A Note on Locally Fair Cake-Cutting

Jiale Chen

Abstract

We studied the local fairness of the cake-cutting problem. We proposed an $\Omega(n^2)$ lower bound for the local envy-freeness under star graphs with constraints and proposed a partial locally envy-free protocol for 4-Path which can be easily extended to 5-Path.

1 Background

The cake-cutting problem is about dividing a heterogeneous and divisible good among multiple agents with different preferences. And we care about the fairness of the allocation result. In mathematical terms, there is a set of agent $N = \{1, 2, \ldots, n\}$ and a cake $[0, 1]$. Each agent i has a valuation function V_i for any subinterval in $[0, 1]$. The valuation function is additive, divisible, non-negative, and normalized. We will allocate a piece of cake X_i to agent i, where X_i is a finite union of disjoint intervals. Our goal is to design an algorithm that allocates a piece of cake X_i to each agent $i \in N$ such that $X_1, X_2, \ldots, X_n$ forms a partition of $[0, 1]$. Also, we require the allocation to satisfy some kind of fairness. Two important fairness criteria are

1. envy-freeness: for each agent, the value of her own piece of cake must be at least as large as her value for any other's piece of cake, i.e.,
$$\forall\ i, j, V_i(X_i) \geq V_i(X_j).$$
2. proportionality: each of n agents must receive a piece of cake with at least $\frac{1}{n}$ value, i.e.,
$$\forall\ i, V_i(X_i) \geq \frac{1}{n}.$$

What the algorithm can do is described by the Robertson-Webb model:

1. $\text{eval}_i(x_1, x_2)$: returns $V_i([x_i, x_2])$.
2. $\text{cut}_i(x_1, \alpha)$: returns $x_2 \in [0, 1]$ such that $V_i([x_1, x_2]) = \alpha$.

And we are interested in the number of queries needed to derive a fair allocation in the cake-cutting problem.

1.1 Proportional cake cutting

The Even–Paz protocol [1], based on recursively halving the cake and the group of agents, requires only $O(n \log n)$ actions. In each stage, we ask every agent's half value point and then sort those points from left to right. We choose the middle point and split the cake into two halves. The left cake and those agents with points on the left become the first subproblem. The right cake and those agents with points on the right become the second subproblem. Recursively solve those subproblems and then we get a proportional allocation.

Surprisingly, the recursive algorithm is proved to be the optimal solution up to a constant factor [2]. The proof of the lower bound itself is also interesting. First, they notice that in proportional cake cutting, the interaction between agents is small. They describe a 'Thin-Rich' game that involves only one agent. The algorithm is expected to find a piece of cake for this agent with width at most $\frac{2}{n}$ and value at least $\frac{1}{n}$. They find that if this game has a lower bound $T(n)$ then the proportional cake cutting has a lower bound $\Omega(nT(n))$. To bound $T(n)$ they introduce the definition value trees to help to construct the value distribution adversarially.

In detail, the value tree is a balanced 3-ary rooted tree, with $\frac{2}{n}$ leaves. Each node represents an interval which is the union of the intervals represented by its sons. The value of the node is the value of the corresponding interval. The construction of the distribution is done by assigning the weight on each edge of the tree. The edge weight represents the ratio of the son node's value to the parent node's value. Since the 'thin-rich' must intersects with a high-density leaf, the goal of the adversary is to find a way to assign edge weight such that heavy weight won't occur frequently along a path. Finally, it can be proved that after k queries, any root-leaf path contains at most $2k$ heavy edges.

1.2 Envy-free cake cutting

Aziz and Mackenzie propose the first discrete and bounded protocol for envy-freeness [3]. The key idea of their protocol is to recursively use the Core Protocol to find a partial envy-free allocation and then to ensure that one set of agents will dominate others, so the problem can be reduced to a subproblem with fewer agents. Here the relationship that agent i dominates agent j means that even if all the remaining cake is allocated to j, agent i won't envy him.

