Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

Optimal Proportional Cake Cutting with Connected Pieces

Xiaohui Bei
Institute for Interdisciplinary Information Sciences
Tsinghua University
Beijing, China

Ning Chen
Division of Mathematical Sciences
Nanyang Technological University
Singapore

Xia Hua
Division of Mathematical Sciences
Nanyang Technological University
Singapore

Biaoshuai Tao
Division of Mathematical Sciences
Nanyang Technological University
Singapore

Endong Yang
Division of Mathematical Sciences
Nanyang Technological University
Singapore

Abstract

We consider the classic cake cutting problem where one allocates a divisible cake to n participating agents. Among all valid divisions, fairness and efficiency (a.k.a. social welfare) are the most critical criteria to satisfy and optimize, respectively. We study computational complexity of computing an efficiency optimal division given the conditions that the allocation satisfies proportional fairness and assigns each agent a connected piece. For linear valuation functions, we give a polynomial time approximation scheme to compute an efficiency optimal allocation. On the other hand, we show that the problem is NP-hard to approximate within a factor of $\Omega\left(\frac{1}{\sqrt{n}}\right)$ for general piecewise constant functions, and is NP-hard to compute for normalized functions.

Introduction

Resource allocation is a fundamental problem studied extensively in economics, social science, as well as in computer science, due to its vast applications. A central question by its nature is *how to allocate a given set of resources to some potential consumers*. For different application domains, there are different requirements to be achieved, or different factors to be balanced. In particular, individual participants may want to receive as many resources as possible, while a centralized authority may have some global objectives. What criteria should be considered to select an allocation?

Fairness, which, balances individual participants' competitive interests, is one of the most important solution conditions considered in practice. There are two commonly used notions that capture fairness in resource allocation: envy-freeness and proportionality. In an *envy-free* allocation, no individual envies the allocation of any other individual; in a *proportional* allocation, every individual is allocated at least an average share with respect to his own measure. It is easy to see that an envy-free allocation is always proportional. However, in many applications an envy-free allocation either does not exist (e.g., determine accepted papers of a conference) or is algorithmically intractable (e.g., cake cutting with large constant number of consumers). Thus, we will focus on proportionality in the current paper.

Another important measure of an allocation is that of efficiency (a.k.a. social welfare). In contrast with fairness, efficiency evaluates the performance of the solution from a global point of view. Specifically, the *efficiency* of an allocation is defined to be the sum of the values of all participants with respect to their own allocation. Efficiency is therefore a concrete quantity that can be optimized.

Fairness and efficiency together are among the most critical factors considered in many applications. An ideal solution is to have both (either satisfied or optimized), but this may not necessarily be achievable. In this paper, we will consider efficiency optimization given the required (proportional) fairness condition. Specifically, we will address the following question:

What is the computational complexity of computing an efficiency optimal allocation while ensuring fairness?

We will study this question in an important resource allocation model: cake cutting, where the resource is a continuous cake and can be divided into pieces and allocated to n individuals. Cake cutting has been a central problem studied in computer science and social science; see, e.g., (Robertson and Webb 1998). An important feature of cake cutting is that the cake is divisible and individuals may have different values for different portions of the cake. The problem is therefore represented by a valuation measure function for each of the individuals.

Our interest is to find a division of the cake that maximizes efficiency given fairness. This question was first considered in (Cohler et al. 2011) where the authors gave a linear program to compute an envy-free allocation with the maximum possible efficiency for piecewise constant measure functions. Their algorithm can be generalized easily to a polynomial time approximation scheme for any functions that can be represented concisely[1], and immediately implies the same result for proportional fairness.

[1] We can divide the cake into small pieces and approximate each small piece by a constant; this leads to piecewise constant functions.

The solution generated by the linear program, however, involves a large number of cuts and the allocation to each participant consists of many separate "tiny" pieces, which is not desirable in some applications (e.g., division of a territory). We therefore focus on divisions with exactly $n-1$ cuts, that is, everyone obtains a connected region. Note that with $n-1$ cuts, a proportional allocation always exists, e.g., the well known moving knife procedure (Dubins and Spanier 1934) generates a fair allocation with $n-1$ cuts for n participants.

We will study two types of measure functions: linear and piecewise constant. For linear functions, we give a polynomial time approximation scheme (PTAS) that computes an $(1-\epsilon)$-approximate efficiency optimal allocation. That is, for any constant $\epsilon > 0$, the solution given by the scheme is within a factor of $(1-\epsilon)$ to the optimum. Note that this is the best we can hope for as for linear functions, an optimal solution may involve irrational numbers. Our algorithm is based on a critical ingredient of determining an optimal order of participants to cut the cake: While the intuition is that the optimal order would be according to the slopes of the linear functions, we show that this is actually not the case and give a precise characterization of the order.

For piecewise constant functions, finding an efficiency optimal allocation turns out to be much harder. While for any given fixed order of participants, an exactly optimal solution can still be found using a dynamic programming approach, the problem is NP-hard to solve without knowing an optimal order, even for normalized valuations. For general piecewise constant functions, we show that it is even NP-hard to find an allocation whose efficiency is within a factor of $\Omega\big(\frac{1}{\sqrt{n}}\big)$ to the optimum. (We refer to (Papadimitriou 1993) as a classic textbook for computational complexity and approximations.)

