Computing Socially-Efficient Cake Divisions

Yonatan Aumann
Department of Computer Science
Bar-Ilan University
Ramat-Gan 52900, Israel
aumann@cs.biu.ac.il

Yair Dombb
Department of Computer Science
Bar-Ilan University
Ramat-Gan 52900, Israel
yairdombb@gmail.com

Avinatan Hassidim
Department of Computer Science
Bar-Ilan University
Ramat-Gan 52900, Israel
avinatanh@gmail.com

ABSTRACT

A frequent task facing a MAS designer is to efficiently divide resources amongst multiple agents. We consider a setting in which a single divisible resource, a.k.a. "cake", needs to be divided amongst n agents, each with a possibly different valuation function over pieces of the cake. For this setting, we address the problem of finding divisions that maximize the social welfare, focusing on divisions where each agent gets a single contiguous piece of the cake. We provide a constant factor approximation algorithm for the problem, and prove that it is NP-hard to find the optimal division, and that the problem admits no FPTAS unless P=NP. These results hold both when the full valuations of all agents are given to the algorithm, and when the algorithm has only oracle access to the valuation functions. In contrast, if agents can get multiple, non-contiguous pieces of the cake, the results vary greatly depending on the input model. If the algorithm is provided with the full valuation functions of all agents, then the problem is easy. However, if the algorithm needs to query the agents for information on their valuations, then no non-trivial approximation (i.e. $< n$) can be guaranteed.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems; F.2.2 [**Nonnumerical Algorithms and Problems**]: Computations on discrete structures

General Terms

Algorithms, Theory

Keywords

Cake cutting, multiagent resource allocation, welfare maximization

1. INTRODUCTION

Consider a group of agents sharing a common resource, e.g. deep sea exploration robots sharing a sonar device, Mars rovers sharing a drilling tool, or astronomers sharing the Hubble telescope. The utility obtained from using the resource may vary depending on the time and the duration it is used, and between agents. Furthermore, some agents may prefer the resource at one time while others prefer it at a different time (e.g. some prefer Hubble in winter while others gain more from it in the summer). The basic question we address here is how to divide the use of such a resource between the different agents so as to maximize the total utility obtained from the resource? We consider this problem both in the setting where agents need to get the resource for one, uninterrupted interval, and for the case where agents may make use of the resource by combining multiple nonconsecutive time intervals.

A natural setting for analyzing the above problem is that of *cake cutting*, where a single divisible good needs to be divided between several agents with possibly different preferences regarding the different parts of the good, or "cake". The cake cutting problem was first introduced in the 1940's, where the original goal was to give each of the n agents "their due part", i.e. a piece worth at least $\frac{1}{n}$ of the entire cake by their own measure. Since then, other objectives have also been considered, with the majority of them requiring that the division be "fair", under some definition of fairness (e.g. envy-freeness).

Here, we address the fundamental problem of *maximizing social welfare* in cake cutting. Given a shared resource and the valuation functions of the agents for this resource, the problem is to find an allocation that maximizes the utilitarian welfare, i.e. the sum of individual utilities. Much of our focus is on the case where each agent needs to get a *single contiguous piece* of the good. The contiguity requirement is natural in many settings, e.g. dividing time (as in the examples above), spectrum, and real-estate.

1.1 Results

We study the problem of maximizing the utilitarian welfare in cake cutting. We consider two models of input. In the first, the algorithm can only query the agents about their valuations; in the second, the valuations are given explicitly to the algorithm, and are assumed to be piecewise-constant.

We show that for the setting where each agent's piece must be contiguous, the problem of maximizing utilitarian welfare is NP-hard in the strong sense, in both input models. This result also implies that there is no FPTAS for the problem, unless P=NP. We furthermore show that this hardness result holds even when the agents' valuations are from the more-restricted class of piecewise-uniform functions.

For the model in which the algorithm gets the full valuations from the agents, we provide an 8-approximation algorithm that runs in polynomial time, and show that the

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013), Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.*

problem of finding the optimal division is fixed-parameter tractable. Both algorithms can also be adapted for the model in which the algorithm needs to query the agents; in this case, we lose a factor of $(1+\epsilon)$ in the obtained welfare, where ϵ is a precision parameter also given to the algorithm.

