A Discrete and Bounded Envy-Free Cake Cutting Protocol for Four Agents

Haris Aziz Simon Mackenzie
Data61 and UNSW
Sydney, Australia
{haris.aziz, simon.mackenzie}@data61.csiro.au

ABSTRACT

We consider the well-studied cake cutting problem in which the goal is to identify an envy-free allocation based on a minimal number of queries from the agents. The problem has attracted considerable attention within various branches of computer science, mathematics, and economics. Although, the elegant Selfridge-Conway envy-free protocol for three agents has been known since 1960, it has been a major open problem to obtain a bounded envy-free protocol for more than three agents. The problem has been termed the central open problem in cake cutting. We solve this problem by proposing a discrete and bounded envy-free protocol for four agents.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity; I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems; J.4 [**Computer Applications**]: Social and Behavioral Sciences - Economics

General Terms

Algorithms, Theory, Economics

Keywords

Fair Division, Elicitation Protocols, Multiagent Resource Allocation, Cake cutting

1. INTRODUCTION

Cake cutting is a metaphor for the allocation of a heterogeneous divisible good among multiple agents with possibly different preferences over different parts of the cake. Its main application is fair scheduling, resource allocation, and conflict resolution [13] and hence it has been extensively studied within computer science [27] and the social sciences [36]. Since various important divisible resources such as time and land can be captured by cake cutting, the problem of fairly dividing the cake is a fundamental one within the area of fair division and multiagent resource allocation [6, 17, 26, 29, 33, 35, 36].

Formally speaking, a cake is represented by an interval $[0, 1]$ and each of the n agents has a value function over pieces of the cake that specifies how much that agent values a particular subinterval. The main aim is to divide the cake fairly. In particular, an allocation should be envy-free so that no agent prefers to take another agent's allocation instead of his own allocation. Although an envy-free allocation is guaranteed to exist even with $n-1$ cuts [35][1], *finding* an envy-free allocation is a challenging problem which has been termed "one of the most important open problems in 20th century mathematics" by Garfunkel [16].

Motivation and Contribution.

Unlike allocation of indivisible items [12], the number of possible allocations in cake cutting is infinite. Since the valuations of agents over the subsets of the cake can be complex, it is not practiceable to elicit each agent's complete valuations function over the cake. A natural approach in cake cutting protocols to query agents about their valuations of different portions of the cake and based on these queries, propose an allocation. A cake cutting protocol is *envy-free* if each agent is guaranteed an envy-free piece if he reports his real valuations.

For the case of two agents, the problem has a well-known solution in the form of the *Divide and Choose* protocol: one agent is asked to cut the cake into equally preferred pieces and the other agent is asked to choose the preferred piece. The protocol even features in the Book of Genesis (Chapter 13) where Abraham divides the land of Canaan and Lot chooses first. In modern times, the protocol has been enshrined in the Convention of the Law of the Sea (Page 10, [6]). For the case of three agents, an elegant and bounded protocol was independently discovered by John L. Selfridge and John H. Conway around 1960 (Page 116, [6]). Since then, an efficient envy-free protocol for four or more agents has eluded mathematicians, economists, and computer scientists.

In 1995, Brams and Taylor [5] made a breakthrough by presenting an envy-free protocol for *any* number of agents [18]. Although the protocol is guaranteed to terminate in finite time, there is one critical drawback of the protocol: the running time or number of queries and even the number of cuts required is unbounded even for four

[1]The existence of an envy-free cake allocation can be shown via an interesting connection with Sperner's Lemma [35].

STOC'16, June 19–21, 2016, Cambridge, MA, USA
ACM. 978-1-4503-4132-5/16/06...$15.00
http://dx.doi.org/10.1145/2897518.2897522

agents. In other words, the number of queries required to identify an envy-free allocation can be arbitrarily large for certain valuations functions. If a protocol is not bounded, then its practicality is compromised [21]. Procaccia [27] terms unboundedness as a "serious flaw". Brams and Taylor were cognizant of their protocol's drawback and explicitly mentioned the problem of proposing a bounded envy-free protocol even for $n = 4$. Lindner and Rothe [21] write that "even for $n = 4$, the development of finite bounded envy-free cake-cutting protocols still appears to be out of reach, and a big challenge for future research." The problem has remained open and has been highlighted in several works [2, 5, 6, 10, 14, 31, 19, 26, 27, 22, 29, 30]. Saberi and Wang [30] term the problem as "one of the most important open problems in the field" and Lindner and Rothe [22] mention the case for $n = 4$ as "the central open problem in the field of cake-cutting". In this paper, *we present a discrete envy-free protocol for four agents that requires a bounded number of queries as well as cuts of the cake*. The maximum number of cuts required is 203.[2] Some of the techniques we use may be useful for cake cutting protocols with other properties or for more agents. In particular, we propose a new technique (called *permutation*) in which by suitably reallocating portions of a partial allocation that is envy-free, we ensure that some agent will not be envious of another agent *even* if the unallocated cake is given to the latter agent.

Related Work.

Cake cutting problems originated in the 1940's when famous mathematicians such as Banach, Knaster, and Steinhaus initiated serious mathematical work on the topic of fair division.[3] Since then, the theory of cake cutting algorithms has become a full-fledged field with at least three books written on the topic [3, 6, 29]. The central problem within cake cutting is finding an envy-free allocation [15, 33].

Since the earliest works, mathematicians have been interested in the complexity of cake cutting. Steinhaus [32] wrote that "Interesting mathematical problems arise if we are to determine the minimal number of cuts necessary for fair division." When formulating efficient cake cutting protocols, a typical goal is to minimize the number of cuts while ignoring the number of valuations queried from the agents. In principle, the actual complexity of a problem or a protocol depends on the number of queries. When considering how efficient a protocol is, it is useful to have a formal query model for cake-cutting protocols. Robertson and Webb [29] formalized a simple query model in which there are two kinds of queries: Evaluate and Cut. In an Evaluate query, an agent is asked how much he values a subinterval. In a Cut query, an agent is asked to identify an interval, with a fixed left endpoint, of a particular value. Although, the query model of Robertson and Webb is very simple, it is general enough to capture all known protocols in the literature. Note that if the number of queries is bounded, it implies that the number of cuts is bounded in the Robertson and Webb model. The protocol that we present in this paper uses a bounded number of queries in the Robertson and Webb model. Cake cutting protocols also provide an interesting connection between the literature on fair division and the field of communication complexity [20].

There is not too much known about the existence of a bounded envy-free protocol for $n \geq 4$ except that any envy-free cake-cutting algorithm requires $\Omega(n^2)$ queries in the Robertson-Webb model [25, 27]. Also, for $n \geq 3$, there exists no finite envy-free cake-cutting algorithm that outputs *contiguous* allocations [34]. Brams et al. [7] and Barbanel and Brams [4] presented envy-free protocols for four agents that require 13 and 5 cuts respectively. However, the protocols are not only unbounded but not even finite since they are *continuous* protocols that require the notion of a *moving knife*. An alternative approach is to consider known bounded protocols and see how well they perform in terms of envy-freeness [21]. Apart from the unbounded Brams and Taylor envy-free protocol for n agents, there are other general envy-free protocols by Robertson and Webb [28] and Pikhurko [24] that are also unbounded.