Related Work

Maximizing efficiency in cake cutting with fairness has been studied extensively in the past. (Reijnierse and Potters 1998) studied the computation of a Pareto-efficient envy-free allocation. (Brams et al. 2012) studied Pareto efficiency of optimal envy-free or equitable allocations. (Caragiannis et al. 2009) introduced the concept of price of fairness to characterize the loss of efficiency at a cost of fairness. (Aumann and Dombb 2010) followed the notion of price of fairness and focused on divisions with only connected pieces. In contrast with our work which is to find exact allocations, the work (Caragiannis et al. 2009; Aumann and Dombb 2010) addressed on the worst case ratio between the optimum efficiency without and with fairness conditions. Note that their benchmark is optimum efficiency without fairness, whereas in our efficient algorithm design the benchmark is optimum efficiency with fairness. The work closest to ours was by (Cohler et al. 2011), where the authors considered efficiency optimization of several valuation functions given the envy-free fairness condition (without requiring connected pieces). Independent to our work, (Aumann, Dombb, and Hassidim 2012) studied egalitarian *or* utilitarian welfare optimization with $n-1$ cuts; however, we consider efficiency optimization *with* the fairness condition.

Preliminaries

There are n agents to share a cake, which is represented by the interval $[0,1]$; we will call '0' the left side and '1' the right side. Each agent i has a density function $f_i(\cdot) : [0,1] \to \mathbb{R}^+ \cup \{0\}$. The valuation of agent i for the interval $[a,b] \subseteq [0,1]$ is denoted by $v_i(a,b) = \int_a^b f_i(x)\,dx$. Assume without loss of generality that $v_i(0,1) > 0$ (otherwise, we can simply remove such agents). For any $S \subseteq [0,1]$, let $v_i(S)$ denote the valuation of agent i for S, i.e., the sum of the valuations over disjoint intervals in S.

An *allocation* (i.e., division) of the cake is a partition into n disjoint subsets, denoted by $A = (A_1, A_2, \ldots, A_n)$. We require allocations to be fair, captured by *proportionality*: $v_i(A_i) \geq \frac{1}{n} v_i(0,1)$ for any i, i.e., every agent gets at least an average share with respect to its own measure. Our objective is to find a proportional allocation that maximizes *efficiency*, defined as $\sum_i v_i(A_i)$. We will restrict our attention on exactly $n-1$ cuts. That is, we can divide the cake by only $n-1$ cuts; thus, each agent will obtain a connected interval.

Our interest is to study efficiency maximization from an algorithmic viewpoint. We say an algorithm is an α-approximation if the efficiency of its solution is always within a factor of α to the optimum efficiency, both conditioned on proportionality. In other words, the compared benchmark of an algorithm is the maximum efficiency among all proportional allocations with $n-1$ cuts.

In this paper we will consider two special families of density functions that can be represented concisely: linear valuations and piecewise constant valuations.

Linear Functions

In this section we will consider linear density functions, i.e., each $f_i(\cdot)$ can be represented as $f_i(x) = a_i \cdot x + b_i$, where a_i, b_i are two given numbers. We assume that $f_i(x) \geq 0$ for any $x \in [0,1]$; this implies in particular that $b_i \geq 0$. Further, we have $v_i(0,1) = \frac{a_i}{2} + b_i > 0$. Linear function is one of the most fundamental and well studied valuation functions. It reflects gradually increasing or decreasing interests of agents for the resource.

We will give a polynomial time approximation scheme (PTAS) to compute an efficiency optimal allocation. First we will characterize the order of allocations to the agents in an optimal solution. Having this characterization, we will then use a dynamic programming algorithm to compute a nearly optimal solution.

Optimal Order

As there are in total $n-1$ cuts, an allocation corresponds to a permutation of the n agents. To design an algorithm to compute an efficiency optimal allocation, we first need to decide what order of the agents should be. For the considered linear functions, one might expect that the optimal order, from left to right, would be according to the increasing order of the slope a_i. Counter-intuitively, this is not the case, as the following example shows.

Example 1. *Consider two agents $\{1,2\}$ with density functions $f_1(x) = x$ and $f_2(x) = 2x + 10000$. Clearly, in an*

optimal allocation, agent 2 *will get as much as he can to maximize efficiency. If the order is* $1, 2$ *from left to right according to slope, then agent* 1 *needs to get* $\left[0, \frac{\sqrt{2}}{2}\right]$ *to ensure his proportional share and agent* 2 *gets the rest. The efficiency in this order is*

$$\frac{1}{4}+\int_{\frac{\sqrt{2}}{2}}^{1}(2x+10000)\,dx = 10000+\frac{3}{4}-5000\sqrt{2} \approx 2930.75.$$

If the order is reversed, then agent 1 *gets* $\left[\frac{\sqrt{2}}{2}, 1\right]$ *due to proportionality, and agent* 2 *gets the rest. The efficiency in this order is*

$$\frac{1}{4}+\int_{0}^{\frac{\sqrt{2}}{2}}(2x+10000)\,dx = \frac{3}{4}+5000\sqrt{2} \approx 7070.75,$$

which is larger. Thus, ordering agents according to their slopes does not yield an optimal solution.