Finally, we consider the case where the contiguity requirement is dropped, i.e. each agent may get *any collection* of intervals. Unlike with connected pieces, we show in this setting that the situation varies greatly depending on the model of input. When the valuations are given explicitly to the algorithm (and are piecewise-constant), the problem can be easily solved in polynomial time. However, if the algorithm has only oracle access to the valuations, it is impossible to do any better than an n-factor approximation.

1.2 Related Work

Cake cutting problems were introduced in the 1940's [21], and were studied in many variants since then. Originally, the goal was to divide the cake *fairly*. Many algorithms were proposed for this problem, including a number of "moving knife" algorithms, which perform an infinite number of valuations by continuously moving a knife over the cake (for some examples, see [22, 14] and [5]). In addition to the algorithmic results, there has also been work on existence theorems [13, 22], lower bounds for the complexity of such algorithms [20, 23, 18], and a number of books on the subject, e.g. [6, 19].

In more recent years, the issue of social welfare in cake cutting has also begun to receive some attention. The study of this subject was initiated by Caragiannis et al. [7] which aimed to quantify the degradation in social welfare that may be caused by different fairness requirements; the same question was studied for connected pieces in [2]. Some other works that have studied the social efficiency of cake cutting are Zivan [24], which suggests a way to trade some of the fairness for more efficiency, and the very recent work by Maya and Nisan [17], which studies the welfare attainable by truthful cake cutting mechanisms (without money) for two agents and piecewise-uniform valuations. Also related are the works of Guo and Conitzer [15] and Han et al. [16], which study the welfare achievable by truthful mechanisms for dividing a set of divisible goods, a setting close (but not identical) to cake cutting.

Especially relevant to our work are the works by Cohler et al. [11] and Bei et al. [4]. These papers study the problem of maximizing utilitarian welfare while maintaining some fairness requirement; both works consider a model in which the algorithm has the full valuation functions of the agents. [11] studies the problem of finding a division that maximizes the welfare among all envy-free, non-connected divisions. They show that for piecewise-constant valuations the problem can be solved in polynomial time using an LP, and provide approximation algorithms for more general valuation functions. [4] studies the problem of finding a division that maximizes the welfare among all proportional divisions. Their work is closer to ours in that it focuses on connected pieces; there is also some similarity in the results, as they show that this problem is NP-hard. Interestingly, [4] also shows hardness of approximation for a factor $O(\sqrt{n})$ (with non-normalized valuations), even if all valuation functions have only two "steps"; this is in contrast to our results, showing that without the proportionality requirement, the optimal welfare can be approximated within a constant.

Our work here can be also viewed as a part of the large body of work on the problem of multiagent resource allocation. This line of work studies the problem of welfare maximization in different allocation settings; for a survey of this literature, see [10].

2. MODEL AND DEFINITIONS

Valuation Functions. In our model, the cake is represented by the interval $[0,1]$. Each agent $i \in [n]$ (where $[n] = \{1,\dots,n\}$) has a non-atomic (additive) measure $v_i(\cdot)$, mapping each measurable subset of $[0,1]$ to its value according to agent i. For most of this work, we are only interested in a value of *intervals* in $[0,1]$, and thus simply write $v_i(a,b)$ for the value of the interval between a and b. (Note that since v_i is non-atomic, single points have no value, and we need not worry about the boundary points a and b themselves.)

An additional common assumption in the cake-cutting literature is that the valuations are *normalized*, i.e. that $v_i(0,1) = 1$ for all i. We present our results here assuming that this indeed holds. However, we stress that our results hold (with small modifications to the algorithms or complexity) for arbitrary valuations as well.

Social Welfare. A social welfare measure is a mapping from each possible vector of the agents' individual utilities to some real number, aiming to measure how good each division is for the *whole society*. Let x be a division (to be formally defined shortly); we write $u_i(x)$ to express the value agent i obtains from the piece she receives in x. We focus on the utilitarian welfare $u(x) = \sum_{i\in[n]} u_i(x)$, which is simply the sum of the agents' individual utilities.

