Algorithmic Game Theory - Fair Division *Martin Hoefer, Giovanna Varricchio*

The Cake-Cutting Problem

Summary: In the cake-cutting problem we are given a divisible resource, a.k.a. a *cake*, which has to be split among n agents. Each agent has a preference over different pieces, and the goal is to split the cake among the agents in a *fair* manner.
We first concentrate on two criteria for fairness: Proportionality and envy-freeness. We discuss their existence and computational complexity.
We then look into further desirable properties other than fairness; namely, Pareto and Nash optimality. Finally, we briefly discuss equitability and its computation.

Resources:

- *Handbook of Computational Social Choice*, F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A.D. Procaccia, 2016, Cambridge University Press (Chapther 13, Cake Cutting Algorithms).
- *Tutorial on Recent Advances in Fair Resource Allocation*, Rupert Freeman and Nisarg Shah `https://www.cs.toronto.edu/~nisarg/papers/Fair-Division-Tutorial.pdf`
- Further readings in the references.

1 Setting

We are given a divisible resource, a *cake* C. It is abstractly represented by the interval $[0, 1]$. Moreover, there is a set $\mathcal{N} = \{1, \dots, n\}$ of n agents. A *piece of cake* is any union of disjoint intervals in C. A piece of cake is called *connected* if it is an interval $[x, y] \subseteq C$. The goal is to partition the cake into n pieces and assign each of them to a distinct agent, which means that the cake has to be fully allocated.

Definition 1 (Allocation)**.** *An* allocation $\mathcal{A} = (A_1, \dots, A_n)$ *is a partition of C into pieces, each of which is assigned to a unique agent. For each $i \in \mathcal{N}$, we denote by A_i the piece received by agent i in the allocation $\mathcal{A}$. An allocation must be complete, that is, $\cup_{i \in \mathcal{N}} A_i = C$.*

An allocation $\mathcal{A}$ is called *simple* if each agent receives a connected piece of C, i.e. $\mathcal{A}$ is obtained by cutting C at $n - 1$ points. The goal is to compute a *fair* allocation of the cake to the agents. We will later define what fair means. First, we need to introduce how agents evaluate a piece of cake.

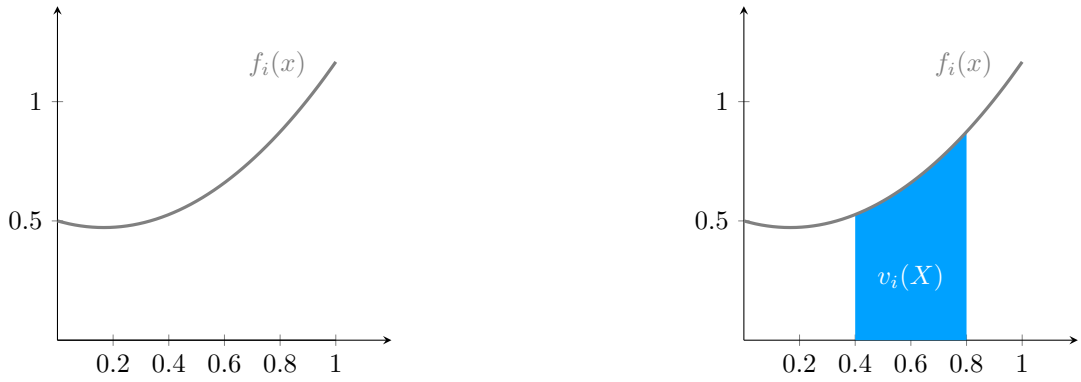
1.1 Agent Valuations

We assume each agent can evaluate any subset of the cake. For this reason, each agent $i \in \mathcal{N}$ is endowed of an integrable density function $f_i : C \to \mathbb{R}_{\geq 0}$. Therefore, given any agent $i \in \mathcal{N}$ and any piece of cake $X \subseteq C$, the value of agent i for X is given by

$$v_i(X) = \int_{x \in X} f_i(x) dx \,.$$

We can assume, without loss of generality, that $\int_{x \in [0,1]} f_i(x) dx = 1$, for each agent $i \in \mathcal{N}$.