There are positive algorithmic results concerning envy-free cake cutting when agents have restricted valuations functions [13, 8] or when some part of the cake is left unallocated [30]. There has also been work on *strategyproof* cake cutting protocols for restricted valuation functions [1, 11, 23] as well as strategic aspects of protocols [9].

Structure of the Paper.

In Section 2, the formal model is presented. In Section 3, we present an envy-free protocol for three agents that serves as a warm-up for the case of four agents. In Section 4, we presents the main protocol. The section is divided into subsections in which three different protocols (*Post Double Domination Protocol*, *Core Protocol*, and *Permutation Protocol*) are described. These three protocols are used as building blocks to formulate the overall protocol.

2. PRELIMINARIES

Model.

We consider a cake which is represented by the interval $[0, 1]$. A *piece of cake* is a finite union of disjoint subsets of $[0, 1]$. We will make the standard assumptions in cake cutting. Each agent in the set of agents $N = \{1, \ldots, n\}$ has his own valuation function over subsets of interval $[0, 1]$. The valuations are (i) *defined on all finite unions of the intervals*; (ii) *non-negative*: $V_i(X) \geq 0$ for all $X \subseteq [0, 1]$; (iii) *additive*: for all disjoint $X, X' \subseteq [0, 1]$, $V_i(X \cup X') = V_i(X) + V_i(X')$; (iv) *divisible* i.e., for every $X \subseteq [0, 1]$ and $0 \leq \lambda \leq 1$, there exists $X' \subseteq X$ with $V_i(X') = \lambda V_i(X)$.

We will call an allocation *partial* if there is some cake that is unallocated. A partial envy-free allocation is a partial allocation that is envy-free. In order to ascertain the complexity of a protocol, Robertson and Webb presented a computational framework in which agents are allowed to make two kinds of queries: (1) for given $x \in [0, 1]$ and $r \in \mathbb{R}^+$, Cut query asks an agent to return a point $y \in [0, 1]$ such that $V_i([x, y]) = r$ (2) for given $x, y \in [0, 1]$, Evaluate query ask agent to return a value $r \in \mathbb{R}^+$ such that $V_i([x, y]) = r$. A cake-cutting protocol specifies how agents interact with queries and cuts. A protocol is *envy-free* if no agent is envious if he follows the protocol truthfully. All well-known

[2]Most of the 203 cuts are technically trims but in the Robertson and Webb model, any marking/trim on the cake is also treated as a proper cut.

[3]Hugo Steinhaus presented the cake cutting problems to the mathematical and social science communities on Sep. 17, 1947, at a meeting of the Econometric Society in Washington, D.C. [29, 32].

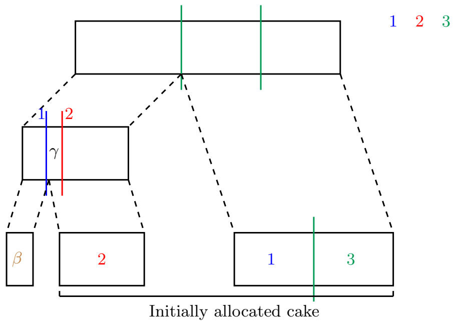

Figure 1: Example of a trim. Agents 1 and 2 trim their most preferred piece (the left most piece) to the value equal to that of their second most preferred piece. In this instance, agent 2 trims more than agent 1, hence his trim is to the right of agent 1's trim. Let us assume agent 1 and 3 each get a complete piece with 1 getting his second most preferred piece. Agent 2 is not envious of other agents if he gets the right side of the trimmed piece up till his trim. If agent 2 gets the part to the right of agent 1's trim instead of his own trim, then he is even happier. We will refer to this extra bit γ as the 'bonus' for agent 2. Agent 1 is still not envious of agent 2 if 2 gets the bonus.

cake cutting protocols can be analyzed in terms of number of queries required to return a fair allocation. A cake cutting protocol is *bounded* if the number of queries required to return a solution is bounded by a function of n *irrespective* of the valuations of the agents.

Terms and Conventions.

We now define some terms and conventions that we will use in the paper. Given a partial envy-free allocation of the cake and an unallocated residue β, we say that agent j *dominates* agent i if j does not become envious of i even if all of β were to be allocated to i. This concept has been referred to as i's *irrevocable advantage* in the cake cutting literature [6].

In the cake cutting protocols, we will describe, an agent may be asked to trim a piece of cake so that its value equals the value of a less valuable piece. Agents will be asked to trim various pieces of the cake so their remaining value is equal to the value of the third (or in some cases their second) most preferred complete piece. In Figure 1, we outline the idea of trimming a piece to equal the value of some other piece. When an agent trims a piece of cake, he will trim it from the left side: the main piece (albeit trimmed) will be on the the right side. The piece minus the trim will be called the partial main piece. The remainder will be referred to as the *residue*. If an agent trims a piece, we say he is *competing* for the piece. When we say an agent is *guaranteed* to get his second/third/etc most favoured piece, this guarantee is based on the *ordinal* preferences of the agents over the pieces. By ordinal, we mean that agents simply give a weak ordering over the pieces but do not tell the exact cardinal utility difference between two pieces. If an agent is indifferent between the top three pieces, then we will still say that the agent is guaranteed to get his third most valued piece.

We introduce notation to represent which agents have trimmed which pieces. So for example $123|1|2|3$ represents the scenario where one piece has three trims (by agents 1,2,3) in any possible order and the other three pieces have one trim each by one of the agents 1, 2, 3. We may enrich this notation further as follows: $1_1 2_1 3_1|1|2|3$ which represents that all agents think that the piece with the three trim marks is their most preferred or equivalently highest valued piece.

