Cake Cutting

—

Lecture 10

We introduce the concept of fair division in the context of dividing a heterogeneous, divisible good among players with different preferences. This is often visualized through the metaphor of *cake cutting*, where the goal is to find a fair allocation among participants.

1 Mathematical Model

We model the cake as the unit interval $[0, 1]$ and define the following:

- A set of players $N = \{1, \dots, n\}$.
- Pieces of cake $X_1, \dots, X_m \subseteq [0, 1]$, each of which is a finite union of disjoint subintervals. The pieces are themselves disjoint, and their union gives the whole cake: $X_i \cap X_j = \emptyset \; \forall i, j$; $X_1 \cup X_2 \cup \dots \cup X_m = [0, 1]$.
- Each player $i \in N$ has a non-negative, absolutely continuous valuation function $V_i(X)$ over pieces of cake, which satisfies the following properties:
  - **Additivity**: If X and Y are disjoint, then $V_i(X \cup Y) = V_i(X) + V_i(Y)$.
  - **Normalization**: $V_i([0, 1]) = 1$.
  - **Divisibility**: Given an interval I and $\lambda \in [0, 1]$, there exists a subinterval $I' \subseteq I$ such that $V_i(I') = \lambda V_i(I)$.

Different players might value different parts of the cake differently. For instance, we can imagine a cake which is chocolate for the interval $[0, 0.5)$ and vanilla for $[0.5, 1]$, and two players: one who values the chocolate part much more highly, and one who values the vanilla much more highly.

2 Fairness Properties

We aim to cut the cake into pieces and allocate pieces to players fairly. Define an allocation $A_1, \dots, A_n$ where player i receives the set of pieces A_i. Then we are interested in two fairness criteria:

Definition 1 (Proportionality)**.** An allocation is *proportional* if for all players $i \in N$, $V_i(A_i) \geq \frac{1}{n}$.
This ensures that each player receives at least their fair share according to their valuation.

Definition 2 (Envy-Freeness)**.** An allocation is *envy-free* if for all players $i, j \in N$, $V_i(A_i) \geq V_i(A_j)$.
This means that no player prefers another player's allocation over their own.

Does one criterion imply the other?

Theorem 1. *For two players, an allocation is proportional if and only if it is envy-free.*

Proof. Envy-freeness implies proportionality: Without loss of generality, if player 1 does not envy player 2, then $V_1(A_1) \geq V_1(A_2)$. Since $V_1(A_1) + V_1(A_2) = 1$, this implies $V_1(A_1) \geq \frac{1}{2}$.

Proportionality implies envy-freeness: If the allocation is proportional, then $V_1(A_1) \geq \frac{1}{2}$. Since $V_1(A_1) + V_1(A_2) = 1$, this implies $V_1(A_1) \geq V_1(A_2)$, so player 1 does not envy player 2. □

Theorem 2. *For three or more players, envy-freeness implies proportionality, but the converse is not true.*

Proof. Envy-freeness implies proportionality: Let A_i be what player i values most among $A_1, \dots, A_n$. Then, since $\sum_{j=1}^{n} V_i(A_j) = 1$, it must be that $V_i(A_i) \geq \frac{1}{n}$. Since the allocation is envy-free, player i must receive A_i, so the allocation is proportional.

Proportionality does not imply envy-freeness: Consider the case of three players. Let player 1's allocation A_1 be such that $V_1(A_1) = \frac{1}{3}$, but $V_1(A_2) = \frac{1}{3} + \epsilon$ and $V_1(A_3) = \frac{1}{3} - \epsilon$ for some small $\epsilon > 0$. Then the allocation is proportional, but player 1 envies player 2. □

3 Algorithmic Approaches

What are some algorithms that can achieve proportional or envy-free allocations? And how do we measure the complexity of these algorithms?

3.1 Measuring Complexity: The Robertson-Webb Query Model

We measure the complexity of cake-cutting algorithms using the Robertson-Webb model, which defines two types of operations:

- **Evaluation Query:** $\text{Eval}_i(x, y)$ returns $V_i([x, y])$. It asks "how much does player i value the piece $[x, y]$?"
- **Cut Query:** $\text{Cut}_i(x, \alpha)$ returns y such that $V_i([x, y]) = \alpha$. It asks "Please cut a piece starting at x that player i values at α."

