1}{m^n}\right)$$

Since A^* is Nash optional,

$$\prod_{i \in N} V_i(A_i^{'}) \leq \prod_{i \in N} V_i(A_i^*)$$

Combining the above two equations, we get two integers that differ by strictly less than 1, both of them have to be the same. That means that $A^{'}$ **is Nash Optimal**

Fair Division	**Summer 2018**
Lecture 5: Open Problems	
Instructor: Rohit Vaish	*Scribes: Varad P.*

We will be discussing some open problems both in divisible and indivisible goods section:

5.1 Cake-Cutting

5.1.1 Query complexity of Envy-Freeness

n=2 Cut and Choose Algorithm.
No. of Queries $\rightarrow 2$

n=3 Selfridge-Conway Algorithm.
No. of Queries $\rightarrow 14$

n$\geq$4 Aziz-Mackenzie Algorithm.
No. of Queries $\rightarrow O\left(n^{n^{n^{n^n}}}\right)$
The Lower bound for given by Procaccia, IJCAI '09 $\rightarrow \Omega(n^2)$

Some questions :

Q. Thus, it can be seen that there is a huge gap in our understanding of the best and worst scenarios of computing EF divisions. So, how to **close the gap?**

Q. Special Cases of Valuation Functions:

- When a value density function is of degree d, then $\rightarrow O(n^2 d)$. This gives an interesting result for lower degree polynomial. **(Branzie IPL '15)**
- Piecewise Linear: As the name says, the valuation function can be linear within each part. For this, the complexity is $\rightarrow O(n^6 k \ln k)$, where k is the no. of breakpoints. **(Kurokawa et al. AAAI '14)**

Question is what will be the bounds for other classes of valuation?

Q. Power of simple algorithms for small n. A recent result by **(Amanatidis et al. SAGT '18)** improved the query complexity of Aziz-Mackenzie from 600 to less than 200. Question is, can it be extended for 5 or 6 or higher number of agents?

5.1.2 Query complexity fo approx EF with Connectedness

Recall that, the query complexity was defined by the number of questions asked to the agents such that all agents are EF. The answers were in the form of triangulation, meaning for a fine enough

traingulation, a lot of questions need to be asked.

$$\text{Worst Case: } \left(\frac{1}{\varepsilon}\right)^{n-2} \text{ No. of Queries } \textbf{(Deng et al. OR '12)}$$

For what valuations does the "path following" algorithm (based on Sperner's lemms) requires a small (say poly$\left(\text{n, } \frac{1}{\varepsilon}\right)$ number of queries?

$\rightarrow$ A useful assumption could be monotonicity. If piece 1 is preferred at say x_1, x_2, x_3, then it is also prreferred whenever $y_1 \geq x_1, y_2 \leq x_2, y_3 \leq x_3$.

Deng et al. have a result for n=3, what about general n?

5.1.3 Cake with both positive and negative value (MIXED CAKE)

So far, we have assumed the non-negativity, $V_i(I) \geq 0$ So, what will happen if someone dislikes a certain part of cake? Now, in

$$V_i(I) = \int_{x \in I} p_i(x)\text{dx, } p_i(x)\text{can be negative}$$

So, that means that a strictly bigger piece is not necessarily a better piece, meaning no Monotonicity. normalization need not hold. many of the algorithms seen so far will no longer work. e.g., The Selfridge-Conway algorithm.

Exercise: Check for 2 agents if "Cut and Choose" algorithm still gives EF allocation.

Known results;

(1) A connected EF allocation exists for n = 3 **(Segal-Halevi AAMAS '18)**

(2) A connected EF allocation exists for n = 4 or prime **(Meunier and Zubin '18)**

Q. What will be the existence and computatuon results of EF allocations?

5.2 Indivisible Goods

5.2.1 Nash Social Welfare

Recall that Nash optimal is EF1 and PO. We also saw that maximizing the Nash objective is NP–Hard. Therefore, computing Nash optimal is hard, but that doesn't mean that computing an allocation which is both EF1 and PO is also hard. Does that mean that we can compute EFF1 and PO in polynomial time? The answer is **Yes**.

(Barman et al. EC' 18) A psuedo-polynomial time algorithm for computing an EF1 + PO allocation. (A psuedo-polynomial time algorithm simply means that the running time of the algorithm not only depends on input but also on how big is the input is.) Running time of psuedo-polynomial : O(poly(m,n, V_{max})) where $V_{max} = max_{i,j} V_{i,j}$

If, $V_{i,j}$ is small and bounded by a constant, algorithm is in polynomial time. Even then, NSW is NP–Hard.

Q. Come up with a polynomial time algorithm for finding EF1 + PO allocation or prove that it is NP-Hard.

Q. What is the complexity of EF1 + PO?

5.2.2 Maximin Share (MMS)

Recall that MMS was defined in the terms of the thought experiment where every agent tries to partition the goods in a way to maximise its worst case utility. That is, $V_i(A_i) \geq \mu_i \; \forall \; i \in N$,

$$\mu_i = max_A \; min_k \; V_i(A_k)$$

– MMS need not always exist. **(Procaccia and Wang '14)**

Q. What is the best approx MMS that is guearateed to exist?
The answer was given by **(Ghodsi et al. EC '18)** which is $\frac{3}{4}$–MMS always exists. Note that this is the best result till date and not the maximum threshold.

Q. What is the best approx MMS that can be computed in polynomial time?
For any constant $\varepsilon > 0$, a $\left(\frac{3}{4} - \varepsilon\right)$−MMS can be computed in polynomial time. The proof for this was again given by **(Ghodsi et al. EC '18)**. Again, we don't know if it is the best.

Q. What the best approx MMS + PO that exists? And what will be its best computation in polynomial time?
The best result till date was given by **Barman et al. 2018**, which is $\frac{1}{n} - MMS + PO$ in polynomial time.

5.2.3 Stronger Fairness Notions

Recall that EF1 always exists.

(Caragiannis et al. '16) Another notion, EF_x, which stands for envy freeness up to the least positively valued good. An allocation A = $(A_1, A_2 ... A_n)$ is EF_x if:

$$\forall \text{ i,j} \in N, \forall \text{ g} \in A_j$$

such that

$$V_i(g) > 0 \text{ and } V_i(A_i) \geq V_i(A_j \backslash \{g\})$$

Note 1. EF $\Rightarrow$ EF_x $\Rightarrow$ EF1
Q. Does EF_x always exists?