3. PROTOCOL FOR THREE AGENTS

We warm-up by presenting a protocol (Algorithm 1) for

Algorithm 1 Envy-free Protocol for 3 Agents.

```
1: Agent 3 divides the cake into 3 equally preferred pieces.
2: if 1 and 2 can each be given a different complete most preferred piece then
3:     give that complete piece to the agent who prefers it the most and give remaining piece to 3 and return.
4: else
5:     1 and 2 trim their highest valued piece from the left side to make the right side of the trim equal to the value of second most preferred piece (they simultaneously put trim marks).
6: end if
7: if 1 and 2 trim the same piece then
8:     consider $\beta_1$, the remainder from the left extreme of the piece to the leftmost trim. The partial piece $P_1^{1'}$ which is the most preferred piece except $\beta_1$ is given to the agent $i \in \{1,2\}$ who trimmed the piece more (let the other agent be $-i$). Let $\gamma_1$ be the part between the two trims of agent 1 and 2. Agent $-i$ gets his second highest valued complete piece $P_2^1$. Agent 3 gets the remaining complete piece. The unallocated cake is $\beta_1$.
9: end if
10: 3 cuts $\beta_1$ into 3 equally preferred pieces.
11: 1 and 2 trim their highest valued piece from the left side to make it equal to the value of second best.
12: if 1 and 2 can be each be given a different complete most preferred piece then
13:     give that complete piece to the agent who prefers it the most and give remaining piece to 3 and return.
14: else if 1 and 2 trim the same piece but $-i$ trims at least as much as $i$ then,
15:     give $-i$ the most preferred piece up till the leftmost trim, give $i$ a complete second most preferred piece, and give 3 the remaining complete piece.
16: else if 1 and 2 trim the same piece but $i$ trims more again then,
17:     let $\beta_2$ be the remainder from the left hand side to the first trim in $\beta_1$. The partial piece $P_1^{2'}$ which is the most preferred piece except $\beta_2$ is given to the agent $i$. Let $\gamma_2$ be the part between the two trims. Agent $-i$ gets his second highest valued complete piece $P_2^2$. Agent 3 gets the remaining complete piece. The only unallocated cake left if any is $\beta_2$.
18:     Since $i$ again got a partial piece, $i$ is asked to identify the lesser preferred $\gamma_j \in \{\gamma_1, \gamma_2\}$. Then $i$ gives $P_1^{j'}$ to agent $-i$ and gets $P_2^j$ in return.
19: end if
20: If some cake is still unallocated, Agent 1 and 2 perform Divide and Choose to allocate it.
```

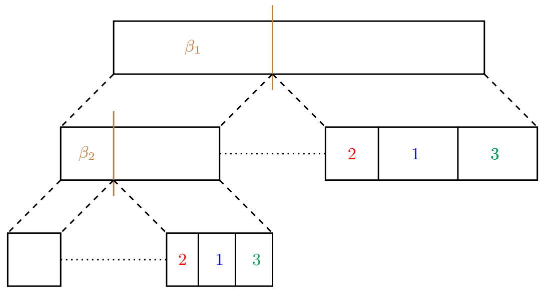

Figure 2: Protocol for 3 agent: In case agent 2 gets a trimmed piece two times, then we need to perform a permutation so that 1 gets a trimmed piece.

$n = 3$ which we will extend to $n = 4$.

Although the protocol requires more cuts than the Selfridge-Conway protocol, it bears similarities with it. It also depends on some ideas that we will exploit for our main protocol for four agents. The main idea of the protocol is that in each step, the cutter (agent 3) cuts the unallocated cake into 3 equally preferred pieces and gets one of the complete pieces. In each step, a partial envy-free allocation is maintained and then the remainder is again allocated which results in the remainder being fully allocated or a smaller remainder left. Note that when $i \neq 3$ is given the partial cake piece, agent 3 dominates i. When the remainder β_1 in the first step is divided among the agents, if now 3 dominates $-i$, then 3 does not care how β_1 is divided among i and $-i$ since he dominates both. So 3 is in this sense 'eliminated' from the protocol and we can perform Divide and Choose for 1 and 2 on the unallocated cake. If 3 again dominates i based on how β_1 is allocated, then we enforce a *permutation* or *reallocation* of some pieces of i and $-i$, so that 3 dominates $-i$ (see Figure 2). Both the ideas of domination and permutation will feature prominently in our protocol for four agents.

4. PROTOCOL FOR FOUR AGENTS

We now extend the ideas for the case of $n = 3$ to $n = 4$. The general flavour of the protocol is similar to that of the protocol for 3 agents. A designated cutter is asked to cut into equally preferred pieces. Based on some finer steps, we are able to achieve an envy-free allocation with possibly some cake still unallocated. We repeat the process on the remaining unallocated piece with the goal that the cutter dominates other agents just as we managed in the protocol for 3 agents. A few additional complications are introduced when dealing with 4 agents. Eliminating an agent would require being able to ensure that we can make an agent dominate all 3 others. Thankfully this is not necessary: we can show that we only need a protocol that ensures a given agent dominates 2 others. This is proved in the Double Domination Lemma.

4.1 Post Double Domination Protocol

We now present the Post Double Domination Protocol (Algorithm 2) that takes as input a partial envy-free allocation in which each agent dominates two agents and it returns a complete envy-free allocation.

Algorithm 2 Post Double Domination Protocol for 4 Agents

```
Input: A partial envy-free allocation and unallocated cake such that each agent dominates 2 other agents
Output: Envy-free complete allocation.
1: There exists some agent 1 who dominates two other agents say 3 and 4.
2: if 2 also dominates 3 and 4 then
3:     3 and 4 can divide the remainder by Divide and Choose.
4: else if 2 does not dominate 3 and 4 then
5:     it dominates 1 and one of 3 and 4 say 4.
6:     if 3 dominates 4 then
7:         give all the remainder to 4 (since everyone dominates 4).
8:     else if 3 does not dominate 4 and hence dominates 1 and 2 then
9:         then let 4 cut the residue into four equally preferred pieces, and agents $1, 2, 3$ pick their most preferred remaining piece in that order.
10:    end if
11: end if
12: return Envy-free complete allocation.
```

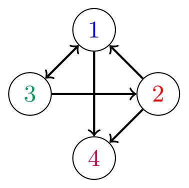

Figure 3: The domination graph of the final case in the proof of the Double Domination Lemma.

LEMMA 1 (DOUBLE DOMINATION LEMMA). *Suppose we have a bounded protocol which given a specified agent i and an unallocated piece of cake returns a partial envy-free allocation such that i dominates 2 other agents, then we can extend this into a 4 agent envy-free bounded protocol.*

PROOF. If we have a bounded protocol in which one agent can be made to dominate two other agents, then we simply run it at most 4 times on any unallocated cake to ensure that each agent dominates two other agents. If while doing this, the cake is completely allocated, we are already done. Otherwise, we can run the Post Double Domination Protocol (Algorithm 2). We now argue for the correctness of the protocol.

Assume 1 dominates 3 and 4. Now if 2 also dominates 3 and 4, then there exists a partial envy-free allocation in which even if all the residue is given to 3 or 4, then agent 1 and 2 will not be envious. All the residue can be divided among 3 and 4 using divide and choose. Agents 1 and 2 don't care because they dominate 3 and 4.

