

DIGITAL ACCESS TO SCHOLARSHIP AT HARVARD
DASH.HARVARD.EDU



HARVARD LIBRARY
Office for Scholarly Communication

Truth, Justice, and Cake Cutting

Citation

Chen, Yiling, John Kwang Lai, David C. Parkes, and Ariel D. Procaccia. 2010. Truth, justice, and cake cutting. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence and the Twenty-Second Innovative Applications of Artificial Intelligence Conference: 11-15 July, 2010, Atlanta, Georgia. Menlo Park, CA: AAAI Press.

Published Version

http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/viewFile/1761/2083

Permanent link

http://nrs.harvard.edu/urn-3:HUL.InstRepos:8896229

Terms of Use

This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Open Access Policy Articles, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#OAP

Share Your Story

The Harvard community has made this article openly available.
Please share how this access benefits you. Submit a story .

Accessibility

Truth, Justice, and Cake Cutting

Yiling Chen
Harvard SEAS
yiling@eecs.harvard.edu

John K. Lai
Harvard SEAS
jklai@post.harvard.edu

David C. Parkes
Harvard SEAS
parkes@eecs.harvard.edu

Ariel D. Procaccia
Harvard SEAS
arielpro@seas.harvard.edu

Superman: "I'm here to fight for truth, justice, and the American way."
Lois Lane: "You're gonna wind up fighting every elected official in this country!"

Superman (1978)

Abstract

Cake cutting is a common metaphor for the division of a heterogeneous divisible good. There are numerous papers that study the problem of fairly dividing a cake; a small number of them also take into account self-interested agents and consequent strategic issues, but these papers focus on fairness and consider a strikingly weak notion of truthfulness. In this paper we investigate the problem of cutting a cake in a way that is truthful and fair, where for the first time our notion of dominant strategy truthfulness is the ubiquitous one in social choice and computer science. We design both deterministic and randomized cake cutting algorithms that are truthful and fair under different assumptions with respect to the valuation functions of the agents.

Introduction

The need for resource allocation arises in many AI domains, and in particular in multiagent systems. This has led to a wide interest in the field known as *Multiagent Resource Allocation*, and to various applications of resource allocation techniques (see the survey by Chevalyere et al. (2006)). Resource allocation problems deal with either *divisible* or *indivisible* resources, where the distinction is based on whether any fraction of a resource can be given to an agent.

Cutting a cake is often used as a metaphor for allocating a divisible good. The difficulty is not cutting the cake into pieces of equal size, but rather that the cake is not uniformly tasty: different agents prefer different parts of the cake, depending, e.g., on whether the toppings are strawberries or cookies. The goal is to divide the cake in a way that is "fair"; the definition of fairness is a nontrivial issue in itself, which we discuss in the sequel. The cake cutting problem dates back to the 1940s, and for over sixty years has attracted the attention of mathematicians, economists, and political scientists. While most of the work in artificial intelligence, and computer science in general, has focused on the allocation of indivisible resources, recent years have seen an increasing interest among computer scientists in the allocation of divisible resources (see, e.g, (Edmonds and Pruhs 2006a; 2006b; Procaccia 2009)).

Slightly more formally, the cake is represented by the interval $[0, 1]$. Each of n agents has a valuation function over the cake, which assigns a value to every given piece of cake and is additive. The goal is to find a partition of the cake among the agents (while possibly throwing a piece away) that satisfies one or several fairness criteria. In this paper we consider the two most prominent criteria. A *proportional* allocation is one where the value each agent has for its own piece of cake is at least $1/n$ of the value it assigns to the entire cake. An *envy-free (EF)* allocation is one where the value each agent assigns to its own piece of cake is at least as high as the value it assigns to any other agent's piece of cake. There is a rather large body of literature on fairly cutting a cake according to these two criteria (see, e.g., the books by Robertson and Webb (1998) and Brams and Taylor (1996)).

So far we have briefly discussed "justice", but have not yet mentioned "truth." Taking the game-theoretic point of view, an agent's valuation function is its private information, which is reported to a cake cutting algorithm. We would like an algorithm to be *truthful*, in the sense that agents are motivated to report their true valuation functions. Like fairness, this idea of truthfulness also lends itself to many interpretations. One variation, referred to as *strategy-proofness* in previous papers by Brams et al. (2006; 2008), assumes that an agent would report its truthful valuation rather than lie if there *exist* valuations of the other agents such that reporting truthfully yields at least as much value as lying. In the words of Brams et al., "...the players are risk-averse and never strategically announce false measures if it does not guarantee them more-valued pieces. ... Hence, a procedure is strategy-proof if no player has a strategy that dominates his true value function." (Brams, Jones, and Klamler 2008, page 362).