Example 1. *In Figure 1 we see an example of a density function of an agent i: $f_i(x) = x^2 - \frac{1}{3}x + \frac{1}{2}$ and of the value of the piece $X = [\frac{2}{5}, \frac{4}{5}]$.*



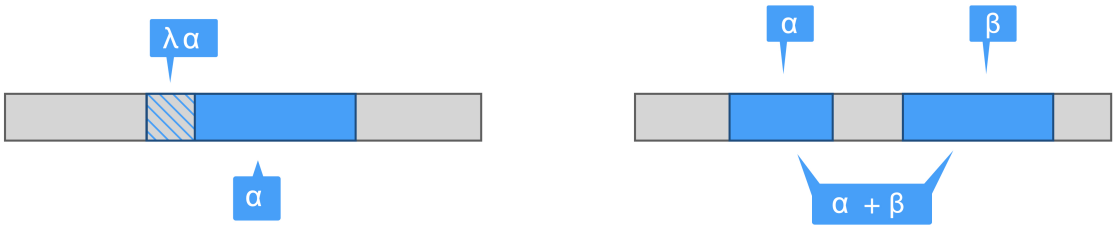
(a) Density function of agent i: $f_i(x) = x^2 - \frac{1}{3}x + \frac{1}{2}$.

(b) Value of piece $X = [\frac{2}{5}, \frac{4}{5}]$.

Figure 1: Example of a density function f_i and valuation of a piece X.

Properties. Throughout our treatment we assume that the every valuation function v_i, for each $i \in \mathcal{N}$, satisfies the following properties:

- *normalized*, i.e. $v_i(C) = 1$;
- *divisible*, i.e. $\forall \lambda \in [0,1]$ and $I = [x,y] \subseteq C$ there exists $z \in I$ such that $v_i([x,z]) = \lambda \cdot v_i([x,y])$;
- *additive*, i.e. for any pair of disjoints intervals $I, I' \subset C$, $v_i(I \cup I') = v_i(I) + v_i(I')$;
- *non-negative*, i.e. for each $I \subseteq C$, $v_i(I) \geq 0$.



(a) Divisible.

(b) Additive.

Figure 2: Properties of valuations.

Whenever we refer to the value of a piece in a given allocation, we mean the *value for the owner*!

1.2 Query Complexity

Suppose the density functions of the agents are given as input. They could require infinitely many bits for representation. In contrast, we study a more reasonable way to access the valuations – by *oracle access*. We assume to have a so-called oracle for each agent i that can answer some meaningful questions about v_i.

Robertson-Webb model. For each agent i, we assume there exists an oracle that can answer the following two queries:

- $Eval_i(x,y)$ returns $v_i([x,y])$;
- $Cut_i(x,\alpha)$ returns y such that $v_i([x,y]) = \alpha$.

As a consequence, we evaluate the performance of an algorithm by its *query complexity*, that is, the number of queries required during its execution.

2 Fairness Criteria

In this section, we introduce some fundamental fairness criteria and discuss their relations.

2.1 Definitions

We distinguish between threshold-based and comparison-based criteria. A threshold-based criterion requires that each agent receives at least some pre-defined value for her piece of cake; a comparison-based criteria determines the satisfaction of an agent by comparing the piece she receives with the pieces received by the others.
A very natural threshold value is the proportional share: The value of the entire cake divided by the number of agents. Formally, the *proportional share of agent i* is given by $\mathsf{PS}_i = \frac{v_i(C)}{n}$. We assumed valuations to be normalized, which implies that each agent has a proportional share of $\frac{1}{n}$.

Definition 2 (Proportionality)**.** *An allocation $\mathcal{A}$ is called* proportional (PROP) *if each agent receives at least her proportional share, that is, $\forall i \in \mathcal{N}$ it holds*

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

Towards comparison-based criteria, a very intuitive one is *envy-freeness*. It requires that no agent envies any other agent.

Definition 3 (Envy-freeness)**.** *An allocation $\mathcal{A}$ is called* envy-free (EF) *if for each $i, j \in \mathcal{N}$ it holds*

$$v_i(A_i) \geq v_i(A_j) \ .$$

Example 2. *Let us consider a cake-cutting instance with three agents having valuations as depicted in Figure 3. The allocation $A_1 = [0, 1/6]$, $A_2 = [1/6, 5/6]$, and $A_3 = [5/6, 1]$ is* PROP *and* EF*.*

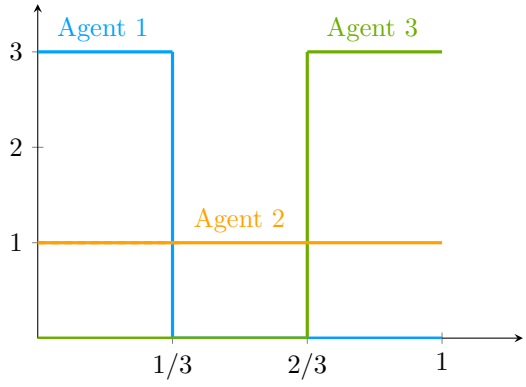


Figure 3: Example instance with three agents.

