Algorithms, Games, and Networks March 28, 2013

Lecture 18

Lecturer: Ariel Procaccia *Scribe: Hanzhang Hu*

1 Strategyproof Cake Cutting

All cake cutting algorithms we have discussed in previous lectures are not strategyproof. For example, in the case that we use cut and choose in the two player case, the first player (the cutter), knowing the second player's valuation, can possibly manipulate by not cutting the cake into two equal value pieces with his own valuation: let position a be the position to cut the cake into equal value pieces by valuation of the first player and b be that of the second player. Then as long as $a \neq b$, player 1 can cut between a and b. Then player 2 would choose the piece not containing b, and player 1 get a piece more than $1/2$, which is what he is supposed to get with cut-and-choose algorithm. In the case of Dubins-Spanier, a player can potentially gain by shouting later.

For the next result, we make the following assumption:

Assumption 1 *Agents report their full valuation functions, which are assumed to be concisely representable.*

Although determinsitic algorithms that are strategyproof (SP) and envy-free (EF) exists in some special cases, they are quite involved [Chen et al. 2010].

2 A Randomized Algorithm

Definition 2 *Cake pieces $X_1, ..., X_n$ is a **perfect partition** if they partition the cake such that $V_i(X_j) = 1/n$ for all i, j.*

Consider the following randomized algorithm for cake cutting:

1. Compute a perfect partition for the agents.
2. Draw a permutaion $\pi \in S_n$ uniformly at random.
3. Allocate agent i the piece $X_{\pi(i)}$.

Theorem 3 (Chen et al. 2010; Mossel and Tamuz 2010) *The randomized cake cutting algorithm above is strategy-proof in expectation and always procduces and envy-free allocation.*

Proof: (EF) By definition of a perfect partition, the allocation is envy-free since every agent receives a cake of value $1/n$.
(SP) For each agent i, regardless the valuation function i reports, the expected utility gain of i by the algorithm is

$$\sum_{j\in[n]} \frac{1}{n} V_i(X'_j) = \frac{1}{n} \sum_{j\in[n]} V_i(X'_j) = \frac{1}{n}$$

where X'_j are the pieces based on the valuation i reports. So in expectation, strategy doesn't increase gains. ■

Here are some intuitions (thus not proofs, theorems, etc.) for what we did here and why we want it in general: We can relax the strategy-proofness (make SP happen in expectation); but we need to keep envy-free determinstic. In particular, envy-free in expectation is not very useful, because, for example, the simple algorithm — randomly give the whole cake away to an agent — is envy-free in expectation. To make the algorithm more intuitively sound, we want EF to always hold, especially after the allocation is made; but since strategy-proofness only matters before the decision time, strategy-proofness in expectation is preventing manipulation at some level (though you could have different variance and so on).

In order for the randomized cake-cutting algorithm to work, we must have an algorithm to find a perfect partition. A theorem proves its existence, and further showed only a polynomial cuts are required, but this theorem is not constructive so it doesn't readily provide a cutting algorithm.

Theorem 4 (Alon, 1986) *A perfect partition always exists, needs polynomially many cuts*

However, if the valuation functions satisfy some special property, a perfect partition can be found. One example of the special cases is that the valuation function is a step function (i.e. a piecewise-constant function as demonstrated in Figure 1). The algorithm described as the following:

(1) Mark the beginning and end of each constant piece for each agent, then we have $t_0 = 0 < t_1 < ... < t_k = 1$ such that on each $[t_j, t_{j+1}]$, the valuation function is constant for all agent. Let X_j be piece $[t_j, t_{j+1}]$
(2) For each X_j, we equally (in length) partition it into n pieces, and label them with $1, 2, ..., n$. Let the union all pieces labeled with the same number l to be the piece l. Note that the length of the each piece l is $1/n$. Note that each agent i values each piece l as $\sum_{j=0}^{k} V_i(X_j)/n = \frac{1}{n}$, since each $[t_j, t_{j+1}]$ chunk contibutes $1/n$ its length — thus $V_i(X_j)/n$ value — to piece l.

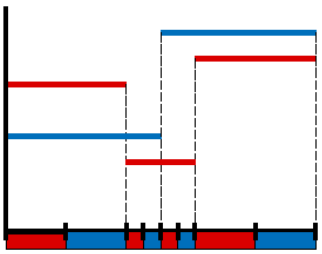
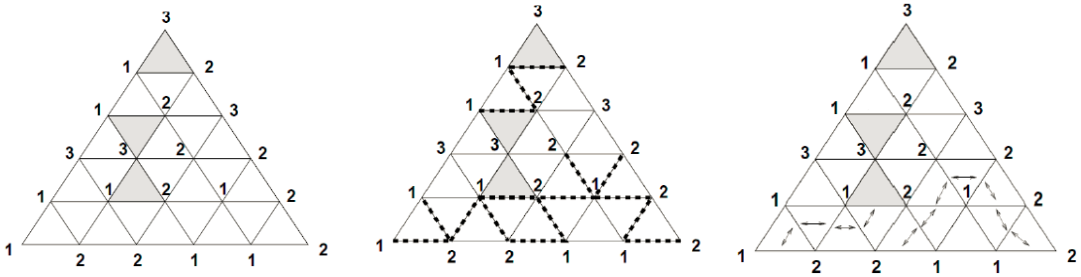