The other case is when 2 does not dominate 3 and 4. Without loss of generality assume that 2 dominates 1 and 4. Now if 3 dominates 4, we are already done because the whole residue can be given to 4 since everyone dominates 4. If 3 does not dominate 4 but dominates 1 and 2, then the domination graph looks like in Figure 3, an envy-free allocation can be found via the following method: agent 4 cuts the residue into equally preferred four pieces, and agents $1, 2, 3$ pick their most preferred remaining piece in that order. Agent 2 dominates 1 so does not care if 1 chooses first; agent 3 dominates 2 and 1 so that he does not care if 1 and 2 choose before him. □

Since we have shown that making an agent dominate two other agents is helpful, we will now explain how to achieve it. The overall protocol will first achieve double domination for each agent and if some cake is still unallocated, it will use the Post Double Domination Protocol. In order to get an agent to dominate other agents, we will repeat a *core protocol* multiple times which gives a partial envy-free allocation. The core protocol is explained in the next section.

4.2 Core Protocol

In the core protocol, a specified agent is asked to cut the cake into equally preferred pieces. The cutter gets a complete piece whereas the other agents may get partial pieces. The core protocol is recursively applied to the unallocated cake. After a bounded number of calls of the core protocol, we are in a position to do some reallocation so as to ensure that the specified cutter dominates two agents. We can then repeat this for another specified agent until each agent dominates two other agents.

We now give a high level description of the *core protocol*. Let us say that agent 4 divides the unallocated cake into 4 pieces. Agents 1, 2, and 3 are asked to trim the left hand side of their most preferred two pieces to make the right side of their trim equally valuable as their third most preferred cake piece. Each agent in set $\{1, 2, 3\}$ trims at most two pieces. In case an agent is indifferent between two or more pieces, we will still assume that the agent trims one piece to make it equal to the other piece. The trim in this extreme case is a trivial trim that coincides with the left edge of the cake. Hence the four pieces have a total of six trims. If an agent who trims a piece most is given that piece (an agent who trims most two pieces can choose which piece to get), up to his trim point, then the agent is envy-free. In fact the agent is not envious even if each other agent who gets a piece is given the piece up till the second rightmost trim. This approach is useful to get a partial envy-free allocation with some cake unallocated. In the core protocol, we do some extra work so that apart from the cutter, at least one more agent is given a complete piece. In order to do this, in a couple of cases, we may ask one or two carefully identified agents to additionally trim their most preferred piece to the value of their second most preferred piece in which case the previous trims of these agents are ignored. This is helpful is ensuring that at least two agents get complete pieces. It may be the case that some cake is left unallocated in which case the core protocol may be implemented on the unallocated cake again. The main thing we will prove is that we do not need to implement the core protocol on the unallocated cake unbounded number of times. For this, we first show in the Core Protocol Lemma that the core protocol returns a partial envy-free allocation with the additional useful property that the cutter and one other agent get complete pieces.

Lemma 2 (Core Protocol Lemma). *For $n = 4$, there exists a discrete and bounded protocol which returns partial envy-free allocation in which one agent cuts the cake into four equally preferred pieces and the cutter as well as at least one other agent gets one of these four pieces.*

Algorithm 3 Core Protocol for 4 agents that returns a partial envy-free allocation

```
Input:  Specified cutter agent (say agent 4)
Output: Partial envy-free allocation.
 1: Agent 4 is asked to cut the cake into 4 equal value pieces.
 2: Agents $1, 2, 3$ are asked to give their values for the 4 pieces.
 3: if each agent in $\{1, 2, 3\}$ can be given a most preferred piece then
 4:     Allocate each agent in $\{1, 2, 3\}$ a most preferred (complete) piece and the remaining complete piece to the cutter (agent 4).
 5:     return the envy-free allocation.
 6: end if
 7: Agents $1, 2, 3$ are asked to trim the left hand side of their most and second most preferred pieces to make them (the right side of the trim) equally valuable as their third most preferred cake piece.
 8: if no piece has exactly one trim then
 9:     if we are in a case $ij|jk|ik$ where $\{i, j, k\} = \{1, 2, 3\}$. then
10:         Since we have already covered the case in which each agent can be given a complete most preferred piece, we end up in situation $i_1j_1|j_2k|i_2k|$. Without loss of generality, the case is $i_1j_1|j_2k_1|i_2k_2|$. In this case $i$ and $k$ are asked to trim their most preferred piece to their second most preferred piece whereas agent $j$ is asked to trim his two most preferred pieces to equal his third most preferred piece. Agent $i$ and $k$'s trims up to their third most preferred piece are ignored. The effective trims look as follows now: $ij|jk|||$.
11:         if $j$ does not have the rightmost trim in both pieces then the right side of each piece with two trims is given to the agent who trimmed it the most. The piece is given up till the second rightmost trim. The remaining agent picks his most preferred complete unallocated piece and then 4 get the remaining unallocated piece.
12:         else if $j$ trimmed both the pieces the most then $j$ can choose which piece (up till the second rightmost trim) to get. The other piece with the trims is given to the agent with the second rightmost trim up till the second rightmost trim. The third non-cutter $i$ or $k$ gets his second most preferred piece completely. The last unallocated complete piece is given to agent 4.
13:         end if
14:     end if
15: else if we are in a case $ijk|ijk|||$ where $\{i, j, k\} = \{1, 2, 3\}$. then
16:     Ask each agent in $\{1, 2, 3\}$ to trim this first and second most preferred piece from the left side to make the right hand side to the value of his third most preferred piece. Each agent is given the piece he trims the most up till the second rightmost trim. If the same agent has the rightmost trims for both pieces, he chooses which piece to get. The agent with second rightmost trim gets the other piece up till the second rightmost trim. The third non-cutter gets his third most piece completely. The last unallocated complete piece is given to agent 4.
17: end if{Continued on next page...}
```

Proof. We argue that when agent 4 cuts the cake into equally preferred piece and then these pieces are partially allocated to the agents, then (1) the partial allocation is envy-free, and (2) the cutter and one other agent get complete pieces. If each agent in $\{1, 2, 3\}$ can be given a most preferred piece, then both conditions are trivially met. Otherwise, the algorithm distinguishes between the following cases: (1) no piece has exactly one trim; (2) exactly one piece has exactly one trim; (3) exactly two pieces have exactly one trim; (4) exactly three pieces have exactly one trim. In each of the cases, one non-cutter gets a complete piece and the cutter is also given an unallocated complete piece.