The best lower bound for this problem is derived by Procaccia [4]. The idea is also to deal with the agents separately and design the valuation function. Procaccia regards both query operations w.r.t. agent i as adding two points on the $[0, 1]$ and splitting the interval into small subintervals. Standing in agent i's view, the adversary should minimize the value i has and maximize the value other agents have. That is, if the allocation of i does not contain an entire small segment, i has no value for this part. But if agent j's allocation has a non-trivial intersection with a segment, j can have the value of the entire segments in i's view. When the algorithm outputs, the allocation must be envy-free even in the adversarial setting. A necessary condition for it is that i's value for her allocation should be at least the value of her favorite segment. Otherwise, some agent j may contain some part of this segment which leads to envy. To derive the concrete lower bound, Procaccia chooses the "uniform-like case" to link the value and the length of the segments.

The proof can be slightly simplified that, in this "uniform-like case", in agent i's view, i's length $\geq$ i's value $\geq$ j's value $\geq$ j's length. So every agent must have a $\frac{1}{n}$ length of $[0, 1]$. In i's view, i's value is at most $\frac{1}{n}$ while others' value is at least $\frac{1}{n}$. Thus the equal sign must be taken, which means that each agent's allocation should be exactly some segments in i's segment set and have a total length of $\frac{1}{n}$. That requires $\Omega(n)$ points in i's segment sets. Totally, we have an $\Omega(n^2)$ lower bound.

1.3 Locally fair cake cutting

The cake cutting problem can be extended to a graph-based model. Suppose there is a graph structure of agents $G(V, E)$. Each node in V represents an agent. Different from the previous setting, the agent here only cares about the allocation results in her neighborhood. Thus, the fairness criteria become a local property.

1. For local proportionality, we are considering

$$\forall\ i \in V, \quad V_i(X_i) \geq \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} V_i(X_j).$$

2. For local envy-freeness, we are considering

$$\forall\ i \in V, j \in \mathcal{N}(i), \quad V_i(X_i) \geq V_i(X_j).$$

Bei et al. [5] design a protocol for the locally proportional cake cutting. There are two parts in their protocol: creating a dominant point and then spreading the dominance. The dominant here is defined as even if all the remaining cake is allocated to her neighbors, she remains proportional. They heavily rely on the Core Protocol in Aziz and Mackenzie's protocol [3]. More concretely, they choose a cutter point and run the Core Protocol to create enough insignificant pieces while maintaining the global envy-freeness. Afterward, they swap at least one insignificant piece to the neighborhood of the cutter point while maintaining local proportionality. As soon as this is done, the cutter point becomes a dominant point. There are two main properties of the Core Protocol they use. Global envy-freeness is helpful and somehow necessary for maintaining local proportionality while swapping. The definition of the insignificant piece helps create dominance.

Our goal is to explore the locally fair cake-cutting problem. Tucker-Foltz proved that on some bipartite graphs, there is an $\Omega(n^2)$ lower bound for algorithms that output a locally proportional allocation [6]. Using the same method in [4], we can easily prove that for any graph with m edges, there is an $\Omega(m)$ lower bound for algorithms that output a locally envy-free protocol. We observe that the number of edges plays an important role in those two proofs. A Natural question is where we can achieve a $\Omega(n^2)$ lower bound in a graph with $o(n^2)$ edges. Therefore we first explore the star graph.

2 Star Graph

2.1 An $O(n^2)$ locally envy-free protocol

The TreeCore protocol in [5] obtains a locally envy-free partial allocation on trees. Implementing it on a star graph can obtain a complete locally envy-free allocation. We present its simplified version on the star graph, the StarCore Protocol.

Algorithm 1 StarCore Protocol

```
Require: A star graph $S$ with center $r$, $|S| = n$, and the cake $[0, 1]$.
Ensure: A locally envy-free allocation on $S$.
1: Center r cuts the cake into $n$ equally preferred pieces in her own measure.
2: for all agent $u \in S \setminus \{r\}$ in an arbitrarily order do
3:     Agent $u$ takes the piece that she values the highest in the remaining pieces.
4: The last piece is allocated to $r$.
```

2.2 An $\Omega(n^2)$ lower bound for local envy-freeness in the discrete case

Given the protocol above, consider the case that the center is fixed as the cutter point, each agent gets one piece and the algorithm is allowed to only query the value of each part. Formally, we call the following problem the discrete case of a star graph.