3.2 Algorithms for Proportionality

3.2.1 Cut-and-Choose algorithm: Proportionality for two players

For two players, a simple algorithm is the *Cut-and-Choose algorithm*:

Cut-and-Choose algorithm (sketch)

1. Player 1 cuts the cake into two pieces they value equally.
2. Player 2 chooses the piece they prefer, and player 1 receives the other piece.

We claim this is envy-free, and hence also proportional.

Proof. Player 1 cuts the cake into two pieces they value equally, so they will not envy player 2. Player 2 then chooses the piece they value most, so they will not envy player 1. Thus, the allocation is envy-free. □

Under the Robertson-Webb model, the Cut-and-Choose algorithm requires two operations:

Cut-and-Choose algorithm (formal description)

1. Player 1 makes a cut: $\text{Cut}_1(0, \frac{1}{2}) = y$ such that the cut occurs at some $y \in [0, 1]$.
2. Player 2 values one of the pieces: $v = \text{Eval}_2(0, y)$.
3. If $v \geq \frac{1}{2}$, player 2 chooses the piece $[0, y]$. Otherwise, they choose the piece $[y, 1]$.

3.2.2 Dubins-Spanier algorithm: Proportionality for n players

A 'moving-knife' algorithm that ensures proportionality:

Dubins-Spanier algorithm (sketch)

1. A referee moves a knife continuously across the cake.
2. When any player finds the segment left of the knife worth at least $\frac{1}{n}$, they shout "stop". They receive that segment and are removed from the game. Ties are broken arbitrarily.
3. The process repeats with the remaining players until one player is left, who receives the remainder.

We claim this is proportional.

Proof. Any player who shouted "stop" received a segment worth at least $\frac{1}{n}$ to them, so they received their proportional share. The last player did not shout "stop" at all, or shouted "stop" at least once at the same time as some other player, so they must have valued the previous $n-1$ pieces each at at most $\frac{1}{n}$, and hence the remaining piece they receive is valued at least $\frac{1}{n}$. □

Under the Robertson-Webb model, the Dubins-Spanier algorithm requires $\Theta(n^2)$ operations:

Dubins-Spanier algorithm (formal description)

1. Let the set of remaining players be N.
2. While $|N| > 1$:
   (a) Denote the remainder of our cake as $[s, 1]$. At the first iteration, initialize $s = 0$.
   (b) For each player $i \in N$, player i makes a cut starting from s of a piece they value at $\frac{1}{|N|}$: $\text{Cut}_i(s, \frac{1}{|N|}) = y_i$.
   (c) Let $i^* = \arg\min_{i \in N} y_i$. Player i^* receives the piece $[s, y_{i^*})$ and is removed: $N = N \setminus \{i^*\}$.
   (d) Update $s = y_{i^*}$.
3. The last player remaining receives the piece $[s, 1]$.

The number of cuts is $n + (n-1) + \cdots + 2$, so the total complexity is $\Theta(n^2)$.

3.2.3 Even-Paz algorithm: Optimal proportionality for n players

The Even-Paz algorithm achieves proportionality in $O(n \log n)$ operations in the Robertson-Webb model.

Even-Paz algorithm (formal description)

For ease of exposition, we describe the algorithm for $n = 2^k$ players:

1. Denote the remainder of our cake as $[s, t]$. At the first iteration, initialize $s = 0$ and $t = 1$.
2. If $n = 1$, the player receives $[s, t]$. Otherwise:
3. Each player i makes a mark z_i that divides $[s, t]$ into two equal-valued pieces: let $v_i = \text{Eval}_i(s, t)$, then $\text{Cut}_i(s, \frac{1}{2}v_i) = z_i$.
4. Let z^* be the $n/2$th mark from the left.
5. Recurse on the left half $[s, z^*]$ with the $n/2$ players who marked $z_i \leq z^*$, and the right half $(z^*, t]$ with the remaining $n/2$ players.

The recursion depth is $O(\log n)$ since we halve the number of players at each step, and each step requires $O(n)$ operations. Thus, the total complexity is $O(n \log n)$.