The foregoing notion is strikingly weak compared to the notion of truthfulness that is common in the social choice literature. Indeed, strategy-proofness is usually taken to mean that an agent can *never* benefit by lying, that is, *for all* valuations of the other agents reporting truthfully yields at least as much value as lying. Put another way, truth-telling is a dom-

Copyright © 2010, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

inant strategy. This notion is worst-case, in the sense that an agent cannot benefit by lying even if it is fully knowledgeable of the valuations of the other agents. It is also the predominant one in the computer science literature, and in particular in the algorithmic mechanism design literature (Nisan and Ronen 2001). In order to prevent confusion we will avoid using the term "strategy-proof," and instead refer to the former notion of Brams et al. as "weak truthfulness" and to the latter standard notion as "truthfulness."

To illustrate the difference between the two notions, consider the most basic cake cutting algorithm for the case of two agents, the *Cut and Choose* algorithm.[1] Agent 1 cuts the cake into two pieces that are of equal value according to its valuation; agent 2 then chooses the piece that it prefers, giving the other piece to agent 1. This algorithm is trivially proportional and EF.[2] It is also weakly truthful, as if agent 1 divides the cake into two pieces that are unequal according to its valuation then agent 2 may prefer the piece that is worth more to agent 1. Agent 2 clearly cannot benefit by lying. However, the algorithm is not truthful. Indeed, consider the case where agent 1 would simply like to receive as much cake as possible, whereas the single-minded agent 2 is only interested in the interval $[0, \epsilon]$ where ϵ is small (for example, it may only be interested in the cherry). If agent 1 follows the protocol it would only receive half of the cake. Agent 1 can do better by reporting that it values the intervals $[0, \epsilon]$ and $[\epsilon, 1]$ equally, since then it would end up with almost the entire cake by choosing to cut pieces $[0, \epsilon], [\epsilon, 1]$.

In this paper we consider the design of truthful and fair cake cutting algorithms. To the best of our knowledge we are the first to do so.[3] However, there is a major obstacle that must be circumvented: regardless of strategic issues, and when there are more than four agents, even finding a proportional and EF allocation in a bounded number of steps with a deterministic algorithm is a long-standing open problem! See Procaccia (2009) for an up-to-date discussion.[4] We shall therefore restrict ourselves to specific classes of valuation functions where efficiently finding fair allocations is a non-issue; the richness of our problem stems from our desire to additionally achieve truthfulness.

Our results. We first consider deterministic algorithms. We restrict ourselves to the case where the agents hold *piecewise uniform* valuation functions, that is, each agent is interested in a collection of subintervals of $[0, 1]$ with the same marginal value for each fractional piece in each subinterval. This is the case when some parts of the cake satisfy a certain property and an agent desires as much of these parts as possible. Our main result is a deterministic algorithm for any number of agents that is truthful, proportional, EF, and polynomial-time. The proof requires many ingredients, including a seemingly unlikely application of the classic Max-Flow Min-Cut Theorem.

We next consider randomized algorithms. We slightly relax truthfulness by asking that the algorithm be *truthful in expectation*, that is, an agent cannot hope to increase its expected value by lying for any reports of other agents. For general valuations, we present a simple randomized algorithm that is truthful in expectation, and always outputs an allocation that is proportional and EF. We further establish that this algorithm is tractable under the relatively weak assumption that the agents hold *piecewise linear* valuation functions, that is where the marginal value in each subinterval of interest is a linear function.

Preliminaries

We consider a heterogeneous cake, represented by the interval $[0, 1]$. A *piece of cake* is a *finite* union of subintervals of $[0, 1]$. We sometimes abuse this terminology by treating a piece of cake as the set of the (inclusion-maximal) intervals that it contains. The length of the interval $I = [x, y]$, denoted $\text{len}(I)$, is $y - x$. For a piece of cake X we denote $\text{len}(X) = \sum_{I \in X} \text{len}(I)$.

The set of agents is denoted $N = \{1, \ldots, n\}$. Each agent $i \in N$ holds a private valuation function V_i, which maps given pieces of cake to the value agent i assigns them. Formally, each agent i has a *value density function*, $v_i : [0, 1] \to [0, \infty)$, that is piecewise continuous. The function v_i characterizes how agent i assigns value to different parts of the cake. The value of a piece of cake X to agent i is then defined as $V_i(X) = \int_X v_i(x)dx = \sum_{I \in X} \int_I v_i(x)dx$. We note that the valuation functions are *additive*, i.e. for any two disjoint pieces X and Y, $V_i(X \cup Y) = V_i(X) + V_i(Y)$, and *non-atomic*, that is $V_i([x, x]) = 0$ for every $x \in [0, 1]$. The last property implies that we do not have to worry about the boundaries of intervals, i.e., open and closed intervals are identical for our purposes. We further assume that the valuation functions are *normalized*, i.e. $V_i([0, 1]) = \int_0^1 v_i(x)dx = 1$.