As we show in the following, the optimal order is actually determined by a quantity d_i, called *order determinant*: For agent i with density function $f_i(x) = a_i x + b_i$, define

$$d_i = \frac{a_i}{a_i + 2b_i}.$$

Note that the denominator is guaranteed to be non-zero.

Proposition 1. *For any agents* i *and* j *and any* $0 \le p < q < r \le 1$*, the order determinant* $d_i < d_j$ *if and only if* $\frac{v_i(p,q)}{v_i(p,r)} > \frac{v_j(p,q)}{v_j(p,r)}$.

Proof. We first prove the case where $p = 0$ and $r = 1$. For agent i, we have $v_i(0,1) = \frac{1}{2}a_i + b_i$ and $v_i(0,q) = \int_0^q a_i x + b_i\,dx = \frac{1}{2}a_i q^2 + b_i q$. Thus, we have

$$\frac{v_i(0,q)}{v_i(0,1)} = \frac{\frac{1}{2}a_i q^2 + b_i q}{\frac{1}{2}a_i + b_i} = q+(q^2-q)\frac{a_i}{a_i+2b_i} = q+(q^2-q)d_i.$$

Similarly, we have $\frac{v_j(0,q)}{v_j(0,1)} = q+(q^2-q)d_j$. Because $q^2-q < 0$, it is obvious that $d_i < d_j$ if and only if $\frac{v_i(0,q)}{v_i(0,1)} > \frac{v_j(0,q)}{v_j(0,1)}$.

For the case where $p \neq 0$ or $r \neq 0$, we apply a transformation to the density function $f_i(\cdot)$: Let

$$\begin{cases} a_i' = a_i \cdot (r-p) \\ b_i' = a_i p + b_i \end{cases}$$

Define a new density function $f_i'(x) = a_i' x + b_i'$ and let $v_i'(a,b) = \int_a^b f_i'(x)\,dx$. Let $y = \frac{q-p}{r-p}$; one can verify that

$$\frac{v_i(p,q)}{v_i(p,r)} = \frac{\frac{1}{2}a_i(q^2-p^2)+b_i(q-p)}{\frac{1}{2}a_i(r^2-p^2)+b_i(r-p)} = \frac{\frac{1}{2}a_i' y^2 + b_i' y}{\frac{1}{2}a_i' + b_i'} = \frac{v_i'(0,y)}{v_i'(0,1)}.$$

Using a similar transformation to $f_j(\cdot)$, we can reduce the problem to the case where $p = 0$ and $r = 1$. Hence, we have

$$\begin{aligned} \frac{v_i(p,q)}{v_i(p,r)} > \frac{v_j(p,q)}{v_j(p,r)} &\Leftrightarrow \frac{v_i'(0,y)}{v_i'(0,1)} > \frac{v_j'(0,y)}{v_j'(0,1)} \\ &\Leftrightarrow \frac{a_i'}{a_i'+2b_i'} < \frac{a_j'}{a_j'+2b_j'} \\ &\Leftrightarrow \frac{a_i(r-p)}{a_i(r+p)+2b_i} < \frac{a_j(r-p)}{a_j(r+p)+2b_j} \\ &\Leftrightarrow \frac{a_i}{a_i+2b_i} < \frac{a_j}{a_j+2b_j} \\ &\Leftrightarrow d_i < d_j \end{aligned}$$

This completes the proof. □

The order determinant precisely characterizes the order of the agents in an optimal allocation, shown by the following lemma.

Lemma 1. *There exists an efficiency optimal allocation in which the allocation (from left to right) is according to the non-decreasing order of* d_i.

Proof. Given an efficiency optimal allocation, assume that the allocation to the agents are in the order $1, 2, \ldots, n$. Then for any $1 \le i < n$, it suffices to show the following:

- $d_i \le d_{i+1}$;
- if $d_i = d_{i+1}$, then switching the order of i and $i+1$ will lead to an efficiency optimal allocation as well.

We will prove the claims by considering each $i = 1, \ldots, n$ in the sequence.

Suppose that in an efficiency optimal allocation, agent i is assigned interval $[p,q]$, and agent $i+1$ is assigned to $[q,r]$. Let $\alpha_i = \frac{v_i(p,q)}{v_i(p,r)}$ and $\alpha_{i+1} = \frac{v_{i+1}(q,r)}{v_{i+1}(p,r)}$ denote the ratio of the valuation of agent i and $i+1$ between their own allocation and the union of their allocations, respectively. We next consider two cases.

(1) $\alpha_i \ge \alpha_{i+1}$. We can find a point $q' \in [p,q]$, such that $\frac{v_i(p,q')}{v_i(p,r)} = \alpha_{i+1}$. Now we consider a new allocation, in which agent $i+1$ gets $[p,q']$, agent i gets $[q',r]$, and the allocation of all other agents remains unchanged. The social welfare difference between the original allocation and the new one is

$$\begin{aligned} \delta &\triangleq (v_i(p,q)+v_{i+1}(q,r)) - (v_{i+1}(p,q')+v_i(q',r)) \\ &= (v_{i+1}(q,r)-v_{i+1}(p,q')) + (v_i(p,q')-v_i(q,r)) \end{aligned}$$