Figure 1: Step function example

3 Fair Rent Division (FRD)

We use Sperner's Lemma to prove existence of fair rent division, a simple setting of which is the following: Assume there are three agents, A,B,C, and three rooms $1, 2, 3$. The Goal is to divide the rent so that each person wants a different room. Sum of the prices for the three agents must be the total rent 1.

Sperner's Labeling: Let T be a triangle partioned into elementary triangles(e.g. shown in Figure 2a). A Sperner labeling is a vertex labeling of all the elementary triangles such that (1) the three vertices of T are labeled differently, (2) vertices on edge of T must be labeled the same as one of the end point of this edge.



(a) Sperner Label (b) Doors and Rooms (c) Walks

Figure 2: Sperner Labeling

Lemma 5 (Sperner's Lemma) *Any Sperner labeling contains at least one fully labeled elementary triangle (a triangle of label (1,2,3)).*

Proof: Let Doors be (1,2) edges (edges of label (1,2)). Let Rooms be elementary triangles. These are demonstrated in Figure 2b

1. Number of doors on the boundary of T must be odd, because (i) they have to be on

$(1, 2)$ edge of T; (ii) walking along the edge of T from 1 to 2, then there must be odd number of switches (doors) between 1 to 2 or 2 to 1.

2. Every room has at most 2 doors (easily checked).

3. A room has one door iff the room is (1,2,3)-labeled.

4. If we walk into the triangle through a door, and keep going into the next room if there is another door (we are not allowed to use the same door more than once). If we exit the triangle, we enter with a door never used if it exists. If we get stuck we have found a (1,2,3) room or out of the boundary and every door is used. The walks are demonstrated in Figure 2c

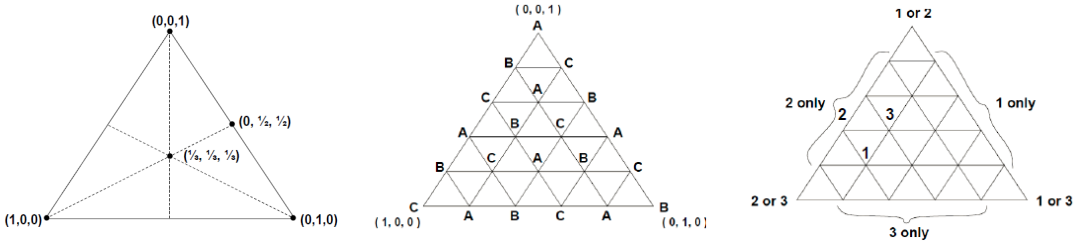
5. It is easy to show that we do not visit the same room twice.

6. Since there are odd number of doors on the boundary, we cannot get stuck outside, thus we must end in some (1,2,3) elementary triangle.

■

As a side note, Sperner's Lemma is closely connected to Brouwer fixed-point theorem, as the former can be used to prove the latter.

We now use Sperner's Lemma to prove existence of fair rent division for three agents. We represent the possible rent share space as a triangle shown in Figure 3a.



(a) Rent Representation (b) Partition (c) approximation

Figure 3: Fair Rent Sharing

Any point in the triangle is a convex combination of its three vertices. Let (x, y, z) be the ratios for the convex combination and the rent for each room is x, y, z, respectively. To approximate a fair rent division, we

1. Triangulate the big triangle T into small elementary triangles, such that each elementary triangle is labeled with ABC, as an example shown in Figure 3b

2. Ask the owner of each vertex (of elementary triangles) to tell its preference of room, assuming 0 rent is always the prefered. Label the vertices with the given room preference 1, 2, or 3.

3. By an augmented Sperner's lemma (since each vertex of T is of form (1 or 2), (2 or 3) or (3 or 1) instead of 1, 2, or 3), there exists an (1,2,3)-labeled elementary triangle. Then any rent share p represented within this elementary triangle is an approximation since each agent prefers a different room for some rent shares close to p. This approximation is shown in Figure 3c.