A cake cutting algorithm is a function f from the valuation function of each agent to an allocation $(A_1, \ldots, A_n)$ of the cake such that the pieces are pairwise disjoint. For each $i \in N$ the piece A_i is allocated to agent i, and the rest of the cake, i.e., $[0, 1] \setminus \bigcup_{i \in N} A_i$, is thrown away. Here we are assuming *free disposal*, that is, the algorithm can throw away resources without incurring a cost.

We say that an allocation $A_1, \ldots, A_n$ is *proportional* if for every $i \in N$, $V_i(A_i) \geq 1/n$, that is, each agent receives at least a $(1/n)$-fraction of the cake according to its own valuation. We say that an allocation is *envy-free (EF)* if for every $i, j \in N$, $V_i(A_i) \geq V_i(A_j)$, i.e., each agent prefers its own piece of cake to the piece of cake allocated to any other agent. A proportional (resp., EF) cake cutting algorithm always returns a proportional (resp., EF) allocation.

[1] This algorithm is described here with the agents taking actions; equivalently, the algorithm acts on behalf of agents using the reported valuations.

[2] Proportionality and envy-freeness coincide if there are two agents and the entire cake is allocated.

[3] Just before the acceptance of this paper we learned of an independent working paper that asks similar questions (Mossel and Tamuz 2010), but the technical overlap is minimal. There is also a loosely related paper by Thomson (2007), who showed that in general a truthful and Pareto-optimal algorithm must be dictatorial in the slightly different setting of pie-cutting.

[4] To be precise, previous algorithmic work assumed that the entire cake has to be allocated, but this does not seem to be a significant restriction in the context of fairness.

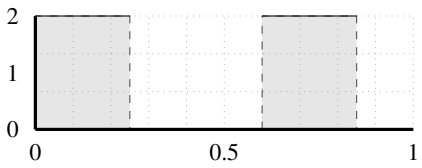


Figure 1: An illustration of the value density function for a piecewise uniform valuation.

Note that when $n = 2$ proportionality implies envy-freeness. Indeed, $V_i(A_i) + V_i(A_{3-i}) \leq 1$, and hence if $V_i(A_i) \geq 1/2$ then $V_i(A_{3-i}) \leq 1/2$. Under the free disposal assumption the converse is not true. For example, an allocation that throws away the entire cake is EF but not proportional. In general, when $n > 2$ proportionality neither implies nor is implied by envy-freeness.[5]

A cake cutting algorithm f is *truthful* if when an agent lies it is allocated a piece of cake that is worth, according to its real valuation, no more than the piece of cake it was allocated when reporting truthfully. Formally, denote $A_i = f_i(V_1, \ldots, V_n)$, and let $\mathcal{V}$ be a class of valuation functions. The algorithm f is truthful if for every agent i, every collection of valuations functions $V_1, \ldots, V_n \in \mathcal{V}$, and every $V_i' \in \mathcal{V}$, it holds that $V_i(f_i(V_1, \ldots, V_n)) \geq V_i(f_i(V_1, \ldots, V_{i-1}, V_i', V_{i+1}, \ldots, V_n))$.

Deterministic Algorithms and Piecewise Uniform Valuations

As noted in the introduction, in general there are no known bounded deterministic proportional and EF cake cutting algorithms for more than four agents, even if one is not concerned about strategic issues. Therefore, in this section we restrict ourselves to a specific class of valuation functions.

We say that a valuation function V_i is *piecewise constant* if and only if its corresponding value density function v_i is piecewise constant, that is $[0, 1]$ can be partitioned into a finite number of intervals such that v_i is constant on each interval. We say that V_i is *piecewise uniform* if moreover v_i is either some constant $c \in \mathbb{R}_+$ (the same one across intervals) or zero. See Figure 1 for an illustration.

Piecewise uniform valuation functions imply that agent $i \in N$ is uniformly interested in a finite union of intervals, which we call its *reference piece of cake* and denote by U_i. For example, in Figure 1, $U_i = [0, 0.25] \cup [0.6, 0.85]$. Given a piece of cake X, it holds that $V_i(X) = \text{len}(X \cap U_i)/\text{len}(U_i)$. From the computational perspective, the size of the input to the cake cutting algorithm is the number of bits that define the boundaries of the intervals in the agents' reference pieces of cake.