If $d_i > d_{i+1}$, by Proposition 1, we have

$$\frac{v_{i+1}(p,q')}{v_{i+1}(p,r)} > \frac{v_i(p,q')}{v_i(p,r)} = \alpha_{i+1} \text{ and } \frac{v_{i+1}(p,q)}{v_{i+1}(p,r)} > \frac{v_i(p,q)}{v_i(p,r)}.$$

Hence,

$$\begin{aligned} v_{i+1}(p,q') &> \alpha_{i+1} \cdot v_{i+1}(p,r) \\ &= \frac{v_{i+1}(q,r)}{v_{i+1}(p,r)} \cdot v_{i+1}(p,r) = v_{i+1}(q,r), \end{aligned}$$

and

$$\begin{aligned} \frac{v_i(q,r)}{v_i(p,r)} &= 1 - \frac{v_i(p,q)}{v_i(p,r)} \\ &> 1 - \frac{v_{i+1}(p,q)}{v_{i+1}(p,r)} = \frac{v_{i+1}(q,r)}{v_{i+1}(p,r)} = \alpha_{i+1}. \end{aligned}$$

Hence, $v_i(q,r) > \alpha_{i+1} \cdot v_i(p,r) = v_i(p,q')$. This means that $\delta < 0$, i.e., the new allocation has larger efficiency than the original allocation, which contradicts the fact that the original one is an efficiency optimal allocation. Thus, we have $d_i \le d_{i+1}$.

On the other hand, if $d_i = d_{i+1}$, then by the same calculation as above, we can get $v_i(q,r) = v_i(p,q')$ and $v_{i+1}(p,q') = v_{i+1}(q,r)$, which implies that $\delta = 0$. Hence, we get another optimal allocation with the order of i and $i+1$ switched.

(2) $\alpha_i < \alpha_{i+1}$. In this case we can find a point $q' \in [q, r]$ such that $\frac{v_{i+1}(q',r)}{v_{i+1}(p,r)} = \alpha_i$. The remaining details of the proof are similar to the first case and are omitted.

Therefore, by considering each $i = 1, \ldots, n-1$ sequently, the lemma follows. □

Polynomial Time Approximation Scheme

Having a characterization of the order of allocations in an optimal solution, we will next compute such an allocation. However, there is one issue: For linear functions, a fair allocation may require to cut the cake at irrational points. For instance, in Example 1, the unique fair allocation requires to cut the cake at point $\frac{\sqrt{2}}{2}$. This barrier prevents us from finding a tractable algorithm that always outputs a precisely fair allocation. As a result, we relax our requirement by considering $(1-\epsilon)$-fair allocation, in which each agent gets no less than $\frac{1-\epsilon}{n}$ share of its value for the entire cake. Such an approximation on fairness has been considered in, e.g., (Cohler et al. 2011; Zivan 2011).

We will use the approach of dynamic programming. For any given $\epsilon > 0$, we divide the interval $[0, 1]$ evenly into $K = \lceil \frac{2n}{\epsilon} \rceil$ subintervals; and we will keep each subinterval as a whole in the allocation. Let $H(i, k)$ denote the maximum efficiency one can get if we allocate the interval $[0, \frac{k}{K}]$ to the first i agents in the given optimal order, under the condition that each agent j, $1 \le j \le i$, gets a share of at least $\frac{1-\epsilon}{n}$. The dynamic programming algorithm is described as follows.

ALG-LINEAR

1: Sort all agents in a non-decreasing order with respect to their order determinant d_i.
2: Let $H(0, 0) = 0$ and $H(0, k) = -\infty$ for all $0 < k \le K$.
3: **for** $1 \le i \le n, 1 \le k \le K$ **do**
4: Among all $0 \le k' \le k$ such that $v_i(\frac{k'}{K}, \frac{k}{K}) \ge (\frac{1-\epsilon}{n}) v_i(0, 1)$, let

$$H(i, k) = \max_{k'} \left\{ H(i-1, k') + v_i\Big(\frac{k'}{K}, \frac{k}{K}\Big) \right\}.$$

5: **end for**
6: Return $H(n, K)$ and its corresponding allocation.

Theorem 1. *When all agents have linear density functions, for any $\epsilon > 0$,* ALG-LINEAR *runs in time polynomial in the input size and $\frac{1}{\epsilon}$, and outputs an $(1-\epsilon)$-fair allocation with efficiency within a factor of $(1-\epsilon)$ to the optimum.*

Proof. It is easy to check that this algorithm can be done in $O(n^3(\frac{1}{\epsilon})^2)$ time, and it will always output an efficiency maximal ϵ-fair allocation in which the cake is cut only at the given separations, assuming there exists one. Let $A = (A_1, \ldots, A_n)$ be an efficiency optimal fair allocation. If we round each cutting point of A to the closest smaller separation given by K, in the new allocation every agent i will get a value no less than $v_i(A_i) - \frac{\epsilon}{2n}(a_i + b_i) \ge v_i(A_i) - \frac{\epsilon}{n} v_i(0, 1) \ge (1-\epsilon) v_i(A_i)$. Thus, it is an ϵ-fair allocation, and the efficiency of this allocation will be at least $(1-\epsilon)\sum_i v_i(A_i)$, which is always within a factor of $(1-\epsilon)$ to the optimum. This finishes the proof of the theorem. □

Piecewise Constant Functions

In this section, we consider piecewise constant functions. A density function $f(\cdot)$ is called *piecewise constant* if one can partition the whole interval $[0, 1]$ into subintervals such that f has a constant value on each subinterval. Piecewise constant functions are a representative class because it is easy to describe algorithmically and can approximate any density functions with arbitrarily good accuracy.