Connected Divisions. In this work, we put most focus on divisions in which every agent gets a *single interval* of the cake. Formally, a connected division of the cake $[0,1]$ between n agents can be defined as a vector

$$x = (x_1,\dots,x_{n-1},\pi) \in [0,1]^{n-1} \times S_n$$

(where S_n is the set of all the permutations of $[n]$), having $x_1 \le x_2 \le \cdots \le x_{n-1}$. This is interpreted as making $n-1$ cuts in positions $x_1,\dots,x_{n-1}$, and allocating the n resulting intervals to the agents in the order determined by the permutation π. Note that the space X of all such divisions is compact; since the utilitarian welfare is continuous in X (as the agents' valuation functions are all non-atomic), we get that $u(x)$ obtains a maximum over X.

Our main problem is thus the following: *given the agents' valuations, what is the (connected) division that maximizes the utilitarian welfare?* For the analysis of this problem, it is useful to consider its decision version, defined as follows.

CONNECTED UTILITARIAN OPTIMUM (CUO)	
Instance:	A set $\{v_i\}_{i=1}^n$ of non-atomic measures on $[0,1]$, and a bound B.
Problem:	Does there exist a connected division x having $u(x) \ge B$?

Complexity and Input Models. In order to analyze the complexity of our problem, we must first define how the input is represented. In most of the cake cutting literature, mechanisms are not explicitly given the agents' valuation functions; instead, the mechanism can *query* the agents on their valuations (see e.g. [14, 19, 23]). Typically, two types of queries are allowed. In the first, an agent i is given points $0 \le a \le b \le 1$ and is required to return the value $v_i(a, b)$. In the second type of query, an agent i is given a point $a \in [0, 1]$ and a value x and is required to return a point b such that $v_i(a, b) = x$; we denote this by $v_i^{-1}(a, x)$.[1] In this work, we refer to this input model as the "oracle access" model.

In contrast, some more recent works (e.g. [9, 11, 4]) consider a model in which the agents give complete descriptions of their valuations to the mechanism. It is usually assumed that the functions have some simple structure, so they can be represented succinctly. Specifically, for each agent i, let $\nu_i : [0, 1] \to [0, \infty)$ be a *value density function*, such that

$$v_i(X) = \int_X \nu_i(x)dx$$

for every measurable subset $X \subseteq [0, 1]$. Following [9], we say that a valuation function $v_i(\cdot)$ is *piecewise-constant* if its value density function $\nu_i(\cdot)$ is a step function, i.e. if $[0, 1]$ can be partitioned into a *finite* number of intervals such that ν_i is constant on each interval. If, in addition, there is some constant c_i such that $\nu_i(\cdot)$ can only attain the values 0 or c_i, we say that $v_i(\cdot)$ is *piecewise-uniform*.[2] Any piecewise-constant valuation function $v_i(\cdot)$ can be therefore represented by a finite set of subintervals of $[0, 1]$ together with the value ν_i attains in each interval. We refer to the model in which the valuations are given in full to the algorithm as the "explicit data" model, and assume that in this model, all valuations are piecewise-constant.

Our hardness results show that the decision problem defined above is computationally hard, even when the valuation functions are of the simplest type—piecewise-uniform—and are given explicitly to the mechanism. In contrast, our positive algorithmic results hold (with some small loss in precision) for the more general oracle model as well. The running time of our algorithms in this case additionally depends on a precision parameter ϵ.

The Discrete Variant. A convenient preprocessing step in our algorithms will be to reduce the problem into one that is purely combinatorial. More precisely, we consider a discrete analogue of the problem, where one is additionally given a set of points in $[0, 1]$ and is only allowed to make cuts at points from this set. An alternative interpretation is to consider, instead of a continuous cake, a *sequence of indivisible items*; a connected division in this setting gives each agent a *consecutive subsequence* of these items. As in the continuous case, the valuations of the agents are assumed to be additive. The decision version of this discrete variant of our problem (which we too show to be computationally hard) is defined as follows:

Discrete-CUO	
Instance:	A sequence $A = (a_1, \dots, a_m)$ of items, a set $\{v_i\}_{i=1}^n$ of valuation functions of the form $v_i : A \to \mathbb{R}^+$, and a bound B.
Problem:	Does there exist a connected division x having $u(x) \ge B$?