In the rest of this section we assume that the valuation functions are piecewise uniform. We believe that piecewise uniform valuations are very natural. An agent would have such a valuation function if it is simply interested in pieces of the good that have a certain property, e.g., a child only likes portions of the cake that have chocolate toppings, and wants as much cake with chocolate toppings as possible. We consider more general valuations in the next section on randomized algorithms.

[5] If free disposal is not assumed, that is, the entire cake is allocated, then envy-freeness implies proportionality for any n.

A deterministic algorithm

Before introducing our algorithm we present some required notation. Let $S \subseteq N$ be a subset of agents and let X be a piece of cake. Let $D(S, X)$ denote the portions of X that are valued by at least one agent in S. Formally, $D(S, X) = \left(\bigcup_{i \in S} U_i\right) \cap X$, and is itself a union of intervals.

Let $\text{avg}(S, X) = \text{len}(D(S, X))/|S|$ denote the average length of intervals in X desired by at least one agent in S. We say that an allocation is *exact* with respect to S and X if it allocates to each agent in S a piece of cake of length $\text{avg}(S, X)$ comprised *only* of desired intervals. Clearly this requires allocating all of $D(S, X)$ since the total length of allocated intervals is $\text{avg}(S, X) \cdot |S| = \text{len}(D(S, X))$. Suppose $S = \{1, 2\}$ and $X = [0, 1]$: if $U_1 = U_2 = [0, 0.2]$ then agents 1 and 2 receiving $[0, 0.1]$ and $[0.1, 0.2]$ respectively is an exact allocation; but if $U_1 = [0, 0.2], U_2 = [0.3, 0.7]$ then there is no exact allocation.

The deterministic algorithm for n agents with piecewise uniform valuations is a recursive algorithm that finds a subset of agents with a certain property, makes the allocation decision for that subset, and then makes a recursive call on the remaining agents and the remaining intervals. Specifically, for a given set of agents $S \subseteq N$ and a remaining piece of cake to be allocated X, we find the subset $S' \subseteq S$ of agents with the smallest $\text{avg}(S', X)$. We then give an exact allocation of $D(S', X)$ to S'. We recurse on $S \setminus S'$ and the intervals not desired by any agent in S', i.e. $X \setminus D(S', X)$. The pseudocode of the algorithm is given as Algorithm 1.

Algorithm 1 $(V_1, \ldots, V_n)$

1. SUBROUTINE$(\{1, \ldots, n\}, [0, 1], (V_1, \ldots, V_n))$

SUBROUTINE$(S, X, V_1, \ldots, V_n)$:

1. If $S = \emptyset$, return.
2. Let $S_{\min} \in \operatorname{argmin}_{S' \subseteq S} \text{avg}(S', X)$ (breaking ties arbitrarily).
3. Let $E_1, \ldots, E_n$ be an exact allocation with respect to $S_{\min}, X$ (breaking ties arbitrarily). For each $i \in S_{\min}$, set $A_i = E_i$.
4. SUBROUTINE$(S \setminus S_{\min}, X \setminus D(S_{\min}, X), (V_1, \ldots, V_n))$.

In particular, Steps 2 and 3 of SUBROUTINE imply that if $S = \{i\}$ then $A_i = D(S, X)$. For example, suppose $X = [0, 1]$, $U_1 = [0, 0.1]$, $U_2 = [0, 0.39]$, and $U_3 = [0, 0.6]$. In this case, the subset with the smallest average is $\{1\}$, so agent 1 receives all of $[0, 0.1]$ and we recurse on $\{2, 3\}, [0.1, 1]$. In the recursive call, set $\{2\}$ has average 0.39 - 0.1 = 0.29, set $\{3\}$ has average 0.6 - 0.1 = 0.5, and set $\{2, 3\}$ has average $(0.6 - 0.1)/2 = 0.25$. As a result, the entire set $\{2, 3\}$ is chosen as the set with smallest average, and an exact allocation of $[0.1, 1.0]$ is given to agents 2 and 3. One possible allocation is to give agent 2 $[0.1, 0.35]$ and agent 3 $[0.35, 0.6]$. Note that, if agent 1 uniformly val-

ues $[0, 0.2]$ instead, the first call would choose $\{1, 2\}$ as the subset with the smallest average, equally allocating $[0, 0.39]$ between agents 1 and 2 and giving the rest, $[0.39, 0.6]$, to agent 3.