2.2 Implications

Proposition 1. *Any* EF *allocation is always* PROP*.*

Proof. Let us show the statement for an arbitrary agent $i \in \mathcal{N}$. Since $\mathcal{A}$ is EF, for each $j \in \mathcal{N}$ it holds

$$v_i(A_i) \geq v_i(A_j) \ .$$

Summing up over all $j \in \mathcal{N}$ we get

$$n \cdot v_i(A_i) \geq \sum_{j \in \mathcal{N}} v_i(A_j) \overset{(1)}{=} v_i\left(A_1 \cup \ldots \cup A_n\right) \overset{(2)}{=} v_i(C) \overset{(3)}{=} 1, \quad \text{and, thus,} \quad v_i(A_i) \geq \frac{1}{n} \ .$$

Note that (1) holds because of the additivity of valuations and the fact that the pieces in A are non-overlapping, (2) because the cake is completely allocated, and (3) because of normalization. □

Proposition 2. *For $n = 2$, any* PROP *allocation is always* EF*.*

Proof. Roughly speaking, if an agent receives a piece she values more than half, from her perspective, the other agent is receiving less than half.
Formally, given $i \in \{1, 2\}$, by proportionality $v_i(A_i) \geq \frac{1}{2}$ and therefore $v_i(A_j) = v_i(C \setminus A_i) \leq \frac{1}{2} \leq v_i(A_i)$. □

3 Existence and Computation

Let us discuss the existence and computation of PROP and EF allocations. We start by considering the simplest scenario with two agents.

3.1 Two Agents: The Cut-and-Choose Protocol

For two agents, where PROP $\Leftrightarrow$ EF, there exists a simple but nonetheless interesting protocol for achieving both PROP and EF, the so-called CutAndChoose. It works as follows:

Cut: Agent 1 cuts the cake into two pieces of value $\frac{1}{2}$ for her.
Choose: Agent 2 selects the piece she prefers the most and agent 1 receives the other.

Theorem 3. *The* CutAndChoose *protocol outputs a proportional (and hence envy-free) allocation.*

Proof. Agent 1 receives a piece of value $\frac{1}{2}$ for him. Agent 2 select the most preferred piece whose has necessarily value $\geq \frac{1}{2}$. □

And the query complexity? Only two queries! We only ask $y \leftarrow Cut_1(0, \frac{1}{2})$ and $Eval_2(0, y)$.

3.2 Proportionality

We now turn our attention to the computation of a proportional allocation for n agents. Proportionality is the easiest criterion to achieve, and several protocols have been proposed.

3.2.1 The Dubins-Spainer Protocol – Proportionality with $O(n^2)$ Queries

We introduce the Dubins-Spainer protocol also known as the MovingKnife.
Intuitively, starting from the left-most position 0, we pretend to move a knife along our cake to the right. Consider the first a point at which the unassigned piece to the left has value of the proportional share to some remaining agent i. At that point, we cut the cake and assign the piece to i. Then, i is disregarded as well as the assigned piece of cake. We continue from the current position, move the knife to the right, and find again the first position where the next agent gets her proportional share.
Formally, let $x_k \in [0, 1]$ be the position of the knife. The MovingKnife protocol proceeds as follows:

- During the ℓ-th iteration of the algorithm, the knife is positioned at $x_k = y_{\ell-1}$, where $y_0 = 0$. Then, x_k is slowly (and contiguously) moved to the right.

- The agents are allowed to shout as soon as the piece $[y_{\ell-1}, x_k]$ reaches their proportional share. Hence, whenever there is an agent i such that $v_i([y_{\ell-1}, x_k]) \geq \frac{1}{n}$ agent i shouts; if there exists more than one such agent we break ties arbitrarily.
- As soon as one agent shouts, that agent receives the piece of cake $[y_{\ell-1}, x_k]$ and leaves the protocol, i.e. $\mathcal{N} \leftarrow \mathcal{N} \setminus \{i_\ell\}$. Set $y_\ell = x_k$. The process repeats on the remaining cake with the remaining agents. Note, however, that each agent continues to strive for a proportional share of $1/n$, where n remains the original number of agents!
- Finally, when $|\mathcal{N}|= 1$, assign $[y_\ell, 1]$ to the unique remaining agent and terminate.