For a given piecewise constant function $f(\cdot)$, we say it contains k segments if there exist

$$0 \le a_1 < b_1 \le a_2 < b_2 \le \cdots \le a_k < b_k \le 1$$

such that (i) f has a constant positive value on each $[a_i, b_i]$, (ii) f has value 0 on all remaining regions, and (iii) for each $[a_i, b_i]$ and $[a_{i+1}, b_{i+1}]$, either $b_i < a_{i+1}$ or f has different values on these two segments. We say $[a_i, b_i]$ a *segment* and a_i, b_i its endpoints.

Next we will consider computational complexity of computing an efficiency optimal allocation for piecewise constant functions. As can be seen, its complexity is completely different from the linear function case.

Fixed Order

For a piecewise constant function, its input consists of its value and two endpoints (all rational numbers) of all of its (finite) segments. It can be seen (below) that there always exists an optimal solution whose cutting points are rational. Indeed, if the winning order of the agents is explicitly given, such an optimal solution can be found in polynomial time. This result is in contrast with the linear function case, in which an optimal order can be found efficiently but an optimal solution can only be approximated.

Suppose the order of the agents, from left to right, is fixed as $1, 2, \ldots, n$. Let $0 \le p_1 \le p_2 \le \ldots \le p_K \le 1$ be the lists of endpoints of all segments of all agents. While in an optimal allocation, the points of cuts are not necessarily in $\{p_1, \ldots, p_K\}$, those cutting points that are not in the set can be nicely characterized. This leads to the following dynamic programming, which computes an *exactly* optimal allocation.

Let $H(i, p_k)$ denotes the maximum efficiency one can get if we allocate the interval $[0, p_k]$ to the first i agents. The dynamic programming is described as follows.

Theorem 2. *Given a fixed winning order of agents,* ALG-PIECEWISE-CONSTANT *computes an efficiency optimal proportional allocation in polynomial time.*

Proof. In order to prove correctness of the algorithm, we first show the following characterization, which immediately implies that the dynamic programming always outputs an optimal solution. That is, to allocate interval $[p_{k'}, p_k]$ to agents $j, j+1, \ldots, i$ under the fairness condition, there always is an efficiency optimal allocation such that one of the following holds:

- Some agent's allocation is cut at one endpoint of one of its own segments.

ALG-PIECEWISE-CONSTANT

1: Let $H(0, p_k) = 0$ for all $1 \le k \le K$.
2: **for** $1 \le i \le n, 1 < k \le K$ **do**
3: Let

$$H(i, p_k) = \max_{\substack{1 \le j \le i \\ 1 \le k' < k}} \Big\{ H(j-1, p_{k'}) + S(j, i, p_{k'}, p_k) \Big\},$$

where $S(j, i, p_{k'}, p_k)$ is the maximum efficiency of proportional allocation one can get by allocating interval $[p_{k'}, p_k]$ to agents $j, j+1, \ldots, i$ under the condition that at most one agent can have valuation more than $1/n$ of its share (if no such allocation exists, define it to be 0).
4: **end for**
5: Return $H(n, p_K)$ and its corresponding allocation.

- At most one agent has valuation more than $1/n$ of its total share.

To prove this, given an optimal allocation, assume that no agent's allocation is cut at any endpoint of its segments and there are two agents x, y both getting more than $1/n$ of their total shares. If we keep the allocation valuations of all other agents fixed, except x, y, and add a disturbance Δ_x to agent x's valuation, it will sequentially result in a change $-\Delta_y$ in agent y's valuation, where $\Delta_x, \Delta_y > 0$. (That is, we slightly move the cutting points for all agents between x and y, including themselves; and all these agents except x and y still obtain the same valuations.) Similarly, if we disturb x's valuation by $-\Delta_x$, y's valuation will be changed by Δ_y. This means that if $\Delta_x \neq \Delta_y$, we can always perform such a disturbance to get a proportional allocation with larger efficiency. If $\Delta_x = \Delta_y$, we can keep performing this disturbance, until either one of x and y has reached exactly $1/n$ of its share, or one agent (in the between of x and y) has reached an endpoint cut. In this process, the efficiency of the allocation remains unchanged. (Note that the piecewise constant condition is critical for the argument to apply.) The claim then follows by applying this process repeatedly until the number of agents whose share is more than $1/n$ is 1 in the final allocation.

Finally we prove that the algorithm runs in polynomial time. It suffices to show that $S(j, i, p_{k'}, p_k)$ can be polynomially computed. In fact, one can just enumerate the agent whose valuation is (potentially) more than $1/n$ of its total share. Once that agent is fixed, all other agents should have exactly $1/n$ of their total shares. Thus the allocation can be uniquely determined, and can be computed in polynomial time. This finishes the proof of the theorem. □

NP-Hardness

While the above algorithm gives a hope to find an optimal solution in polynomial time, we do not necessarily know the optimal order of the agents in an optimal allocation. In general, we have the following hardness result, which says that computing an allocation whose efficiency is within a factor of $\Omega\big(\frac{1}{\sqrt{n}}\big)$ to the optimum, conditioned on proportionality, is NP-hard.