An analysis of the two agent algorithm. To gain intuition, consider the case of two agents; designing truthful, proportional and EF algorithms even for this case is nontrivial. Assume that $\text{len}(U_1) \leq \text{len}(U_2)$ for ease of presentation. If in addition, $\text{len}(U_1) > \text{len}(U_1 \cup U_2)/2$ then set $\{1, 2\}$ has the smallest average and we divide $U_1 \cup U_2$ exactly, with each agent getting all of $U_i \setminus U_{3-i}$ and sharing $U_1 \cap U_2$ in a way that $\text{len}(A_1) = \text{len}(A_2)$. Otherwise, agent 1 gets all of U_1 and agent 2 gets $U_2 \setminus U_1$. The algorithm tries to give both agents the same length, with each agent always getting at least half of its desired intervals, leading to proportionality and EF because of piecewise uniform valuations. For sufficient overlap in desired intervals, each receives exactly half of $U_1 \cup U_2$. For totally disjoint reference pieces, each receives just its reference piece. We defer a discussion of truthfulness to the general algorithm; the crux here is to note that each agent i receives all of $U_i \setminus U_{3-i}$, and the algorithm precludes overclaims through providing a decreasing share of $U_i \cap U_{3-i}$ as $\text{len}(U_i)$ increases.

Exact Allocations and Maximum Flows. Before turning to properties of truthfulness and fairness, we point out that so far it is unclear whether Algorithm 1 is well-defined. In particular, the algorithm requires an exact allocation E with respect to the subset $S_{\min}$ and X, but it remains to show that such an allocation exists, and to provide a way to compute it. To this end we exploit a close relationship between exact allocations and maximum flows in networks.

For a given set of agents $S \subseteq N$ and a piece of cake to be allocated X, define a graph $G(S, X)$ as follows. We keep track of a set of marks, which will be used to generate nodes in $G(S, X)$. First mark the left and right boundaries of all intervals that are contained in X. For each agent $i \in N$ and subinterval in U_i, mark the left and right boundaries of subintervals that are contained in $U_i \cap X$. When we have finished this process, each pair of consecutive markings will form an interval such that each agent will either uniformly value the entire interval or value none of the interval. In $G(S, X)$, create a node for each interval I formed by consecutive markings, and add a node for each agent $i \in N$, a source node s, and a sink node t. For each interval I, add a directed edge from source s to I with capacity equal to the length of the interval. Each agent node is connected to t by an edge with capacity $\text{avg}(S, X)$. For each interval-agent pair (I, i), add a directed edge with infinite capacity from node I to the agent i if agent i desires interval I.

For example, suppose $U_1 = [0, 0.25] \cup [0.5, 1]$ and $U_2 = [0.1, 0.4]$. If $X = [0, 1]$ then the interval markings will be $\{0, 0.1, 0.25, 0.4, 0.5, 1\}$. Agent 1 values $[0, 0.1]$, both agents value $[0.1, 0.25]$, agent 2 values $[0.25, 0.4]$, neither agent values $[0.4, 0.5]$ and agent 1 values $[0.5, 1]$. It holds that $\text{len}(D(\{1, 2\}, [0, 1])) = 0.9$. Average values are 0.75, 0.3 and 0.45 for sets $\{1\}, \{2\}$ and $\{1, 2\}$ respectively. See Figure 2 for an illustration of the induced flow network.

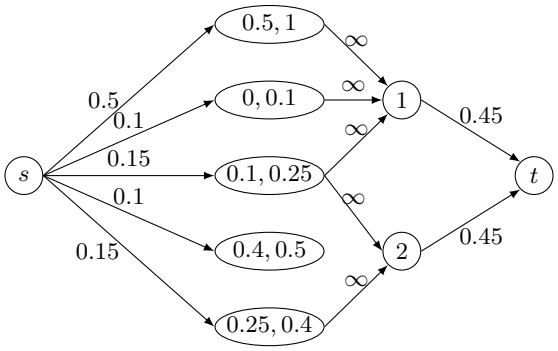


Figure 2: The flow network induced by the example.

Lemma 1. *Let $S \subseteq N$, and let X be a piece of cake. There is a flow of size $\text{len}(D(S, X))$ in $G(S, X)$ if and only if for all $S' \subseteq S$, $\text{avg}(S', X) \geq \text{avg}(S, X)$.*

Below we only prove the "if" direction, which is the one we need, using an application of the classic Max-Flow Min-Cut Theorem (see, e.g., (Cormen et al. 2001)).