It remains to be shown that the partial allocation is envy-free. When an agent i gets a (possibly partial) piece a, he was the one who trimmed that piece the most. For each other piece b that is allocated, some other agent j trimmed b at least as much as agent i, i.e., j's trim in b was not left of i's trim. Hence i is not envious of j if j gets b up till j's trim from the right hand side. The reason is that i thinks that j's piece has more value than i's allocation only if j gets the right side of b beyond the trim of agent i. Thus, if i has the second rightmost trim in b, then i is not envious of j even if j gets the right side of b up till i's trim (e.g., see Figure 4). We note that in each of the four cases, when an agent i get a piece, he is not envious of another agent j because of the reason above. Moreover, if in piece a, the second rightmost trim is strictly to the left of i's trim in a or if there is no other trim in a, then i not only gets the right hand side of a up to i's trim but an additional bonus up till the second rightmost trim or the edge of the cake (whichever comes first). Hence, no agent i is envious of another agent j. □

Remark 1. *As soon as the agents' ordinal ranking of the four pieces cut by the cutter are known, it can be ascertained whether in the core protocol, an agent is guaranteed to get a piece of value equal to his third or second most preferred piece. Each agent gets a piece that is of same value as his third most preferred piece. An agent is guaranteed to get a second most preferred piece during the core protocol, if he is asked in the worst case to trim his most preferred piece to his second most preferred piece.*

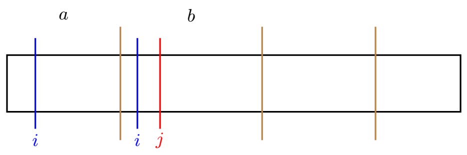

Figure 4: Example of a scenario where i has the rightmost trim for piece a and second rightmost trim of piece b whereas agent j has the rightmost trim for piece b. If i gets the right hand side of a up till his trim, and j gets the right hand side of b till i's trim in b, then i is not envious of j's allocation.

We make another observation about the outcome of the core protocol.

Lemma 3. *During the core protocol, when an agent i makes trims to equal his second or third most preferred piece respectively, i either gets such a piece completely or some other agent gets such a piece completely that i values as much as second or third most preferred piece respectively.*

```
18: if exactly one piece has exactly one trim then {The trims look like $i|ijk|jk|$}
19:     if the agent who trimmed it views it as his most preferred then
20:         Give the complete piece to him.
21:     else if the agent who trimmed it find it his second most preferred then
22:         ask him to trim his first most preferred to equal his second most preferred piece.
23:     end if
24:     The pieces are allocated up to the second rightmost trim to the agents who trimmed them most. If 2 pieces were trimmed most by the same agent, he decides which to get and the other is given to the agent who trimmed that piece second most.
25:     Give the last unallocated piece completely to the cutter (agent 4).
26: end if
27: if exactly 2 pieces have exactly one trim then {The trims look like $jk|ik|i|j$}
28:     Give those pieces with exactly one trim completely to the agents in $\{1, 2, 3\}$ who trimmed them if it is their most preferred.
29:     Agents who trimmed a piece with a single trim but do not value that piece most are asked to re-trim their most preferred piece up to their second most preferred piece. Their trims to make them equal to their third most preferred ignored from now on.
30:     The right hand side of the two pieces with the two trims are given to the agents with the rightmost trims. The pieces are allocated up till the second rightmost trim. If the 2 pieces were trimmed most by the same agent (agent $k$), he choses which to get and the other piece is given to whoever trimmed it second most.
31:     If one non-cutter has not been allocated a piece, he gets the most preferred piece among the two unallocated complete pieces.
32:     Give the last unallocated piece completely to the cutter (agent 4)
33: end if.
34: if exactly 3 pieces have exactly one trim then
35:     The trims look like $1|2|3|123$.
36:     If an agent most prefers the piece where he made a single trim, give him that complete piece.
37:     The ones who most prefer the piece with the three trims (call it $a$) compete for it by trimming up to their second most preferred piece. Their trims to make them equal to their most preferred ignored from now on.
38:     Cut $a$ at the second rightmost trim, and then allocate $a$ to whichever agent trimmed it most. The other agents get their second most preferred complete piece.
39:     Give the cutter (agent 4) the remaining complete piece.
40: end if
41: return envy-free allocation and any cake that is still not allocated.
```

Proof. Assume agent i is not the cutter in the core protocol and he was asked to trim his first and second most preferred pieces to equal his third most preferred piece. If agent i gets the complete piece that is his third most preferred piece, we are already done. Let us say that he got a partial piece. Then, at most one other agent got a partial piece. This means that some other agent j got a complete piece that is either agent i first, second or third most preferred piece.

Assume agent i is not the cutter in the core protocol and he was asked to trim his most preferred pieces to equal his second most preferred piece. This means that i got either his most preferred piece up to the value of the second most preferred piece or he got the second most preferred piece completely. If he got the second most preferred piece completely, we are done. If i got the most preferred piece up to the value of the second most preferred piece, then some other agent got a possibly partial piece from i's most preferred piece. But note that since i was asked to trim up to his second most piece in the protocol, only i was competing for his second most preferred piece. Hence, some other agent j got i's second most preferred piece completely. $\square$

The core protocol for partial envy-free allocation can be extended to obtain protocol for partial envy-free allocation with a single domination. If the core protocol is run again on the unallocated cake, we show in Lemma 4 that the cutter dominates at least one agent. When the core protocol is implemented, then the piece from which the highest valued (from the perspective of the cutter) residue is trimmed is called a *significant piece*. We will show that the cutter can be made to dominate an agent who got the significant piece. If residues from both pieces that are partially allocated are of the same value to the cutter, then we say that both non-cutters who got partial pieces were given significant pieces. In this case, a second run of the core protocol is enough for the cutter to dominate two agents.

LEMMA 4 (SINGLE-DOMINATION PROTOCOL LEMMA). *For $n = 4$, there exists a discrete and bounded protocol which returns an envy-free partial allocation in which one agent dominates another agent.*

PROOF. We run the core protocol a first time. This guarantees that the cutter (say agent 4) gets a complete piece (of value 1/4 of the whole cake) and that the residue is composed from the trims of at most 2 pieces. Since from the cutter's perspective all pieces were equal, the residue cannot sum up to more than 1/2 of the cake for him. Recall that the piece which agent 4 thinks was trimmed is the significant piece. The total residue is composed of the residue from the significant piece (call the residue β_1) and the residue from the other trimmed piece (let us call this residue β_2). The cutter thinks that he got $V_4(\beta_1)$ more value than the agent who got the significant piece. Since $V_4(\beta_1) \geq V_4(\beta_2)$, the value of the total residue from the cutter's perspective is at most $2V_4(\beta_1)$. If we run the core protocol again, at most two pieces are partial and hence the residue's value for the cutter is at most $2 \times 2V_4(\beta_1)/4 = V_4(\beta_1)$. This implies that even if the agent who got the significant piece gets all the residue which is of value $V_4(\beta_1)$ to the cutter, the cutter would still not envy him. This implies the cutter dominates the agent who got the significant piece. $\square$

Although the core protocol can be easily used to enable the cutter to dominate one agent, dominating two agents is more challenging. In the next section, we show how to overcome this challenge.

4.3 Permutation Protocol

If we run the core protocol repeatedly on the remaining unallocated cake, it may be that the cutter keeps dominating the same agent. We show that we only need to run the core protocol 5 times in total to achieve double domination. It may be that each time, the core protocol is run, the same agent gets the significant piece and hence the cutter dominates the same agent. If a different agent gets a significant part of the residue in any of the iterations, then the double domination is already achieved with one more iteration since from the cutter's perspective we have 2 agents who may be given all that is left of the cake without him being envious of them. If not, then we have one agent who ends up with the piece from which a significant trim was obtained for all iterations. The permutation lemma tells us that it is possible to give one of the 4 significant pieces to another agent while still preserving envy-freeness. This ensures that agent 1 ends up dominating two agents.