Theorem 3. *When all agents have piecewise constant density functions, it is NP-hard to approximate the optimum efficiency within a ratio of $\Omega\big(\frac{1}{\sqrt{n}}\big)$, even if all functions have at most 2 segments.*

Proof. Our reduction is from the partition problem. Consider the following partition problem instance: given a set A' of $2k$ positive integers $A' = \{a'_1, a'_2, \ldots, a'_{2k}\}$; let $2m' = \sum_i a'_i$. It is NP-complete to decide whether A' can be divided into two subsets such that both of them have exactly k integers, and the sum of the numbers in each subset equals m' (Garey and Johnson 1979). Without loss of generality, we assume that all a'_i are even numbers. Further, we add an integer $2km'$ to each a'_i and get a new set $A = \{a_1, a_2, \ldots, a_{2k}\}$ where $a_i = a'_i + 2km'$. It is easy to see that A has a partition of sum $m \triangleq 2k^2m' + m'$ if and only if A' has a partition of sum m'. Let $M = \max_i a_i$.

For the instance A, we have the following observations, which can be verified easily.

1. If A can be partitioned into A_1 and A_2 with equal sums, then $|A_1| = |A_2| = k$.
2. If the absolute value of the difference between the total values of A_1 and A_2 is less than or equal to 1, then they have the same total value (as all numbers are even).

We next construct a cake cutting instance from the partition instance A. Assume that the cake is modeled as an interval $[0, 3m(2k-1) + 2k(n+1) - 1]$ (which can be easily normalized to $[0, 1]$). There are in total $n = 4k^2$ agents, divided into 3 groups: A, B and C.

- Group A has $2k-1$ agents. We label them by

$$1, 1 + (2k+1), 1 + 2(2k+1), ..., 1 + (2k-2)(2k+1),$$

and define their valuation density functions to be

$$f_{1+t(2k+1)}(x) = \begin{cases} 1 & x \in \big[m + 3mt, m + 3mt + 1\big] \\ 0 & \text{otherwise} \end{cases}$$

for $t = 0, 1, 2, ..., 2k-2$. Note that the total value of each agent is 1.
- Group B has in total $2k(2k-1)$ agents, which are further divided into $2k-1$ subgroups $B_0, B_1, \ldots, B_{2k-2}$. Each subgroup B_t has $2k$ agents, which are labeled by

$$2+t(2k+1), 3+t(2k+1), 4+t(2k+1), ..., 2k+1+t(2k+1).$$

Corresponding to each group B_t, define two intervals on the cake as follows:

$$I_{1t} \triangleq \big[3mt, 3mt + 2m + 1\big] \quad \text{and}$$

$$I_{2t} \triangleq \big[3m(2k-1)+t(n+1)+n, 3m(2k-1)+t(n+1)+n+1\big]$$

The valuation functions of the agents in B_t are

$$f_{i+1+t(2k+1)}(x) = \begin{cases} 1 & x \in I_{1t} \\ na_i - (2m+1) & x \in I_{2t} \\ 0 & \text{otherwise} \end{cases}$$

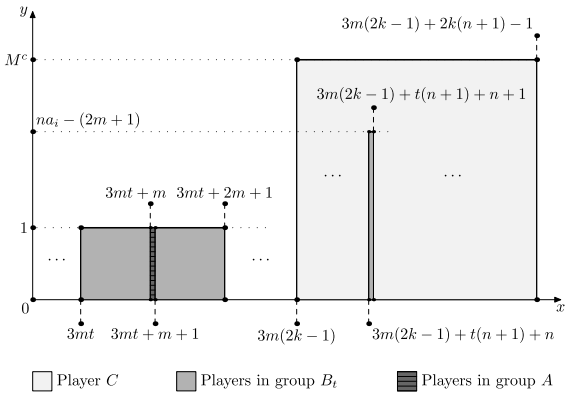
where $i = 1, 2, ..., 2k$. Note that it is guaranteed that $na_i - (2m+1) > 0$.

It can be seen that the total value of each agent $i+1+t(2k+1)$ is na_i; thus, in a proportional allocation, its share is at least a_i. In addition, the intervals are disjoint for agents in different subgroups B_t, and each B_t contains exactly one agent (i.e., $1+t(2k+1)$) from A.

- Group C contains only one agent, and his density function is (where c is a large integer)

$$f_n(x) = \begin{cases} M^c & x \in \big[3m(2k-1), \\ & \quad 3m(2k-1)+2k(n+1)-1\big] \\ 0 & \text{otherwise} \end{cases}$$

The figure below shows the construction of the cake cutting instance. Note that all agents have piecewise constant valuations with at most two segments.