Proof of "if". Assume that for all $S' \subseteq S$, $\text{avg}(S', X) \geq \text{avg}(S, X)$. By the Max-Flow Min-Cut Theorem, the minimum capacity removed from a graph in order to disconnect the source and sink is equal to the size of the maximum flow. The only edges with finite capacity in $G(S, X)$ are the ones that connect agent nodes to the sink, and the ones that connect the source to the interval nodes.

Construct a candidate minimum cut by disconnecting some set of agent nodes $T \subseteq S$ from the sink at cost $|T| \cdot \text{avg}(S, X)$ and then disconnecting all the (s, I) connections to interval nodes I desired by an agent $i \in S \setminus T$. This means that the total additional capacity we need to remove is $\text{len}(D(S \setminus T, X))$, the total length of intervals desired by at least one agent in $S \setminus T$. By assumption, this is at least $|S \setminus T| \cdot \text{avg}(S, X)$. As a result, this cut has capacity of at least $|T| \cdot \text{avg}(S, X) + |S \setminus T| \cdot \text{avg}(S, X) = |S| \cdot \text{avg}(S, X) = \text{len}(D(S, X))$. □

The following lemma establishes that this flow of size $\text{len}(D(S, X))$ in $G(S, X)$ is, in particular, characterizing an exact allocation. We omit the proof, which follows from the construction of the network.

Lemma 2. *Let $S \subseteq N$, and let X be a piece of cake. There exists an exact allocation with respect to S, X if and only if there exists a maximum flow of size $\text{len}(D(S, X))$ in $G(S, X)$.*

By combining Lemma 1 and Lemma 2 we see that the algorithm is indeed well-defined: if S has the smallest average then there exists an exact allocation with respect to S, X.[6] Moreover, we obtain a tractable algorithm for computing an exact allocation, by computing the maximum flow and deriving an exact allocation. A maximum flow can be

[6]Note that the network in Figure 2 does not satisfy the average minimality requirement and does not provide a corresponding exact allocation.

computed in time that is polynomial in the number of nodes, that is, polynomial in our input size (see, e.g., (Cormen et al. 2001)). We remark without proof that it is also possible to implement Step 2 of SUBROUTINE in polynomial time, using similar (but slightly more involved) network flow arguments. Therefore, Algorithm 1 can be implemented in polynomial time.

Truthfulness and fairness. Our main tool in proving that Algorithm 1 is truthful, proportional and EF is the following lemma (we omit its proof due to space constraints).

Lemma 3. *Let $S_1, \ldots, S_m$ be the ordered sequence of agent sets with the smallest average as chosen by Algorithm 1 and $X_1, \ldots, X_m$ be the ordered sequence of pieces to be allocated in calls to* SUBROUTINE*. That is, $X_1 = [0,1], X_2 = X_1 \setminus D(S_1, X_1), \ldots, X_m = X_{m-1} \setminus D(S_{m-1}, X_{m-1})$. Then for all $i > j$, $avg(S_i, X_i) \geq avg(S_j, X_j)$, and agents that are members of later sets receive weakly more in desired lengths.*

Envy-freeness now follows immediately from the lemma. Indeed, consider an agent $i \in N$. By "chosen" we mean that the agent was part of the subset with smallest average. The agent does not envy agents chosen in the same call to SUBROUTINE since all agents receive the same length in desired intervals and their valuations are piecewise uniform. By Lemma 3, the agent does not envy agents chosen in earlier calls because the amount agents receive weakly increases with each call. The agent does not envy agents chosen in later calls because all intervals desired by the agent are removed from consideration when the agent receives its allocation.

We provide a sketch of truthfulness, which follows by showing that an agent $i \in N$ has no incentive to change the choice of $S_{\min}$ and cannot profitably manipulate the exact allocation for a given $S_{\min}$.