- There are n items and n people.
- Each agent i has a value v_{ij} on each item j.
- An allocation is called valid iff each agent is allocated one item and for any agent $i \in \{2, 3, \dots, n\}$, she will not envy agent 1. Formally, a valid allocation is a permutation σ such that

$$\forall\ 2 \le i \le n, v_{i\sigma_i} \ge v_{i\sigma_1}.$$

- The algorithm doesn't know $\{v_{ij}\}$ but it can query for one of them each time.

Note that a valid allocation always exists by running the round-robin algorithm in the order of $2, 3, \dots, n, 1$. The goal is to prove that any algorithm that always outputs a valid allocation needs $\Omega(n^2)$ queries. As an adversary, we are going to assign values to $\{v_{ij}\}$ such that

- For any agent $2 \leq i \leq n$ and items $j \neq k$ $v_{ij} \neq v_{ik}$
- For any two agents $2 \leq i \neq j \leq n$, their favorite items $f_i \neq f_j$.
- For the remaining item r^* that no agent $2 \leq i \leq n$ values the highest, it's their second favorite.

If $\{v_{ij}\}$ satisfies the above conditions, then there is only one valid allocation, $\sigma_1 = r^*$ and for the rest agents, $\sigma_i = f_i$. So intuitively, the algorithm must find the favorite piece of each agent.

Theorem 1. *Any algorithm cannot output a valid allocation with less than $\frac{1}{2}(n-1)(n-2)$ queries.*

Proof Without loss of generality, we can fix $r^* = 1$ and tell the algorithm about it. Also, for each query, we will return the rank in that agent's view rather than the absolute value. These would both help the algorithm. I'm going to prove that the algorithm still needs $\frac{1}{2}(n-1)(n-2)$ queries to find f_is.

Consider a bipartite graph G with $V(G) = L \cup R$, where each node in L represents an agent and each node in R represents an item. Since the algorithm already knows that $r^* = 1$, we can assume that $|L| = |R| = n - 1$. Each edge between agent i and item j represents that it's possible for $f_i = j$. Initially, each edge between L and R exists. A valid allocation exists if and only if there is a perfect matching, while an algorithm could successfully output a valid allocation if and only if there is a unique perfect matching.

Each time the algorithm queries v_{ij},

1. If v_{ij} is queried before, return the same answer.
2. If there is a perfect matching after deleting the edge (i, j), return a rank larger than 1, i.e., $f_i \neq j$, and delete the edge (i, j) from G.
3. Otherwise, return a rank 1, i.e., $f_i = j$.

It can be seen that in each step, the algorithm can delete at most one edge and it's guaranteed that a perfect matching exists. Thus we only need to bound the number of edges in a graph with unique perfect matching. Consider 2 edges in the perfect matching and the corresponding 4 vertices. For these 4 vertices, we can add at most one more edge, otherwise, there will be another perfect matching. Thus the graph can have no more than $(n-1) + \binom{n-1}{2} = \frac{1}{2}n(n-1)$ edges. Therefore, there should be at least $\frac{1}{2}(n-1)(n-2)$ queries. □

Ghalme, Huang, and Rathi consider a similar discrete case without the constraint on the algorithm, and give a $\Omega(n^2)$ lower bound [7]. The discrete case mainly corresponds to the situation in the StarCore protocol after step 1. And we believe it somehow describes the difficulty of the envy-freeness towards a single agent.

3 4-path

We then move towards a polynomial protocol of local proportionality on the tree. As a start point, we were surprised that the 4-path case is already non-trivial. For convenience, we denote the points in path agents 1,2,3,4 in order. We propose an algorithm for the 4-path case below which can be easily extended to the 5-path case.