We described the MovingKnife protocol as a contiguous process over the cake. How to implement such a process in the Robertson-Webb model? At round ℓ, we use $Cut_i(y_{\ell-1}, \frac{1}{n})$ for all $i \in \mathcal{N}$. Roughly speaking, we ask all the agents at which point they would shout. By taking the minimum of all these positions, we know the first shouter and the place to cut.

Example 3. *We consider an example run of* MovingKnife *protocol on the instance depicted in Figure 3. At the beginning, no agent owns any piece of cake. The protocol starts from the position $x_k = 0$ and asks every agent where to cut to obtain her proportional share, i.e. $Cut_i(0, 1/3)$ for each $i \in \{1, 2, 3\}$. Since $Cut_i(0, 1/3) = 1/9$, $Cut_2(0, 1/3) = 1/3$, and $Cut_3(0, 1/3) = 7/9$, agent 1 decleares the smallest value, gets selected, and receives the piece $[0, 1/9]$. Agents 2 and 3 remain in the process, and the knife is positioned in $x_k = 1/9$. Since $Cut_2(1/9, 1/3) = 4/9$ and $Cut_3(1/9, 1/3) = 7/9$, agent 2 receives $[1/9, 4/9]$ and agent 3 receives $[4/9, 1]$.*

Theorem 4. *The* MovingKnife *protocol outputs a simple and proportional allocation.*

Proof. Simple: The algorithm assigns every agent a contiguous piece of cake.
Proportional: The algorithm assigns every agent (but the last one) a piece with value of the proportional share. We need to show that (1) it never consumes the entire cake before every agent receives one piece, and (2) the remaining of the cake is always enough to guarantee the proportional share to the remaining agents. We prove the following claim: At the beginning of any round $\ell \in [n-1]$, let $C' \subseteq C$ be the remaining of the cake; if agent i has not received a piece of cake yet, then $v_i(C') \geq 1/n$.
Let $C_h = [y_{h-1}, y_h]$ be the piece of cake assigned at the h-th round. Since i has not received the cake, no such piece has value more than $1/n$. Therefore, $v_i(C') = v_i(C) - \sum_{h<\ell} v_i(C_h) \geq 1 - (\ell - 1)/n \geq 1/n$.
Since the above argument applies also to the agent receiving the last piece, proportionality holds also for her. □

And query complexity? At each step, we ask the remaining agents where we should cut for her proportional share. Hence, we need $\sum_{i=1}^{n-1} n - i + 1 = \Theta(n^2)$ queries.
Can we do better?

3.2.2 The Even-Paz Protocol – Proportionality with $O(n \log n)$ Queries

We present a recursive protocol. For simplicity, we assume $n = 2^k$ for some integer k.

Input: An interval $[x, y]$ and n agents
- If $n = 1$, then give $[x, y]$ to the agent and terminate;
- else each agent i computes z_i such that $v_i([x, z_i]) = \frac{1}{2} v_i([x, y])$.
- Select z^* the $n/2$-th value z_i from the left in $[x, y]$;
- Recruse on $[x, z^*]$ with the left $n/2$ agents, and on $[z^*, y]$ with the right $n/2$ agents.

Theorem 5. *The Even-Paz protocol returns a proportional allocation.*

Proof idea. Invariant for each recursive call: There is enough cake for the involved players to get at least their proportional share. By induction, at the last step, only one agent is considered and therefore she receives her proportional share. □

Query complexity: The protocol runs in $O(\log n)$ rounds; in every round each agent replies to exactly one query $\rightarrow O(n \log n)$.

Theorem 6 (Edmonds and Pruhs, 2006)**.** *Any proportional protocol needs $\Omega(n \log n)$ queries in the Robertson-Webb model.*

Therefore the Even-Paz protocol is asymptotically optimal!

3.3 Envy-Freeness

We have seen that the CutAndChoose protocol provides an EF allocation for two agents. Around 1960, Selfridge and Conway (independently) constructed the same algorithm for three agents.