Algorithm 4 Permutation Protocol for 4 Agents

```
Input: An outcome of a core protocol in which one agent (say agent 4) is the cutter and another specified agent (say agent 1) gets the significant piece.
Output: An allocation in which each agent gets a piece equal to the value he trimmed to in the core protocol and in which agent 2 or 3 gets the significant piece.
1: if agent 2 was competing with someone for the piece and therefore had a trimmed piece then
     1. If the agent who made the second rightmost trim on agent 2's piece is agent 1, then we can simply permute agents 1 and 2 i.e., exchange their pieces.
     2. If the agent who made the second rightmost trim on agent 2's piece is agent 3, then we can move 3 to agent 2's piece. Agent 2 can be given 1's (trimmed) piece. Agent 1 can be given one of the complete pieces (which was given to 3 or 4). Agent 4 can be given the remaining complete piece.
2: else if agent 2 was in possession of a complete piece for which he was not competing with another agent then
     1. If 2's piece is the piece such that agent 1 trimmed up to that value in the core protocol, then simply permute 1 and 2.
     2. If 4 is holding the piece such that agent 1 trimmed up to that value in the core protocol then we simply move 4 to 2's piece since it is a complete piece and agent 1 gets 4's piece. Agent 2 is given 1's piece.
     3. If 3 has a complete piece such that agent 1 trimmed up to that value in the core protocol and 3 is indifferent between two pieces among his top 3 pieces, then 3 can be given another complete piece (such that he trimmed up to the value of that piece) of either 2 and 4. Agent 1 can be given 3's piece. Agent 2 gets 1's piece and 4 gets the remaining complete piece.
3: end if
```

We will use this idea in the argument of the Permutation Lemma. Before presenting the Permutation Lemma, we present another lemma that is useful for the proof of the Permutation Lemma.

LEMMA 5. *Consider m rows each with $m-1$ entries of positive reals. Then there exists at least one row such that for each entry in the row, the sum of other entries in the column corresponding to that entry is greater than or equal to the entry in the row.*

PROOF. It is sufficient to find a row in which each entry is not the unique maximum entry for that column. We go row by row and eliminate a row if it has at least one entry that is a maximal value among all entries in the corresponding column. Even if $m-1$ rows are eliminated, we are left with one row in which each entry is not the unique maximum entry for that column. □

We will rely on Lemma 5 while reasoning about when reallocating pieces does not cause any envy. In particular let us say that an agent gets slightly more than the value he wanted to guarantee. He may not want to let go of this extra value lest it leads to him being envious. However, let us say he gets similar extra values again, then we may ask the agent to choose which one of the extra values he rates least and give this extra value to some other agent. The other extra values, make up for this loss. Intuitively, Lemma 5 will help identify that if we have enough subcases (rows) then there will be one row on which an agent will be happy to compromise.

We are now in a position to present the Permutation Lemma.

LEMMA 6 (PERMUTATION LEMMA). *There exists a discrete and bounded protocol for 4 agents that returns a partial envy-free allocation in which one agent dominates two other agents.*

PROOF. Assume that agents 1, 2, 3, 4 get pieces p_1, p_2, p_3, and p_4 respectively with 4 getting complete pieces. When 4 cut the pieces, p_1 is a piece P_1 without the part left of the second rightmost trim. Now, assume that in four iterations of the 1 gets the significant piece and we want to reallocate so that some other agent among 2 and 3 gets the significant piece. We need to show that when the reallocation is done then barring the bonus part that agents get in the core protocol, the agents get at least as preferred a piece. If another agent aside from agent 1 gets the significant piece then we are done. If agent 1 is repeatedly getting it, then we zoom in to a case to permute it i.e., reallocate some of the pieces so as to make sure that an agent other than 1 is dominated. If agent 1 is repeatedly getting the significant piece and there is no reallocation in which instead of agent 1 some other agent gets the significant piece, this means that either (a) agent 1 likes the bonus from his significant piece so much that he is not willing to take some other piece or (b) for some other agent j, the bonus corresponding to the significant piece is not enough for j to be attracted towards the significant piece.

CLAIM 1. *For a partial allocation as a result of the core protocol in which agents make trims, a reallocation can be done in which some agent other than 1 gets the significant piece and each agent gets a piece of value corresponding to his original trims but in which he may lose out on the additional bonus due to the second rightmost trims.*

PROOF. The algorithm to perform the reallocation is stated as the Permutation Protocol (Algorithm 4). Imagine that agent 1 is holding the significant piece. For the piece to be significant, another agent must have been competing with agent 1 for it. If no other agent was competing for the piece, then agent 1 would have got the whole piece and hence the piece would not be significant.

Let us say that the agent who competes for the same piece is agent 2. Since the piece was cut up to the second rightmost trim, agent 2 is not envious of any other agent if agent 2 gets the piece (while not getting his trim).

We now distinguish between 2 possibilities. We show how both cases can be handled.

1. **Agent 2 was competing with someone for the piece and therefore had a trimmed piece.** We distinguish between two subcases.
   (a) If the agent who made the second rightmost trim on agent 2's piece is agent 1, then we can simply permute agents 1 and 2 i.e., exchange their pieces.
   (b) If the agent who made the second rightmost trim on agent 2's piece is agent 3, then we can move 3 to agent 2's piece. Notice that 3 must have been allocated a complete piece since only 2 pieces may be trimmed. This means that 1 did not compete with 3 for 3's piece which implies that if 1 can get 4 or 3's piece he will get the piece he was guaranteed before the order of the trim was determined (either second most preferred or third most preferred). Agent 2 can be given 1's (trimmed) piece. Agent 1 can be given one of the complete pieces (which was given to 3 or 4). Agent 4 can be given the remaining complete piece.

2. **Agent 2 was in possession of a complete piece for which he was not competing with another agent.** Let us focus on the piece agent 1 desires and was guaranteed to get a piece of that value (either his second most preferred or third most preferred). We will use the fact that by Lemma 3, some other agent got a complete piece that was of at least as much value to agent 1. We distinguish between three subcases.
   (a) If 2's piece is the piece agent 1 desires, then simply permute 1 and 2.
   (b) If 4 is holding the piece to be freed then we simply move 4 to 2's piece since it is a complete piece and agent 1 gets 4's piece. Agent 2 is given 1's piece.
   (c) Let us assume that 3 is holding the piece to be freed and agent 1 is guaranteed to get a piece of a value as much as this complete piece (which we refer to as a).