[1] Note that using only one type of query it is possible to give approximate answers (in polynomial time) to queries of the other type using binary search.

[2] In this case the constant c_i is uniquely determined by the total fraction of $[0, 1]$ in which $\nu_i(x) \neq 0$, since we assume that the valuation of the entire cake is 1.

3. ALGORITHMS

We show two algorithms for the problem of finding a connected cake division with high utilitarian welfare. The first algorithm (described in Section 3.2) runs in polynomial time, and is guaranteed to find a division achieving at least 1/8 of the highest possible welfare. The second algorithm (described in Section 3.3) finds a division with optimal welfare, and runs in time that is polynomial in the number of items, but exponential in the number of agents n.

Both our algorithms work on instances of the discretized variant of the problem. Therefore, we begin by showing (in Section 3.1) how a continuous cake can be "discretized" into a sequence of items in polynomial time. For the explicit data model, this can be done in such a way that the optimal welfare in the discretized instance is as high as that of the original instance. For the oracle access model, on the other hand, we give a procedure that additionally gets a precision parameter ϵ, and produces an instance in which the optimum welfare is at least $(1 - \epsilon)$ times that of the original instance.

3.1 Discretizing a Continuous Cake

As in Cohler et al. [11], given n piecewise-constant valuation functions, we denote by $\mathcal{I}$ the set of all boundary points in these valuations, together with the points $\{0, 1\}$. Let $\mathcal{J}$ be the set of intervals created by every two consecutive points in $\mathcal{I}$; note that all the value density functions of the agents are constant over each of the intervals in $\mathcal{J}$. It is easy to observe that for any division x of the cake, there is a division x' using only points from $\mathcal{I}$, and having $u(x') \ge u(x)$. We therefore set $A = \mathcal{J}$, keeping the order and the values for all the agents as in $\mathcal{J}$, to obtain a discretized instance in which the optimum welfare is just as high as in the original cake.

However, this method cannot be used in the oracle model (where the valuations may not even be piecewise-constant). Instead, we use Algorithm 1 below to produce a partition of the cake into a sequence of items. At each step, let a be the position of the last (rightmost) cut. Algorithm 1 asks each agent i for the leftmost point b_i such that the value $v_i(a, b_i) = \epsilon/(n - 1)$; it then makes a cut at the leftmost of these points, and repeats the process.

Algorithm 1: Discretization Procedure

```
Data: Oracle access to $v_i(\cdot)$ for each $i \in [n]$, and $\epsilon > 0$.
begin
    $a \longleftarrow 0$
    $C \longleftarrow \{0\}$
    while $\exists i : v_i(a, 1) > \frac{\epsilon}{n-1}$ do
        for $i \in [n]$ do
            $b_i \longleftarrow v_i^{-1}\left(a, \frac{\epsilon}{n-1}\right)$
        $b \longleftarrow \min_i b_i$
        $C \longleftarrow C \cup \{b\}$
        $a \longleftarrow b$
    $C \longleftarrow C \cup \{1\}$
    return $C$
```

Again, the sequence A of items is simply the set of intervals created from each two consecutive points in the set C returned by the procedure. The following lemma establishes that the set C can be computed efficiently, and that the loss in welfare due to this process is small.

Lemma 1. *Let $\{v_i(\cdot)\}_{i\in[n]}$ be a cake instance, and consider some precision parameter ϵ. Then:*

1. *Algorithm 1 runs for a total time of $O(n^3/\epsilon)$ on this instance.*
2. *Let x be a division of the original cake; then there exists a division y making cuts only at points in the set C returned by Algorithm 1, and having $u(y) \geq u(x) - \epsilon$.*

Proof. For item (1), note that in each iteration of the while-loop, the value $\sum_{i\in[n]} v_i(a,1)$ decreases by at least $\frac{\epsilon}{n-1}$; since at the beginning we have $\sum_{i\in[n]} v_i(a,1) = n$, there can be at most $O(n^2/\epsilon)$ iterations. The claim follows by noting that in each iteration we make $\leq 2n$ queries.