R	the remaining cake
$\mathcal{A}_i$	the ith allocation
$\mathcal{A}_{ij}$	agent j's part in the ith allocation
V_i	the valuation function of agent j
Π_{ijk}	$V_j(\mathcal{A}_{ij}) - V_j(\mathcal{A}_{ik})$

Table 1: Notations

Algorithm 2 4-Path-Core

```
Require: A 4-path $P$, and a cake.
Ensure: A locally envy-free allocation on $P$, and the remaining cake.
1: Agent 2 cuts the cake into 4 equally preferred pieces in his own measure.
2: Agent 1 chooses his favorite piece.
3: Agent 3 chooses two remaining pieces that she values the highest. She chops the higher ones so
   that she values them equally. Let agent 4 chooses his favorite one. The other piece is allocated to
   agent 3.
4: The last piece is allocated to 2.
5: return the allocation and the chopped part.
```

Algorithm 3 4-Path Protocol

```
Require: A 4-path $P$, and a cake.
Ensure: A locally envy-free allocation with agent 2 dominating agent 3, and the remaining cake $R$.
1: Let $R$ be the entire cake initially.
2: for t from 1 to 2 do
3:     $\mathcal{A}_t, R \leftarrow$ 4-Path-Core$(P, R)$
4:     if $R = \emptyset$ or Agent 3 receives an insignificant piece then
5:         return the combination of $\mathcal{A}_t$, and $R$.
6: Let $S = \{\mathcal{A}_1, \mathcal{A}_2\}$.
7: Let $\mathcal{A}_p \in S$ be the allocation such that $\Pi_{p43}$ is the higher. $S = S \setminus \{\mathcal{A}_p\}$.
8: Let $\mathcal{A}_q \in S$ be the remaining allocation in $S$. Swap $\mathcal{A}_{q3}$ and $\mathcal{A}_{q4}$.
9: return the combination of modified $\mathcal{A}_t$ and $R$.
```

Theorem 2. *The 4-Path Protocol gives an envy-free partial allocation with agent 2 dominating agent 3.*

Proof There is only one chopped piece in each allocation of 4-Path-Core and can only be given to agent 3 or 4. The only case we need to be careful of is that agent 4 always gets it. Intuitively, as long as we can swap the chopped piece in some allocation from agent 4 to 3 while maintaining the locally envy-freeness, we're done.

First of all, agent 1 never envies other agents. Secondly, in 4-Path-Core, we can see that agent 3 values the pieces for agents 3 and 4 the same. Therefore, the swap will not make agent 3 envious. So we only need to deal with the envy-freeness of agent 4.

To do so, we should first see that envy-freeness leads to $\Pi_{p43} \geq \Pi_{q43} \geq 0$. If we swap the pieces of agents 3 and 4 in $\mathcal{A}_q$, the total advantage of agent 4 will be $\Pi_{p43} - \Pi_{q43} \geq 0$, which means that agent 4 will not envy agent 3. □

Remark Regarding the Dominance Creating and Spreading framework in [5], they are trying to move an insignificant piece to one neighbor of the cutter. The pieces are exchanged along the shortest path. To maintain local proportionality of the points outside of the path, envy-freeness from the outside points to the points in the path is critical. That means a local envy-free partial allocation is not enough for this method. So although there is a partial envy-free protocol on the tree, it cannot be the driving horse of the whole protocol. We need a partial protocol with the stronger property as our core. One sufficient condition in the tree case is that any point doesn't envy its parent's parent and the descendant nodes of its parent. Further, a sufficient condition in a general graph would be a similar condition on a layer diagram generated by the shortest path algorithm. This might be a future direction, though the complexity of the whole algorithm will probably remain exponential.

Acknowledgement

We would like to thank Ariel Procaccia and Jamie Tucker-Foltz for their helpful discussions during this work.

References

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

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

[3] 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.

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

[5] Xiaohui Bei, Xiaoming Sun, Hao Wu, Jialin Zhang, Zhijie Zhang, and Wei Zi. Cake cutting on graphs: A discrete and bounded proportional protocol, 2019.

[6] Jamie Tucker-Foltz. Thou shalt covet the average of thy neighbors' cakes. *arXiv preprint arXiv:2106.11178*, 2021.

[7] Ganesh Ghalme, Xin Huang, and Nidhi Rathi. Envy-free cake cutting with graph constraints. *arXiv preprint arXiv:2205.12559*, 2022.