   If 3 is indifferent among any two of his most preferred three pieces, then either he is indifferent among the top 2 or among the second and third. For the former, if he got a top 2 piece, he will be willing to get another top 2 piece and if he got a third piece, he would certainly be happy to get one of the complete top 2 pieces. For the latter, if he got one of the second or third preferred pieces, he would be happy to get the other equally preferred one. If he had got the top piece, then this means that only 3 had a proper trim on a. But this means that 3 is willing to move to another complete piece with as much value barring the extra bonus he got in piece a. Hence in all the cases above, 3 can be given another complete piece, agent 1 can be given 3's complete piece, and 2 can get 1's partial piece.

We now assume that 3 is not indifferent among any of his 3 most preferred pieces. We show that in this scenario, we would already be in one of the previous cases in which 1 is okay with taking 2 or 4's piece.

We first argue that agent 3's most preferred piece is the significant piece. Piece a cannot be agent 3's most preferred piece since otherwise 1 would not be guaranteed to get it: if 1 got it, 3 would be envious. Neither can the piece held by 4 or 2 be 3's most preferred piece since envy-freeness of the partial allocation will be violated. Since the pieces allocated to 2, 3 and 4 are not 3's most valued pieces, it implies that agent 3's most preferred piece is the significant piece.

Next, we argue that 3 was allocated his second most preferred piece in which had put a proper trim. First we assume that 3 trimmed up to his third most preferred piece. If a is 3's third most preferred piece, then 3 would have been envious of either 2 or 4 whoever got 3's most preferred piece. 3 would have competed for such as piece and such a piece would not have been allocated completely. This means that 3 got his second most preferred piece. Second, we assume that 3 trimmed unto his second most preferred piece, then this means 3 was allocated his second most preferred piece since he was guaranteed it. In both cases 3 put a proper trim on a.

We now argue that a (3's piece) is also agent 1's second most preferred piece. If the piece allocated to 2 or 4 was agent 1's second most preferred piece, we would be in a different case.

Therefore a is agent 1's and 3's second most preferred piece. Since 1 is guaranteed to get a piece of value of a and since a is agent 1's second most preferred piece, it means that only 1 has a proper trim on a. But we have already shown that 3 also put a proper trim on a. But if two agents put a proper trim on a piece, the piece cannot be completely allocated to an agent hence a contradiction.

This complete the proof of the claim. □

We have shown that reallocation can be done in which each agent gets a piece of value corresponding the trims the agents made.

Note that in each iteration of protocol, each agent gets some (possibly non-zero) bonus. Since agent 1 gets a significant piece, he gets a non-zero bonus each time. Although 1 may object to any reallocation for a particular iteration of core protocol, we show that he will not envious if one particular iteration of the core protocol is identified in which corresponds to the row of entries in Table 1 in which for each entry in the row, the sum of other entries in the column corresponding to that entry is greater than or equal to the entry in the row. By Lemma 5, such a row exists. This means that even if each agent loses his bonus because of the permutation, he gets enough bonus in the other iterations to make for this loss so that there is no envy.

We have shown so far that in four iterations of the core protocol, the partial envy-free allocation is such that two different agents got a significant piece in one of the calls of the core protocol. If the same agents get a significant piece in all the first four iterations of the core protocol, then we have shown above that the permutation protocol can implemented to give the significant piece to another agent and still not maintain envy-free across the four iterations of the core protocol. The fifth iteration of the core protocol ensures that the cutter dominates two agents and the partial allocation is envy-free. □

	1	2	3	4
Bonus in 1st iteration	b_1^1	b_2^1	b_3^1	0
Bonus in 2nd iteration	b_1^2	b_2^2	b_3^2	0
Bonus in 3rd iteration	b_1^3	b_2^3	b_3^3	0
Bonus in 4th iteration	b_1^4	b_2^4	b_3^4	0

Table 1: Bonus of each agent in the first 4 calls of the core protocol with agent 4 as the cutter.

4.4 Overall Protocol

In the previous sections, we built the building blocks for our overall protocol: Post Double Domination Protocol, Core Protocol, and Permutation Protocol. We are now in a position to formalize the protocol to compute a complete envy-free allocation and presents the main result. The protocol is formalized as Algorithm 5.

Algorithm 5 Discrete and Bounded Envy-free Protocol for 4 Agents.

```
1: while some agent $i$ does not dominate two other agents and there is still some unallocated cake do
2:     Run the Core Protocol (Algorithm 3) 4 times on the unallocated cake with $i$ as the cutter. Return at any point if there is no cake left unallocated. {After two iterations, $i$ already dominates one agent}
3:     if the same agent (say agent $j$) gets the significant piece in each of the 4 calls of the core protocol then
4:         Identify in which call of the protocol, the non-cutter agents get less bonus than the sum of bonuses in the other calls of the core protocol. {see Table 1}
5:         Implement reallocation via the Permutation Protocol (Algorithm 4) for pieces allocated by this particular call of the core protocol where $i$ is the cutter and $j$ gets the significant piece.
6:     end if
7:     Run core protocol in the unallocated cake if some cake is still unallocated.
8: end while
9: if there is some unallocated cake then
10:    Run Post Double Domination Protocol (Algorithm 2) on the remaining cake.
11: end if
12: return envy-free complete allocation.
```

THEOREM 2. *For four agents, there exists a discrete and bounded envy-free protocol that requires constant number of queries in the Robertson and Webb model.*

Proof. The protocol is formalized as Algorithm 5. The theorem follows from Lemmas 1 and 6. The protocol first ensures that there exists a partial envy-free allocation in which each agent dominates two other agents. The protocol then used the Double Domination Protocol to obtain a complete envy-free allocation.

We now argue in more detail that the protocol is bounded not only in terms of number of cuts but also in terms of Robertson-Webb queries.

In each run of the core protocol, the cutter makes three cuts and then each non-cutter makes at most two trims. Two agents may be asked to replace their cuts and make a cut to make their most preferred piece equal to their second most preferred piece or in another case agents perform Divide and Choose which requires 1 cut. Hence, in the core protocol, there are at most 3+6+1=11 cuts. Since we run 5 iterations of the core protocol to get one double domination, we need 50 cuts to get one double domination. Since we need each agent to double dominate, we need $50 \times 4 = 200$ cuts. After we achieve double dominations for all agents, we need at most three more cuts. So we need at most 203 cuts.