For item (2), let x be a division of the cake. We define the division y as follows. For every cutpoint x_j of x, set y_j to be the leftmost of the points in C that are to the right of x_j, and keep the permutation similar to that of x. Let k be the agent getting the leftmost piece in both divisions; clearly $u_k(y) \geq u_k(x)$. Since for any two points $c', c'' \in C$ and any agent i, $v_i(c', c'') \leq \frac{\epsilon}{n-1}$, we also have that for all other agents $i \neq k$ it holds that $u_i(y) \geq u_i(x) - \frac{\epsilon}{n-1}$; this immediately implies item (2). □

Note that this also allows a *multiplicative* approximation of the optimal welfare: to achieve an approximation factor of $(1+\epsilon)$ (with $\epsilon \leq 1$), run Algorithm 1 with parameter $\epsilon' = \epsilon/2$. Let u^* be the optimal welfare attainable in the original instance, then by Lemma 1, the obtained instance has a division with welfare at least

$$u^* - \frac{\epsilon}{2} \geq u^* - \frac{\epsilon}{1+\epsilon} = \frac{u^* + \epsilon(u^*-1)}{1+\epsilon} \geq \frac{u^*}{1+\epsilon},$$

where the last inequality holds since $u^* \geq 1$ (as we can always just give the whole cake to one agent).

3.2 A Polynomial-Time Approximation Algorithm

Given a sequence A of m discrete items, together with the value of each of these items for each of the agents, we show how to find a connected division achieving at least 1/8 of the optimal utilitarian welfare.

We use the notation (s,t) to refer to the consecutive sequence of items $\{s, s+1, \ldots, t-1, t\}$; hence, e.g. $v_i(s,t) = \sum_{j=s}^{t} v_i(j)$. In the course of the algorithm, we (tentatively) assign agents sequences of items; at each point, (c_i, d_i) is the (possibly empty) sequence assigned to agent i at the moment. We also use the notation $V_{-k}(s,t)$ to refer to the sum of values that the items of (s,t) have in the eyes of their (tentative) owners at this time.

Our algorithm works iteratively, where in the t-th iteration it finds a good division for the first t items. We begin with the trivial null allocation of 0 items. Assuming that we have a good allocation for the first $t-1$ items, for every $s \leq t$ (in ascending order), we search for an agent k whose value for the interval (s,t) exceeds the *cost* of giving this interval to her. This cost is comprised of two components. The first component is the value of a piece (c_k, d_k) that k herself may currently own, and has to give up in order to get the new piece (s,t). The second component is the sum $V_{-k}(s,t)$ of values that the other agents who currently own items in (s,t) derive from these items. We only give the segment (s,t) to agent k if her value for this interval is at least twice the cost of giving it to her. When no more such changes exist, we increase s. Iteration t ends when we are done with the interval (t,t), at which point we begin iteration $t+1$, considering the first $t+1$ items, starting with the interval $(1, t+1)$.

Algorithm 2: Discrete Welfare Approximation

```
Data: n valuation vectors of the form $v_i : [m] \to \mathbb{R}^+$.
begin
    $\forall i \in [n] : c_i \longleftarrow 0\ ,\ d_i \longleftarrow 0$
    for $t = 1, \ldots, m$ do
        for $s = 1, \ldots, t$ do
            while
            $\exists k \in [n] : v_k(s,t) \geq 2\big[v_k(c_k, d_k) + V_{-k}(s,t)\big]$
            do
                $c_k \longleftarrow s\ ,\ d_k \longleftarrow t$
                $(c_i, d_i) \longleftarrow (0,0)$   for all $i$ with $c_i \geq s$
                $d_i \longleftarrow c - 1$   for $i$ with $c_i < s \leq d_i$
    return $\{(c_i, d_i)\}_{i\in[n]}$
```

Clearly, for each interval (s,t) we need only consider each agent i once (in the while-loop), and thus the algorithm clearly halts in polynomial time. We now show that at the end of iteration t, no interval ending at t is ever "attractive" for any agent i.