For each agent in subgroup B_t, he can get at least $1/n$ of his total valuation only if he gets an allocation in I_{1t} of length at least a_i or in I_{2t} of at least certain length. If there is a valid partition in the partition instance A, then, for each t, agents in B_t can share the interval $[3mt, 3mt+m]$ and $[3mt+m+1, 3mt+2m+1]$ completely to get $1/n$ of their total valuation. Thus, we can allocate agents in group A and group C all of their interested intervals. The total value of all agents in this case is therefore

$$M^c(2k(n+1)-1) + (2k-1)(2m+1).$$

On the other hand, if the partition instance does not have a valid solution, by the two observations established above, we cannot partition A into two subsets such that the difference between their sums is less than or equal to 1. This means that each subgroup B_t has at least one agent allocated a piece inside the interval I_{2t}. As there are $2k-1$ subgroups and the allocation must be continuous, agent C can get only one of the $2k$ leftover segments:

$$\begin{aligned}
&[3m(2k-1), 3m(2k-1)+(n+1)-1] \\
&[3m(2k-1)+n+1, 3m(2k-1)+2(n+1)-1] \\
&[3m(2k-1)+2(n+1), 3m(2k-1)+3(n+1)-1] \\
&\cdots\cdots \\
&[3m(2k-1)+(2k-1)(n+1), \\
&\qquad 3m(2k-1)+2k(n+1)-1]
\end{aligned}$$

where each segment has length n. The total value of all agents in this case is then at most

$$M^c(n+1) + (2k-1)(2m+1) + (2k-1)nM,$$

where the first term is an upper bound on the valuation for the agent in group C, and the last two terms are for the agents in groups A and B (note that $na_i - (2m+1) < nM$).

Note that

$$\begin{aligned}
& \frac{M^c(n+1)+(2k-1)(2m+1)+(2k-1)nM}{M^c(2k(n+1)-1)+(2k-1)(2m+1)} \\
< & \frac{M^c(n+1)+(2k-1)(2m+1)+(2k-1)nM}{M^c(2k(n+1)-1)} \\
= & \frac{n+1}{2k(n+1)-1} + \frac{(2k-1)(2m+1)+(2k-1)nM}{M^c(2k(n+1)-1)} \\
\le & \Omega\Big(\frac{1}{2k}\Big) = \Omega\Big(\frac{1}{\sqrt{n}}\Big)
\end{aligned}$$

where the last inequality follows from the observations that $m \le kM$ and we can pick c to be a sufficiently large number. Therefore, it is NP-hard to approximate efficiency within a ratio of $\Omega\big(\frac{1}{\sqrt{n}}\big)$. □

Normalized Valuations

We say a density function f *normalized* if $\int_0^1 f(x)\,dx = 1$. For such functions, computing an efficiency optimal allocation is NP-hard as well[2].

Theorem 4. *When all agents have normalized piecewise constant density functions, it is NP-hard to compute the optimum efficiency, even if all functions have exactly 1 segment.*

Proof. We again reduce from the partition problem: There is a set of $2k$ positive integers $A' = \{a'_1, a'_2, ..., a'_{2k}\}$ with $\sum_{i=1}^{2k} a'_i = 2m'$. The problem is to partition these integers to two subsets with equal size such that both sets have sum m'. Without loss of generality, we assume that $a'_1 \le a'_2 \le \cdots \le a'_{2k}$ and each a'_i is a multiple of 3. We construct a new set $A = \{a_1, a_2, ..., a_{2k}\}$, where each $a_i = a'_i + (4k+8)m'$. It is easy to see that A has a partition of sum $m \triangleq m' + k(4k+8)m'$ each if and only if A' has a valid partition.

We have the following observations.

1. If A can be partitioned into A_1 and A_2 with equal sums, then $|A_1| = |A_2| = k$.
2. If the absolute value of the difference between the sums of A_1 and A_2 is less than or equal to 2, then they have the same sum (as all numbers are multiples of 3).
3. $(2k+3)a_1 > 2m+2$. This is because
$$\begin{aligned}
(2k+3)a_1 &= (2k+3)(a'_1 + (4k+8)m') \\
&> (2k+3)(4k+8)m' \\
&\ge 2(m' + k(4k+8)m') + 2 = 2m+2
\end{aligned}$$
4. We have
$$\begin{aligned}
2m+2 &> m = m' + k(4k+8)m' \\
&> 2m'(2k+3) > a'_{2k}(2k+3) \\
&> (a'_{2k} - a'_1)(2k+3) \\
&= (a_{2k} - a_1)(2k+3)
\end{aligned}$$

This implies that $a_1(2k+3) > a_{2k}(2k+3) - (2m+2)$.

[2]Note that Theorem 3 does not imply the claim here, as in its proof agents' valuations are not normalized. On the other hand, the NP-hardness result established by Theorem 4 applies to non-normalized setting as well; but the statement of Theorem 3, $\Omega\big(\frac{1}{\sqrt{n}}\big)$-hardness, is much stronger.

Let $b_i = \frac{1}{a_i(2k+3)}$, for $i = 1, \ldots, 2k$, and define

$$b = \frac{1}{a_{2k}(2k+3) - (2m+2)}.$$

Hence, $b > b_1 \geq b_2 \geq \cdots \geq b_{2k}$. Further, a careful calculation shows that $2m(b_1 - b_{2k}) < a_1(b - b_1)$.