Some remarks: (1) Such a triangle is nothing but an approximately envy free allocation. (2) By making the triangulation finer, we can increase accuracy. (3) In the limit we obtain a completely envy free allocation. (4) Same techniques generalize to more agents [Su, 1999]

4 Dominant Resource Fairness (DRF)

We study the following setting: Allocating mutiple homogeneous resources to agents with different requirements. A typical real world running example is cloud computing, in which an agent may have requirement such as (2 CPU, 1 RAM) for each copy of its tasks, and the agent values the allocation based on the number of tasks he can run. So for example, with the above demands the agent is indifferent between allocation of (4 CPU,2 RAM) and (5 CPU,2 RAM), since in both case, we can run exactly 2 copies of the task. The state-of-the-art systems employ a single resource abstraction (bundling fixed amounts of each resource together). Here we study the case such that:

Assumption 6 *Agents have proportional demands for their resources*

We use the following mathematical model:

1. Set of agents $N = \{1, .., n\}$ and set of resources $R, |R| = m$.
2. Demands of agent i is $\vec{d_i} = (d_{i1}, ..., d_{im})$, $d_{ir} \in [0, 1]$, such that there exists r, $d_{ir} = 1$.
3. Allocation $\vec{A_i} = (A_{i1}, ..., A_{im})$, where A_{ir} is the **fraction** of resource r that is allocated to i.
4. Utility is induced by the number of copies (fraction allowed) of tasks we can run with the given allocation, which is

$$u_i(\vec{A_i}) = \min_{r \in R} A_{ir}/d_{ir}$$

We say the **dominant resource** of i is r, if $d_{ir} = 1$; and the **dominant share** of i is A_{ir}, if r is the dominant resource of i.

Mecahnism/Key-Idea: allocate proportionally to demands and equalize dominant shares. Formally, we seek the maximum x such that we can allocate to agent i an xd_{ir} fraction of resource r:

$$\max x s.t. \forall r \in R, \sum_{i \in N} x d_{ir} \leq 1$$

This can be reduced to

$$x = \frac{1}{\max_{r \in R} \sum_{i \in N} d_{ir}}$$

Example: $m = 2$, $N = 2$, $d_{11} = 1/2$, $d_{12} = 1$, $d_{21} = 1$, $d_{22} = 1/6$, then $x = \frac{1}{\frac{1}{2}+1} = 2/3$.

Axiomatic properties of DRF,

1. **Pareto optimality (PO)**: no one can be made better off without making at least one individual worse off.
2. **Envy-freeness (EF)**: $u_i(\vec{A}_i) \geq u_i(\vec{A}_j)$ for all $i, j \in N$.
   (Note that in cake cutting: EF $\Rightarrow$ Proportionality, but here it doesn't)
3. **Proportionality** (or sharing incentives, individual rationality): For all $i \in N$,

$$u_i(\vec{(}A_i)) \geq u_i((\frac{1}{n}, ..., \frac{1}{n}))$$

4. **Strategyproofness (SP)**: Faking $\vec{d}_i$ results in $\vec{A}'_i$, then $u_i(\vec{A}_i) \geq u_i(\vec{A}'_i)$.

Definition 7 *An allocation $\vec{A}_i$ is **non-wasteful** if there exists x such that $\vec{A}_{ir} = xd_{ir}$ for all r.*

Intuitively, non-wasteful means no resources are given to i but not used to create tasks. By definition it is easy to see: if $\vec{A}_i$ is non-wasteful and $u_i(\vec{A}_i) < u_i(\vec{A}'_i)$ then $\vec{A}_{ir} < \vec{A}'_{ir}$ for all r.

Theorem 8 (Ghodsi et al., 2011) *: DRF is PO, EF, proportional, and SP*

Proof:

1. PO: By definition of DRF, we increase x until we cannot do so any more, thus if any agent i increases utility, then his the dominant resource r must be overused (the total usage is more than 1, which is forbidden), unless someone decreases his utility.
2. EF: Let r be the dominant resource of i, then $\vec{A}_{ir} = x = xd_{ir} \geq xd_{jr}$, so switching is guaranteed to not do better.
3. Proportionality: For every resource r,

$$\sum_{i \in N} d_{ir} \leq n$$

   Thus we have

$$\max_{r \in R} \sum_{i \in N} d_{ir} \leq n$$

So that

$$x = \frac{1}{\max_{r \in R} \sum_{i \in N} d_{ir}} \geq \frac{1}{n}$$

4. SP: Let i manipulate, resulting in allocations $\vec{A}'_j = x' d'_{jr}$. (Assume everyone but i reports the same d_{ir} as before for each i, r).
   Case 1: $x' \leq x$, then i receives a smaller dominant share.
   Case 2: $x' > x$, Let r be the resource that is saturated by $\vec{A}$, which means

$$\sum_{j \in N} x d_{jr} = 1$$

$$\vec{A}_{ir} = 1 - \sum_{i \neq j} \vec{A}_{jr} = 1 - \sum_{i \neq j} x d_{jr} > 1 - \sum_{i \neq j} x' d_{jr} = 1 - \sum_{i \neq j} \vec{A}'_{jr} \geq \vec{A}'_{ir}$$

■

References

[1] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica. Dominant resource fairness: Fair allocation of multiple resource types. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation (NSDI)*, pages 24–37, 2011.

[2] F. E. Su. Rental harmony: Sperner's lemma in fair division. *American Mathematical Monthly*, 106(10):930–942, 1999.