Lemma 2. *For an iteration t, denote by $cost_k^t(s,t)$ the cost of giving (s,t) to agent k at the end of iteration t. Then for any $s \leq t$ and $k \in [n]$ it holds that*

$$cost_k^t(s,t) > \frac{v_k(s,t)}{2}.$$

Proof. Fix some t, $s \leq t$, and $k \in [n]$. Clearly, once the inner for-loop of the algorithm is done with the interval (s,t), the cost of giving (s,t) to k is more than half of $v_k(s,t)$. It thus suffices to show that this cost does not decrease at least until iteration t is over.

It is convenient to reformulate this cost: this new formulation will also have two components. The first component will be the value of items in (s,t) that are currently held by *all* agents (including k); we denote this by $V(s,t)$. The second component will be the value of k's items that are not taken into account in $V(s,t)$, i.e. the items *before* s that k owns; we denote this value by $v_k(<s)$.

Consider some change in the cost of (s,t) to k that has occurred within iteration t, after we were done with (s,t). This change must be the result of giving some interval (s',t) (with $s' > s$) to an agent $j \in [n]$. We distinguish between two cases:

- $\underline{j \neq k}$: In this case, the value $v_k(<s)$ clearly does not change. In addition, the value $V(s,t)$ strictly increases: $V(s',t)$ must increase, and $V(s,s'-1)$ either does not change, or that agent j herself loses some value there, which must be (more than) fully compensated by the increase in $V(s',t)$. Therefore, the cost of giving (s,t) to k must increase.

- $\underline{j = k}$: In this case, $v_k(<s)$ vanishes, but this decrease (plus any decrease in $V(s,s'-1)$, which can only result from k herself having to give up items there) must be

fully compensated by the increase in $V(s',t)$. Again, the cost of giving (s,t) to k strictly increases.

□

To analyze the approximation ratio of the algorithm, we use indicator variables x_i^j, for $i \in [n]$ and $j \in [m]$. We will have $x_i^j = 1$ if and only if agent i has owned the item j at the end of some iteration t.

Lemma 3. *Let $\{(c_i^A, d_i^A)\}_{i\in[n]}$ be the allocation returned at the end of Algorithm 2, then*

$$\sum_{i\in[n]} v_i(c_i^A, d_i^A) \le \sum_{i\in[n]}\sum_{j\in[m]} x_i^j \cdot v_i(j) \le 2\cdot \sum_{i\in[n]} v_i(c_i^A, d_i^A)\ .$$

Proof. The first inequality is trivial; we prove the second inequality by induction on the number of items m. This inequality clearly holds if $m = 1$, as we have only one iteration, and the values x_i^j describe exactly the allocation at the end of this iteration.

Suppose that this is true for some m, and consider the algorithm's operation on an input with $m+1$ items. Consider the penultimate iteration $t = m$ of the algorithm on this input, and fix the values $\{(c_i, d_i)\}_{i\in[n]}$ and $\{x_i^j\}_{i\in[n], j\in[m]}$ as they are at the end of this iteration. Since the operation of the algorithm until this point is identical to its operation on an appropriate input with only m items, we have that

$$\sum_{i\in[n]}\sum_{j\in[m]} x_i^j \cdot v_i(j) \le 2\cdot \sum_{i\in[n]} v_i(c_i, d_i)\ ;$$

we show that all the changes made in the iteration $t = m+1$ maintain this inequality.

Note that any change in these values must be the result of an allocation change. Suppose that such a change was made, allocating an interval $(s, m+1)$ to some agent k. The result of this change is that the left-hand side sum $\sum_{i\in[n]}\sum_{j\in[m]} x_i^j \cdot v_i(j)$ increases by at most $v_k(s, m+1)$. In addition, the right-hand side expression $\sum_{i\in[n]} v_i(c_i, d_i)$ gains $v_k(s, m+1)$, but loses the cost of giving $(s, m+1)$ to k. However, from the condition in the algorithm, we know that this cost is at most half of $v_k(s, m+1)$. Therefore, the expression $\sum_{i\in[n]} v_i(c_i, d_i)$ gains at least $v_k(s, m+1)/2$, and the inequality is maintained. □

Theorem 1. *The division returned by Algorithm 2 approximates the optimal welfare to a factor of* 8.