1. Manipulations that change $S_{\min}$. Consider two subcases.
   (a) When i reports truthfully, $S_{\min} = S', i \notin S'$. An agent cannot affect $\text{avg}(T, X)$ if $i \notin T$, so the agent cannot cause some other $S'', i \notin S''$ to be chosen. The agent can cause $S'', i \in S''$, to be chosen, but then $\text{avg}(S'', X) \leq \text{avg}(S', X)$ and it follows from Lemma 3 that the agent does not gain.
   (b) When i reports truthfully, $S_{\min} = S', i \in S'$. Assume without loss of generality that $|S| \geq 2$. In this case, all agents in S', including i, receive exactly $\text{avg}(S', X) = k$ in intervals. Agent i can cause selection of some S'' by misstating its valuation. If $i \in S''$, then $\text{avg}(S'', X) \geq k$ for this to be profitable. If $i \notin S''$, then S'' was not chosen when i reports truthfully, so $\text{avg}(S'', X) \geq k$. In either case, agents $j \in S' \setminus \{i\}$ previously received k, but now receive at least k by observing that $\text{avg}(S'', X) \geq k$ and applying Lemma 3. Agent i receives at most $\text{len}(D(S', X))$ minus the intervals received by agents $j \in S' \setminus \{i\}$.[7] These agents receive weakly more if i manipulates, and thus, manipulations are not profitable.
2. Manipulations that change the exact allocation for a given $S_{\min}, i \in S_{\min}$. By definition each agent in $S_{\min}$ receives exactly $\text{avg}(S_{\min}, X)$ in desired intervals. If agent i decreases this value, it receives strictly less. If agent i increases this value by lying, then other agents receive more of the actual $D(S_{\min}, X)$, leaving less for agent i.

We omit the proof of proportionality due to lack of space, but it follows after establishing that no desired pieces are thrown away. Overall, we have the following theorem.

Theorem 4. *Assume that the agents have piecewise uniform valuation functions. Then Algorithm 1 is truthful, proportional, EF, and polynomial-time.*

Randomized Algorithms and Piecewise Linear Valuations

In the previous section we saw that designing deterministic truthful and fair algorithms is not an easy task, even if the valuation functions of the agents are rather restricted. In this section we shall demonstrate that by allowing randomness we can obtain significantly more general results.

A *randomized cake cutting algorithm* outputs a random allocation given the reported valuation functions of the agents. There are very few previous papers regarding randomized algorithms for cake cutting. A rare example is the paper by Edmonds and Pruhs (2006a), where they give a randomized algorithm that achieves approximate proportionality with high probability. We are looking for a more stringent notion of fairness. We say that a randomized algorithm is *universally proportional* (resp., *universally EF*) if it always returns an allocation that is proportional (resp., EF).

One could also ask for *universal truthfulness*, that is, require than an agent may never benefit from lying, regardless of the randomness of the algorithm. A universally truthful algorithm is simply a probability distribution over deterministic truthful algorithms. However, asking for both universal fairness and universal truthfulness would not allow us to enjoy the additional flexibility that randomization provides. Therefore, we slightly relax our truthfulness requirement. Informally, we say that a randomized algorithm is *truthful in expectation* if, for all possible valuation functions of the other agents, the expected value an agent receives for its allocation cannot increase by lying, where the expectation is taken over the randomness of the algorithm.

We remark that while truthfulness in expectation seems natural, fairness (i.e., proportionality and envy-freeness) is something that we would like to hold *ex-post*; fairness is a property of the specific allocation that is being made, and continues to be relevant after the algorithm has terminated. Interestingly enough, if we were to turn this around, then achieving universal truthfulness and envy-freeness/proportionality in expectation is trivial: simply allocate the entire cake to a uniformly random agent!

A randomized algorithm

In order to design a randomized algorithm that is truthful in expectation, universally proportional, and universally EF, we consider a very special type of allocation. In the following we will not require the free disposal assumption, that

[7] Lemma 3 also applies to agent i, but since it lies, it may receive intervals that are not desired and outside of $D(S', X)$.

is, we will consider partitions $X_1, \ldots, X_n$ of the cake such that $\bigcup_i X_i = [0, 1]$. We say that a partition $X_1, \ldots, X_n$ is *perfect* if for all $i, j \in N$, $v_i(X_j) = 1/n$. Consider the following randomized algorithm.

Algorithm 2 $(V_1, \ldots, V_n)$

1. Find a perfect partition $X_1, \ldots, X_n$.
2. Draw a random permutation π over N.
3. For each $i \in N$, set $A_i = X_{\pi(i)}$.

Lemma 5. *Algorithm 2 is truthful in expectation, universally proportional, and universally EF.*[8]

Proof. The fact that the algorithm is universally proportional and universally EF follows from the definition of perfect partitions: every agent has value $1/n$ for every piece!