We now count the maximum number of Robertson-Webb query operations required during the protocol. The cutter is asked the value of the whole cake (1 query), and then made to cut to equal 1/4 of the value (3 queries). The non-cutter agents are asked to give values of the pieces ($3 \times 4 = 12$ queries.) They are then asked to trim to equal the value of the third most preferred piece (3×2 queries). Two agents may be asked to make one additional cut to make their most preferred piece equal to their second most preferred piece (2×1 queries) or in another case, agents perform Divide and Choose which requires 4 queries. Hence the core protocol requires in total $1 + 3 + 12 + 6 + 4 = 26$ queries. The core protocol is run 5 times so that takes 26×5 queries. In addition to running the core protocol 5 times, we also need to run permutation protocol four times to check which permutation to implement. In each call of the permutation protocol, agents are queried about the amount of bonus which they get in the core protocol which takes additional 3 queries so in total of 12 queries for the permutation protocol. In order to get all double dominations, we require $((26 \times 5) + 12) \times 4 = 568$ queries. After we achieve double dominations, we require maximum 16 more queries. So the total number of queries is 584. □

5. DISCUSSION

In this paper, we proposed the first bounded and envy-free protocol for four agents. Some of our insights such as the one of exploiting the bonus cake given to the agents as well as analyzing the domination graph may be useful in attacking the problem for *any* number of agents. Our protocol is based on three main ingredients: core protocol, permutational protocol, and post double domination protocol. The higher level ideas of our overall protocol could be applied to general problem for more number of agents. Some of the proof ideas may be generalizable to more than four agents. For example, a suitable generalization of the core protocol is feasible. On the other hand, the Permutation Protocol appears to be challenging to extend to arbitrary number of agents and will require more interesting insights and techniques. There are also some other interesting directions for future research such as existence and properties of equilibria under our new protocol.

6. ACKNOWLEDGMENTS

Data61 is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program. Part of the research was carried out when the authors were visiting LAMSADE, University Paris Dauphine in May 2015. The authors thank Rediet Abebe, Steven Brams, Simina Brânzei, Ioannis Caragiannis, Katarína Cechlárová, Serge Gaspers, David Kurokawa, and Ariel Procaccia for useful comments. They also thank the reviewers of STOC 2016 for suggestions to improve the presentation. Haris Aziz thanks Ulle Endriss for introducing the subject to him at the COST-ADT Doctoral School on Computational Social Choice, Estoril, 2010.

7. REFERENCES

[1] H. Aziz and C. Ye. Cake cutting algorithms for piecewise constant and piecewise uniform valuations. In *Proceedings of the 10th International Workshop on Internet and Network Economics (WINE)*, pages 1–14, 2014.

[2] B. Barbanel and A. D. Taylor. Preference relations and measures in the context of fair division. *American Mathematical Monthly*, 123(7):2061–2070, 1995.

[3] J. B. Barbanel. *The Geometry of Efficient Fair Division.* Cambridge University Press, 2005.

[4] J. B. Barbanel and S. J. Brams. Cake division with minimal cuts: envy-free procedures for three persons, four persons, and beyond. *Mathematical Social Sciences*, 48(3):251–269, 2004.

[5] S. J. Brams and A. D. Taylor. An envy-free cake division protocol. *The American Mathematical Monthly*, 102(1):9–18, 1995.

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

[7] S. J. Brams, A. D. Taylor, and W. S. Zwicker. A moving-knife solution to the four-person envy-free cake division. *Proceedings of the American Mathematical Society*, 125(2):547–554, 1997.

[8] S. Branzei. A note on envy-free cake cutting with polynomial valuations. *Information Processing Letters*, 115(2):93–95, 2015.

[9] S. Branzei and P. B. Miltersen. A dictatorship theorem for cake cutting. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 482–488. AAAI Press, 2015.

[10] C. Busch, M. S. Krishnamoorthy, and M. Magdon-Ismail. Hardness results for cake cutting. *Bulletin of the EATCS*, 86:85–106, 2005.

[11] Y. Chen, J. K. Lai, D. C. Parkes, and A. D. Procaccia. Truth, justice, and cake cutting. *Games and Economic Behavior*, 77(1):284–297, 2013.

[12] R. Cole and V. Gkatzelis. Approximating the nash social welfare with indivisible items. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 371–380. ACM Press, 2015.

[13] X. Deng, Q. Qi, and A. Saberi. Algorithmic solutions for envy-free cake cutting. *Mathematics of Operations Research*, 60(6):1461–1476, 2012.

[14] J. Edmonds and K. Pruhs. Cake cutting really is not a piece of cake. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 271–278, 2006.

[15] G. Gamow and M. Stern. *Puzzle-math*. Macmillan, 1958.

[16] S. Garfunkel. For all practical purposes social choice. *COMAP*, 1988.

[17] R. Guo. *Cross-Border Management: Theory, Method and Application*. Springer, 2015.

[18] W. Hively. Dividing the spoils. *Discover Magazine*, March, 1995.

[19] D. Kurokawa, J. Lai, and A. D. Procaccia. How to cut a cake before the party ends. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI)*, pages 555–561. AAAI Press, 2013.

[20] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 2006.

[21] C. Lindner and J. Rothe. Degrees of guaranteed envy-freeness in finite bounded cake-cutting protocols. In *Proceedings of the 5th International Workshop on Internet and Network Economics (WINE)*, volume 5929 of *Lecture Notes in Computer Science (LNCS)*, pages 149–159. Springer-Verlag, 2009.

[22] C. Lindner and J. Rothe. Cake-cutting: Fair division of divisible goods. In J. Rothe, editor, *Economics and Computation: An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*, chapter 7. Springer-Verlag, 2015.

[23] A. Maya and N. Nisan. Incentive compatible two player cake cutting. In *Proceedings of the 8th International Workshop on Internet and Network Economics (WINE)*, volume 7695 of *Lecture Notes in Computer Science (LNCS)*, pages 170–183. Springer-Verlag, 2012.

[24] O. Pikhurko. On envy-free cake division. *The American Mathematical Monthly*, 107(8):736–738, 2000.

[25] A. D. Procaccia. Thou shalt covet thy neighbor's cake. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 239–244. AAAI Press, 2009.

[26] A. D. Procaccia. Cake cutting: Not just child's play. *Communications of the ACM*, 56(7):78–87, 2013.

[27] A. D. Procaccia. Cake cutting algorithms. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. D. Procaccia, editors, *Handbook of Computational Social Choice*, chapter 13. Cambridge University Press, 2016.

[28] J. M. Robertson and W. Webb. Near exact and envy-free cake division. *Ars Combinatorica*, 45:97–108, 1997.

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

[30] A. Saberi and Y. Wang. Cutting a cake for five people. In *Proceedings of the 5th International Conference on Algorithmic Aspects in Information and Management (AAIM)*, Lecture Notes in Computer Science (LNCS), pages 292–300. Springer-Verlag, 2009.

[31] E. Segal-Halevi, A. Hassidim, and Y. Aumann. Waste makes haste: Bounded time protocols for envy-free cake cutting with free disposal. In *Proceedings of the 14th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 901–908. IFAAMAS, 2015.

[32] H. Steinhaus. The problem of fair division. *Econometrica*, 16:101–104, 1948.

[33] I. Stewart. Division without envy. *Scientific American*, 1999.

[34] W. Stromquist. Envy-free cake divisions cannot be found by finite protocols. *The Electronic Journal of Combinatorics*, 15(R11), 2008.

[35] F. E. Su. Rental harmony: Sperner's lemma in fair division. *American Mathematical Monthly*, 10:930–942., 1999.

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