3.3.1 Envy-Freeness for Three Agents: The Selfridge-Conway Protocol

Initialization:

- Agent 1 divides the cake into three equally-valued pieces $X_1, X_2, X_3 : v_1(X_1) = v_1(X_2) = v_1(X_3) = 1/3$.
- Agent 2 trims the most valuable piece according to v_2 to create a tie for the most valuable. For example, we assume w.l.o.g. that X_1 is the most valuable piece for agent 2.
  - If $v_2(X_1) > v_2(X_2) \geq v_2(X_3)$, agent 2 removes $X' \subseteq X_1$ such that $v_2(X_1 \setminus X') = v_2(X_2)$. We call the three pieces – one of which is trimmed – *cake 1* ($\{X_1 \setminus X'\} \cup X_2 \cup X_3$), and we call the trimmings (X') *cake 2*.
  - If $v_2(X_1) = v_2(X_2)$, *cake 2* is empty.

Division of cake 1:

- Agent 3 chooses one of the three pieces of cake 1;
- If agent 3 chose the trimmed piece ($X_1 \setminus X'$), agent 2 chooses between the two other pieces of cake 1. Otherwise, agent 2 receives the trimmed piece. We call refer to the agent $i \in \{2, 3\}$ that received the trimmed piece by agent T. The other agent is agent $\overline{T}$;
- Agent 1 receives the remaining piece of cake 1.

Division of cake 2:

- Agent $\overline{T}$ divides cake 2 into three equally-valued pieces;
- Agents $T, 1, \overline{T}$ select a piece of cake 2 each, in that order.

For an example, apply the procedure on the instance depicted in Figure 3.

Theorem 7. *The Selfridge-Conway protocol outputs an* EF *allocation for three agents.*

Proof. Let us denote cakes 1 and 2 by C_1 and C_2, respectively.
Observation: The division of C_1 is EF. Indeed, agent 1 always receives a piece of value $1/3$, and no other piece has a higher value; agent 2 always receives one of the most two preferred (and equally liked) pieces; agent 3 selects the most preferred piece. Therefore, if $C_2 = \emptyset$, the theorem follows.
Otherwise, let us consider the final allocation of C (that is after the division of C_1 and $C_2 \neq \emptyset$): Agent $\overline{T}$, who is splitting the cake, is the one who receives the remaining piece of C_2; anyway, she will be EF in the final allocation, because she likes the three pieces of C_2 equally. The agent T selecting first piece is also EF in the final allocation. It remains to show agent 1 is EF.

Agent 1 does not envy $\overline{T}$, since 1 selects the piece of C_2 before $\overline{T}$. But does agent 1 envy agent T in the final division of the whole cake C?
On the one hand, T is the agent who received $X_1 \setminus X'$, and $\{X_1 \setminus X'\} \cup C_2 = X_1$. Therefore, no matter which piece of C_2 agent T receives, agent 1 will value the final piece of T at most $1/3$ (because of the initialization of the algorithm X_1 has a value of $1/3$ for agent 1). On the other hand, during the allocation of C_1, agent 1 received a piece of value $1/3$ and by adding another piece from C_2 cannot decrease the value attained by 1, showing that 1 does not envy T. □

3.3.2 Envy-Freeness for any Number of Agents

What about the existence of EF allocations for general n? We will see in the next section that they always exist. What about computation?

Theorem 8 (Aziz and Mackenzie, 2016)**.** *There exists a finite protocol for computing an* EF *allocation with a query complexity of* $O\left(n^{n^{n^{n^n}}}\right)$.

Theorem 9 (Procaccia 2009)**.** *Any protocol for finding an envy-free allocation requires* $\Omega(n^2)$ *queries in the Robertson-Webb model.*

There still is a large gap between these lower and upper bounds.

4 Efficiency

To obtain fairness, we might produce extremely inefficient partitions of a cake, as the following example shows.

Example 4. *Assume there are two agents* $\mathcal{N} = \{1, 2\}$*,* $f_1 = U[0, 1/2]$ *and* $f_2 = U[1/2, 1]$*, where* $U[x, y]$ *is the uniform distribution over* $[x, y]$*. Consider the partition where agent* 1 *receives* $[0, 1/4] \cup [3/4, 1]$ *and agent* 2 *receives* $[1/4, 3/4]$*. Such an allocation is* EF *and therefore* PROP*; notice that both agents receive their proportional share. However, there is a much better allocation which is still* EF *but both agents get a utility of* 1*; namely, it is sufficient to cut the cake at* $x = 1/2$ *and assign the left side o agent* 1*, and the right side to agent* 2*.*

This example might look artificial; it is not hard to verify that all the protocols we provided so far are inefficient in the sense we are going to define in the next subsection.