We prove that this is proportional:

Proof. We can prove by induction that after k steps, the players sharing a piece of cake value it at least at $1/2^k$. For $k = 0$, this is true because the piece is initially the whole cake. For the induction step, if the piece of cake at step k is $[x, y]$, at step $k+1$, players are sharing a piece valued at least $V_i([x, y])/2$. Therefore, if a player's value was at least $1/2^k$, now it is at least $1/2^{k+1}$.

The process ends after $\log n$ steps with a single player being allocated a piece. We conclude that the value of each player is at least $1/2^{\log n} = 1/n$. □

It is also known that:

Theorem 3. *Any proportional cake-cutting algorithm requires $\Omega(n \log n)$ operations in the Robertson-Webb model.*

Hence, the Even-Paz algorithm is optimal under the Robertson-Webb model.

3.3 Algorithms for Envy-Freeness

In general, envy-free cake-cutting is algorithmically more challenging than proportional cake-cutting.

3.3.1 Selfridge-Conway algorithm: Envy-freeness for three players

This algorithm gets an envy-free allocation for three players.

Selfridge-Conway algorithm (sketch)

The algorithm takes place in three stages.

Stage 0:

1. Player 1 cuts the cake into three pieces they value equally.
2. Player 2 trims the largest piece such that the two largest pieces are now equal in their eyes. There are now four pieces: the two most valuable equal pieces (one of which was trimmed), the trimming, and the remaining original piece.
3. Let Cake 2 be the trimming, and Cake 1 be the other three pieces.

Stage 1: We now divide Cake 1 first.

1. Player 3 chooses their highest-valued piece in Cake 1.
2. If Player 3 did not choose the trimmed piece, Player 2 is allocated the trimmed piece. Otherwise, if Player 3 chose the trimmed piece, Player 2 chooses their highest-valued piece out of the two remaining ones.
3. The remaining piece is allocated to Player 1.

At this point, either Player 2 or 3 received the trimmed piece. Among players 2 and 3, let T be the player that received the trimmed piece, and T' be the other player.

Stage 2: We finally divide Cake 2.

1. Player T' divides Cake 2 into three pieces they value equally.
2. Players $T, 1$, then T' choose one of the three pieces each in that order.

We claim that this is envy-free.

Proof. For each player, since valuations are additive, if we show that they are envy-free on Cake 1 and Cake 2 then they are envy-free overall. We will do this for each player.

Player T'. This is clearly true on Cake 2 since they divided it equally. On Cake 1, if T' is Player 3, they got to choose their highest-valued piece out of all 3 original pieces, so they are envy-free. If T' is Player 2, recall that they trimmed the two largest pieces such that they were equal in Stage 0, and they received one of those largest pieces in Stage 1. Hence, they are envy-free, since they received their highest-valued piece overall.

Player T. On Cake 2, they chose their highest-valued piece first, so they are envy-free. On Cake 1, the argument is the same as above: Player 3 and 2 are both envy-free, so Player T is certainly envy-free.

Player 1. On Cake 1, since they did not receive the trimmed piece, then they received a piece certainly worth 1/3 as they cut the cake equally in Stage 0. On Cake 2, Player 1 does not envy T' because they choose before T'. The only concern left is T. But notice that T received the trimmed piece from Cake 1, which can only be worth at most 1/3 if T then receives the entire Cake 2! In this way, Player 1 has an *irrevocable advantage* over T in the allocation of Cake 2, so they do not envy T. □

3.3.2 Complexity of Envy-Free Algorithms

It has been shown that an envy-free cake-cutting algorithm always exists.

Theorem 4 (Brams and Taylor 1995)**.** *There exists an envy-free cake-cutting algorithm for any number of players under the Robertson-Webb model.*

However, by changing the valuation function of the players, the complexity of the algorithm described by Brams and Taylor can be made arbitrarily high, i.e. cannot be bounded as a function of n.

It was later shown that a bound does exist, but it is extremely high.

Theorem 5 (Aziz and Mackenzie 2016)**.** *A bounded envy-free algorithm exists for any n, but its complexity is extremely high:* $O\left(n^{n^{n^{n^{n^n}}}}\right)$.

A lower bound of $\Omega(n^2)$ was also shown for envy-free cake-cutting algorithms.

Theorem 6 (Procaccia 2009)**.** *Any envy-free cake-cutting algorithm requires $\Omega(n^2)$ operations in the Robertson-Webb model.*