We turn to truthfulness in expectation. The value an agent $i \in N$ obtains by reporting truthfully is exactly $1/n$. If agent i lies then the algorithm may choose a different partition $X'_1, \ldots, X'_n$. However, for any partition $X'_1, \ldots, X'_n$ the expected value of agent i when given a random piece is

$$\sum_{j \in N} \frac{1}{n} \cdot V_i(X'_j) = \frac{1}{n} \left(\sum_{j \in N} V_i(X'_j) \right) = \frac{1}{n},$$

where the second equality follows from the fact that the valuation functions are additive. □

Finding perfect partitions. Lemma 5 holds much promise, in that it is valid for all valuation functions. But there still remains the obstacle of actually finding a perfect partition given the valuation functions of the agents. Does such a partition exist, and can it be computed? More than two decades ago, Noga Alon (1987) proved that if the valuation functions of the agents are defined by the integral of a continuous probability measure then there *exists* a perfect partition; this is a generalization of his famous theorem on necklace splitting. Unfortunately, Alon's elegant proof is nonconstructive (which is unusual for a proof in combinatorics), and to this day there is no known constructive method under general assumptions on the valuation functions. This is not surprising since a perfect partition induces an EF allocation, and finding an EF allocation in a bounded number of steps for more than four agents is an open problem.

To obtain a computational method, we consider valuation functions that are *piecewise linear*. A valuation function V_i is considered piecewise linear if and only if its corresponding value density function v_i is piecewise linear on $[0, 1]$. Piecewise linear valuation functions are significantly more general than the class of piecewise constant valuation functions. A piecewise linear valuation function can be concisely represented by the intervals on which v_i is linear, and for each interval the two parameters of the linear function. The following lemma provides us with a tractable method of finding a perfect partition when the agents have piecewise linear valuation functions.

[8] Mossel and Tamuz (2010) make the same observation.

Lemma 6. *Assume that the agents have piecewise linear valuation functions. Consider the following procedure. We make a mark at 0 and 1, and for each agent $i \in N$ make a mark at the left and right boundaries of each interval where v_i is linear. Next, we divide each interval I_j between two consecutive marks into $2n$ consecutive and connected subintervals $I_j^1, \ldots, I_j^{2n}$ of equal length. For each such I_j and every $i \in N$ add the subintervals I_j^i and I_j^{2n-i+1} to X_i. Then the overall partition is perfect.*

The lemma's proof is omitted. By combining Lemma 6 with Lemma 5 we obtain the following result.

Theorem 7. *Assume that the agents have piecewise linear valuation functions. Then there exists a randomized algorithm that is truthful in expectation, universally proportional, universally EF, and polynomial-time.*

Discussion

We have made progress on truthful and fair algorithms for cake cutting. In unpublished work, we can also preclude simpler methods that make only contiguous allocations (and look closer to generalizations of the classic cut-and-choose algorithm) even for two agents both of whom are uniformly interested in a single (but different) subinterval. In future work we would like to generalize the deterministic algorithm to piecewise constant valuations and drop the free-disposal assumption. For practical settings, allowing more expressiveness (e.g., piecewise linear but a requirement that intervals are above some threshold length) seems important.

References

Alon, N. 1987. Splitting necklaces. *Advances in Mathematics* 63:241–253.

Brams, S. J., and Taylor, A. D. 1996. *Fair Division: From Cake-Cutting to Dispute Resolution*. Cambridge University Press.

Brams, S. J.; Jones, M. A.; and Klamler, C. 2006. Better ways to cut a cake. *Notices of the AMS* 53(11):1314–1321.

Brams, S. J.; Jones, M. A.; and Klamler, C. 2008. Proportional pie-cutting. *Int. Journal of Game Theory* 36(3–4):353–367.

Chevaleyre, Y.; Dunne, P. E.; Endriss, U.; Lang, J.; Lemaître, M.; Maudet, N.; Padget, J.; Phelps, S.; Rodríguez-Aguilar, J. A.; and Sousa, P. 2006. Issues in multiagent resource allocation. *Informatica* 30:3–31.

Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; and Stein, C. 2001. *Introduction to Algorithms*. MIT Press, 2nd edition.

Edmonds, J., and Pruhs, K. 2006a. Balanced allocations of cake. In *Proc. of 47th FOCS*, 623–634.

Edmonds, J., and Pruhs, K. 2006b. Cake cutting really is not a piece of cake. In *Proc. of 17th SODA*, 271–278.

Mossel, E., and Tamuz, O. 2010. Truthful fair division. Manuscript.

Nisan, N., and Ronen, A. 2001. Algorithmic mechanism design. *Games and Economic Behavior* 35(1–2):166–196.

Procaccia, A. D. 2009. Thou shalt covet thy neighbor's cake. In *Proc. of 21st IJCAI*, 239–244.

Robertson, J. M., and Webb, W. A. 1998. *Cake Cutting Algorithms: Be Fair If You Can*. A. K. Peters.

Thomson, W. 2007. Children crying at birthday parties. Why? *Journal of Economic Theory* 31:501–521.