4.1 Pareto Optimality

The most prominent definition of efficiency is *Pareto optimality.* Roughly speaking, an allocation is called Pareto optimal (or Pareto efficient) if there exists no other allocation where each agent is not decreasing and at least one agent is strictly increasing her utility. Formally:

Definition 4 (Pareto optimal allocation)**.** *Given a pair of allocations* $\mathcal{A}, \mathcal{B}$ *of the cake* C*,* $\mathcal{B}$ Pareto dominates $\mathcal{A}$*, if for each* $i \in \mathcal{N}$*,* $v_i(B_i) \geq v_i(A_i)$ *and at least one of the inequalities is strict.*
An allocation $\mathcal{A}$ *of* C *is* Pareto optimal *(PO) if there is no allocation* $\mathcal{B}$ *that Pareto dominates* $\mathcal{A}$*.*

Notice that a Pareto optimal allocation always exists. Pareto optimality is not per se an interesting property, we will strive for fair allocations which also satisfy Pareto optimality.

4.2 Nash Social Welfare – Fair and Efficient

A way to obtain Pareto optimal allocations is to resort to an optimization problem related to a particular welfare function. It turns out that by maximizing the Nash social welfare function, we obtain an extremely fair allocation and also guarantee Pareto optimality.

Definition 5. *Given an allocation $\mathcal{A}$ of a cake C, the* Nash social welfare *(NSW) of $\mathcal{A}$ is defined as follows:*

$$NSW(\mathcal{A}) = \left(\prod_{i \in \mathcal{N}} v_i(A_i) \right)^{\frac{1}{n}}.$$

Clearly, an allocation maximizing the Nash social welfare always exists. We call such an allocation *NSW-optimal*.

Example 5. *Consider the instance depicted in Figure 3 and the allocation $A_1 = [0, 1/6]$, $A_2 = [1/6, 5/6]$, and $A_3 = [5/6, 1]$ has a Nash welfare of $\left(\frac{1}{2} \cdot \frac{4}{6} \cdot \frac{1}{2}\right)^{\frac{1}{3}} = \left(\frac{1}{6}\right)^{\frac{1}{3}}$. This allocation is* EF *but is not Nash optimal. Consider the allocation where the first third of the cake is assigned to agent 1, the second third to agent 2, and the remaining piece to agent 3. Such an allocation has a Nash welfare of $\left(1 \cdot \frac{1}{3} \cdot 1\right)^{\frac{1}{3}} = \left(\frac{1}{3}\right)^{\frac{1}{3}}$*

Proposition 10. *Every NSW-optimal $\mathcal{A}$ is PO.*

Proof. By contradiction, if an allocation $\mathcal{B}$ Pareto dominates $\mathcal{A}$ then $\mathcal{B}$ has strictly better NSW. □

Notice that maximum Nash social welfare is scale-invariant: if we multiply the valuation v_i of any agent i by any positive number $\alpha_i > 0$, then this does not change the optimal allocations w.r.t. NSW.
An interesting aspect of the NSW is that it is a good (and fair) trade-off between egalitarian and utilitarian social welfare objectives.

- Utilitarian social welfare: Sum of agents' utilities, i.e. $USW(\mathcal{A}) = \sum_i v_i(A_i)$. The USW only focuses on overall happiness without caring about each individual.
- Egalitarian social welfare: Minimum of agents' utilities, i.e. $ESW(\mathcal{A}) = \min_i v_i(A_i)$. The ESW cares about a specific individual without caring about the overall happiness.

5 Existence of Envy-Free Allocations

In this section, we discuss the existence of EF allocations. The main result is the following:

Theorem 11. *Every NSW-optimal $\mathcal{A}$ is* EF.

Let's first build an intuition for why this is true.