Now we construct a cake cutting instance with $2k+3$ agents. For the first $2k$ agents, their density functions are

$$v_i(x) = \begin{cases} b_i & x \in [0, a_i(2k+3)] \\ 0 & \text{otherwise} \end{cases}$$

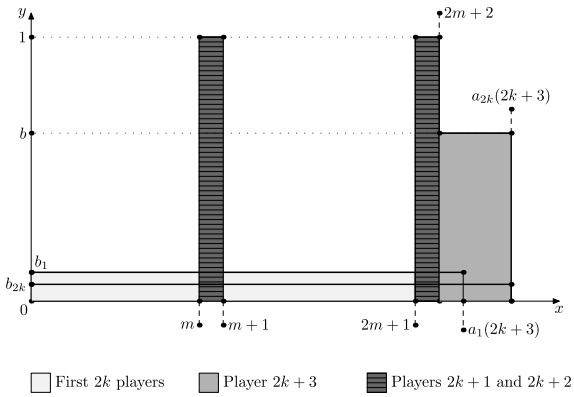
For the last three agents, their density functions are

$$v_{2k+1}(x) = \begin{cases} 1 & x \in [m, m+1] \\ 0 & \text{otherwise} \end{cases}$$

$$v_{2k+2}(x) = \begin{cases} 1 & x \in [2m+1, 2m+2] \\ 0 & \text{otherwise} \end{cases}$$

$$v_{2k+3}(x) = \begin{cases} b & x \in [2m+2, a_{2k}(2k+3)] \\ 0 & \text{otherwise} \end{cases}$$

The figure below shows the construction.



In this construction, note that each agent's valuation density function is normalized, and each of the first $2k$ agents needs a length of a_i to get its proportional share. Moreover, the two intervals $[m, m+1]$ and $[2m+1, 2m+2]$ are "reserved" to agents $2k+1$ and $2k+2$ due to their large density values and the second observation above.

It can be seen that if the partition instance has a valid solution, the total value of all agents V_{yes} is at least

$$V_{yes} = \sum_{i=1}^{2k} a_i b_i + 2 + 1 \geq 2mb_{2k} + 3.$$

On the other hand, if the partition instance does not have a valid solution, the total value of all agents V_{no} is at most

$$V_{no} < 2mb_1 + 2 + a_1 b_1 + [a_{2k}(2k+3) - (2m+2) - a_1]b.$$

Therefore,

$$\begin{aligned} & V_{yes} - V_{no} \\ > \; & 1 - a_1 b_1 - [a_{2k}(2k+3) - (2m+2) - a_1]b - 2m(b_1 - b_{2k}) \\ = \; & a_1(b - b_1) - 2m(b_1 - b_{2k}) > 0 \end{aligned}$$

Since the partition problem is NP-complete and $V_{yes} > V_{no}$, computing an optimal allocation for the constructed normalized cake cutting problem is NP-hard. □

Concluding Remarks

We study efficiency optimization in cake cutting. The solution conditions are that of proportional fairness and $n-1$ cuts. While the fairness condition is relatively easy to handle (see, e.g., (Cohler et al. 2011)), the requirement on the number of cuts makes the problem much more difficult to analyze. For instance, some powerful tools like convex programming do not apply. We give an optimal design for linear valuation functions, and show hardness results for piecewise constant functions. A natural and intriguing direction for future research is to design (approximate) algorithms, especially for (normalized) piecewise constant functions, to maximize efficiency given fairness and $n-1$ cuts. In particular, an important question is whether there exists a constant approximation algorithm.

Our work purely focuses on efficiency and fairness—we do not attempt to consider strategic behaviors of the agents in the model. Designing incentive compatible protocols in cake cutting has been considered in (Chen et al. 2010). It is an interesting direction to study efficiency, fairness and incentive altogether in order to design more robust protocols.

Acknowledgements

The first author was supported in part by the National Basic Research Program of China Grants 2011CBA00300, 2011CBA00301, and the National Natural Science Foundation of China Grants 61033001, 61061130540, 61073174. The last two authors wish to acknowledge the funding support for this project from Nanyang Technological University under the Undergraduate Research Experience on CAmpus (URECA) programme.

References

Aumann, Y., and Dombb, Y. 2010. The efficiency of fair division with connected pieces. In *WINE*, 26–37.

Aumann, Y.; Dombb, Y.; and Hassidim, A. 2012. Computing socially-efficient cake divisions. *Working paper*.

Brams, S.; Feldman, M.; Morgenstern, J.; Lai, J.; and Procaccia, A. 2012. On maxsum fair cake divisions. In *AAAI*.

Caragiannis, I.; Kaklamanis, C.; Kanellopoulos, P.; and Kyropoulou, M. 2009. The efficiency of fair division. In *WINE*, 475–482.

Chen, Y.; Lai, J.; Parkes, D.; and Procaccia, A. 2010. Truth, justice, and cake cutting. In *AAAI*, 756–761.

Cohler, Y.; Lai, J.; Parkes, D.; and Procaccia, A. 2011. Optimal envy-free cake cutting. In *AAAI*.

Dubins, L., and Spanier, E. 1934. How to cut a cake fairly. *American Mathematical Monthly* 68:1–17.

Garey, M., and Johnson, D. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman.

Papadimitriou, C. 1993. *Computational Complexity*. Addison Wesley.

Reijnierse, J., and Potters, J. 1998. On finding an envy-free pareto-optimal division. *Mathematical Programming* 83:291–311.

Robertson, J., and Webb, W. 1998. *Cake-Cutting Algorithms: Be Fair if You Can*. Peters/CRC Press.

Zivan, R. 2011. Can trust increase the efficiency of cake cutting algorithms? In *AAMAS*, 1145–1146.