Example 6. *Consider an instance with two agents. Agent 1 has density function $U[0, 1/2]$ while agent 2 density function $U[0, 1]$. Suppose we have the allocation $\mathcal{A}$ with $A_1 = [0, 1/6]$ and $A_2 = [1/6, 1]$. Now $\mathcal{A}$ is not* EF *and has a Nash welfare of $\sqrt{\frac{1}{3} \cdot \frac{5}{6}} = \sqrt{\frac{5}{18}}$.*
To prove our theorem, we show that whenever an allocation is not EF *it is possible to move a piece of cake from the bundle of the envied to the bundle of the envious agent while increasing the Nash welfare.*
In the specific example, 1 envies 2 and we can move $[1/6, 1/4]$ from A_2 to A_1. After this move, we obtain a new allocation $\mathcal{A}'$, and the new Nash welfare is $\sqrt{\frac{1}{2} \cdot \frac{3}{4}} = \sqrt{\frac{3}{8}}$. Thus, we increased the Nash welfare.

Proof. Let $\mathcal{A}$ be a NSW-optimal allocation. Let us assume by contradiction that it is not EF. Then there exist $i, j \in \mathcal{N}$ such that $v_i(A_j) > v_i(A_i)$. To reach a contradiction, we show there exists a piece of cake $Z \subset A_j$ such that we can strictly improve the NSW by moving Z from A_j to A_i.
Let j split A_j into k equally liked pieces for her, for some $k \in \mathbb{N}$, i.e. each of them has value $\frac{1}{k} \cdot v_j(A_j)$. Let i choose the most preferable piece, let's call this piece Z. Then the following conditions hold:

$$v_j(Z) = \frac{1}{k} \cdot v_j(A_j) \qquad \text{and} \qquad v_i(Z) \geq \frac{1}{k} \cdot v_i(A_j).$$

Let us call $\mathcal{A}'$ the allocation obtained by moving Z from A_j to A_i.
Notice that only i and j have a different utility in $\mathcal{A}$ and $\mathcal{A}'$, respectively. Therefore, to understand which allocation is better it suffices to compare the product of the utilities of i and j in the two allocations.
Formally:

$$\frac{\mathrm{NSW}(\mathcal{A}')}{\mathrm{NSW}(\mathcal{A})} > 1 \quad \Leftrightarrow \quad \left(\frac{\prod_{k\in\mathcal{N}} v_k(A'_k)}{\prod_{k\in\mathcal{N}} v_k(A_k)}\right)^{\frac{1}{n}} > 1 \quad \Leftrightarrow \quad \left(\frac{v_i(A'_i)\cdot v_j(A'_j)}{v_i(A_i)\cdot v_j(A_j)}\right)^{\frac{1}{n}} > 1 \quad \Leftrightarrow \quad \frac{v_i(A'_i)\cdot v_j(A'_j)}{v_i(A_i)\cdot v_j(A_j)} > 1$$

If $v_i(A'_i) \cdot v_j(A'_j) > v_i(A_i) \cdot v_j(A_j)$, we reach the desired contradiction.
We have

$$\begin{aligned} v_i(A'_i) \cdot v_j(A'_j) &= (v_i(A_i) + v_i(Z)) \cdot (v_i(A_j) - v_j(Z)) \\ &\geq \left(v_i(A_i) + \frac{1}{k} \cdot v_i(A_j)\right)\left(1 - \frac{1}{k}\right) \cdot v_j(A_j) \\ &= v_i(A_i) \cdot v_j(A_j) - \frac{1}{k} \cdot v_i(A_i) \cdot v_j(A_j) + \frac{1}{k}\left(1 - \frac{1}{k}\right) \cdot v_j(A_j) \cdot v_i(A_j)\,. \end{aligned}$$

Therefore if $-\frac{1}{k} \cdot v_i(A_i) \cdot v_j(A_j) + \frac{1}{k}\left(1 - \frac{1}{k}\right) \cdot v_j(A_j) \cdot v_i(A_j) > 0$ we get our contradiction. Such an inequality holds iff $-v_i(A_i) + \left(1 - \frac{1}{k}\right) \cdot v_i(A_j) > 0$ if and only if $k > \frac{v_i(A_j)}{v_i(A_j) - v_i(A_i)}$.
To conclude, it is possible to find a piece of cake Z such that by moving Z from A_j to A_i, the NSW strictly improves. This is a contradiction to the optimality of $\mathcal{A}$. The theorem follows. □

Since any NSW-optimal allocation is PO we also have the following:

Proposition 12. *An allocation that is simultaneously* EF *and PO always exists.*

What about computation? It is not necessarily easy and depends on the type of valuations we are considering.

6 Equitability

To conclude our discussion of fair cake cutting, we consider another comparison-based notion that takes into account not only pairs of pieces but also the valuations of the owners.

Definition 6 (Equitability)**.** *An allocation $\mathcal{A}$ is called* equitable (EQ) *if for each $i, j \in \mathcal{N}$ it holds*

$$v_i(A_i) = v_j(A_j)\ .$$

Proposition 13. *Equitability is incomparable with both proportionality and envy-freeness.*

Proof. Consider an instance with two agents where agent 1 has a positive value only for the first half of the cake and agent 2 values only the second half. The allocation where $A_1 = [\frac{1}{2}, 1]$ and $A_2[0, \frac{1}{2}]$. Such an allocation is neither proportional nor envy-free.
It is also possible to provide an allocation that is envy-free, and therefore proportional, but not equitable. □

Theorem 14. *An equitable allocation always exists.*

We show the claim for only two agents.

Proof. We set $g_1(x) = v_1([0,x])$, which is an non-decreasing and contiguous function with $g_1(0) = 0$ and $g_1(1) = 1$, and $g_2(x) = v_2([x,1])$, which is a non-increasing and contiguous function with $g_2(0) = 1$ and $g_2(1) = 0$. There exists x^* such that $g_1(x^*) = g_2(x^*)$. Therefore, the allocation $A_1 = [0, x^*]$ and $A_2 = [x^*, 1]$ is equitable. □

Problem: There is no way to implement an equitable protocol with a finite number of queries in the Robertson-Webb model!

For two players, to find x^* we can use a bisection algorithm, which possibly requires an infinite number of refinements steps to find the location of x^*.
However, if we perform a bounded amount of such refinements we can still obtain a good approximation to equability leading to the next theorem.
The bisection algorithm takes as input agents valuations and some value ε and proceeds as follows:

- Set $a \leftarrow Cut_1(0, \frac{1}{2})$, $b \leftarrow Cut_2(0, \frac{1}{2})$
- If $a = b$ output a and terminate
- If $b < a$ exchange agents identities, therefore, in what follows we always have $a < b$
- Initialize $j \leftarrow 2$, $c \leftarrow 0$
- While $(\frac{1}{2})^j \geq \varepsilon$ do
  - $x \leftarrow Cut_1(a, (\frac{1}{2})^j)$, $y \leftarrow Cut_2(c, (\frac{1}{2})^j)$
  - if $x = y$ return x and terminate
  - if $x < y$ then $a \leftarrow x$ and $b \leftarrow y$
  - if $x > y$ then $c \leftarrow y$
  - $j \leftarrow j + 1$
- output $(a + b)/2$

An allocation $\mathcal{A}$ is ε-EQ if $|v_i(A_i) - v_j(A_j)| \leq \varepsilon$ for each i, j.

Theorem 15 (Cechlárová and Pillárová, 2012)**.** *Using the bisection algorithm, for two agents, it is possible to find an ε-EQ allocation with $O(\log(1/\varepsilon))$ queries.*

The above theorem can be extended to any number of agents.

Theorem 16 (Cechlárová and Pillárová, 2012)**.** *It is possible to find an ε-EQ allocation with $O(n \cdot \log(1/\varepsilon))$ queries.*

References

[1] Haris Aziz and Simon Mackenzie. A discrete and bounded envy-free cake cutting protocol for any number of agents. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 416–427. IEEE, 2016.

[2] Katarína Cechlárová and Eva Pillárová. A near equitable 2-person cake cutting algorithm. *Optimization*, 61(11):1321–1330, 2012.

[3] Katarína Cechlárová and Eva Pillárová. On the computability of equitable divisions. *Discrete Optimization*, 9(4):249–257, 2012.

[4] Lester E Dubins and Edwin H Spanier. How to cut a cake fairly. *The American Mathematical Monthly*, 68(1P1):1–17, 1961.

[5] Jeff Edmonds and Kirk Pruhs. Cake cutting really is not a piece of cake. In *SODA*, volume 6, pages 271–278, 2006.

[6] Shimon Even and Azaria Paz. A note on cake cutting. *Discrete Applied Mathematics*, 7(3):285–296, 1984.

[7] Ariel D Procaccia. Thou shalt covet thy neighbor's cake. In *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

[8] Erel Segal-Halevi and Balázs R Sziklai. Monotonicity and competitive equilibrium in cake-cutting. *Economic Theory*, 68(2):363–401, 2019.