Proof. Fix a discrete cake instance. Let $\left\{(c_i^A, d_i^A)\right\}_{i\in[n]}$ be the final output of Algorithm 2 on this instance, and let $\left\{(c_i^*, d_i^*)\right\}_{i\in[n]}$ be the optimal division for this instance. Denote by $OPT = \sum_{i\in[n]} v_i(c_i^*, d_i^*)$ the utilitarian welfare achieved by the optimal division.

For each agent k, consider the iteration $t = d_k^*$ in which the rightmost item given to k in the optimal division was first considered. From Lemma 2 we have that

$$v_k(c_k^*, d_k^*) < 2\cdot cost_k^{d_k^*}(c_k^*, d_k^*)\ ;$$

recall that $cost_k^{d_k^*}(c_k^*, d_k^*) = v_k(c_k^{d_k^*}, d_k^{d_k^*}) + V_{-k}^{d_k^*}(c_k^*, d_k^*)$ is the cost of giving (c_k^*, d_k^*) at the end of the iteration $t = d_k^*$.

We can first observe the first component of these costs has

$$\sum_{k\in[n]} v_k(c_k^{d_k^*}, d_k^{d_k^*}) \le \sum_{k\in[n]}\sum_{j\in[m]} x_k^j \cdot v_k(j)\ ,$$

as for each agent k, the expression $\sum_{j\in[m]} x_k^j \cdot v_k(j)$ sums the values of all the items k has owned during the algorithm, including the items she had at the end of the iteration $t = d_k^*$.

For similar reasons we have that

$$\begin{aligned}\sum_{k\in[n]} V_{-k}^{d_k^*}(c_k^*, d_k^*) &\le \sum_{k\in[n]}\sum_{j=c_k^*}^{d_k^*}\sum_{i\ne k} x_i^j \cdot v_i(j)\\ &\le \sum_{k\in[n]}\sum_{j\in[m]} x_k^j \cdot v_k(j)\end{aligned}$$

where the second inequality holds since for any two agents $k \ne k'$ the sequences (c_k^*, d_k^*) and $(c_{k'}^*, d_{k'}^*)$ must be disjoint, as the optimal division cannot give the same item to more than one agent.

Combining all this, we have:

$$\begin{aligned}OPT = \sum_{k\in[n]} v_k(c_k^*, d_k^*) \ &<\ 2\cdot \sum_{k\in[n]} cost_k^{d_k^*}(c_k^*, d_k^*)\\ &= 2\cdot\left(\sum_{k\in[n]} v_k(c_k^{d_k^*}, d_k^{d_k^*}) + \sum_{k\in[n]} V_{-k}^{d_k^*}(c_k^*, d_k^*)\right)\\ &\le 2\cdot\left(\sum_{k\in[n]}\sum_{j\in[m]} x_k^j\cdot v_k(j) + \sum_{k\in[n]}\sum_{j\in[m]} x_k^j \cdot v_k(j)\right)\\ &= 4\cdot\sum_{k\in[n]}\sum_{j\in[m]} x_k^j\cdot v_k(j)\ \le\ 8\cdot\sum_{i\in[n]} v_i(c_i^A, d_i^A)\end{aligned}$$

where we plug in Lemma 3 to obtain the last inequality. □

3.3 A Fixed-Parameter Tractable Algorithm

Sometimes, the social efficiency of the division may be highly important, and even a constant-factor approximation may be unsatisfactory. We show an algorithm that finds the optimal discrete division, in time exponential in the number n of agents, but polynomial in the number m of items. Using the terminology of the theory of Parametrized Complexity [12], we say that such an algorithm is *fixed-parameter tractable* with respect to the number n of agents. This may be applicable when the number of agents is relatively small, even if the number of items is significantly larger.

Theorem 2. *For discrete cake instances, it is possible to find a division achieving the optimal utilitarian welfare in time $2^n \cdot poly(n, m)$.*

Proof. We use a dynamic programming approach. Our algorithm creates a table U having a row of length m for each pair (S, k), where $S \subseteq [n]$ is a non-empty subset of agents, and $k \in S$. Denote by $u_j^{(S,k)}$ the value in the j-th column of row (S, k) in U. The value $u_j^{(S,k)}$ will be the largest total utility obtainable by dividing the first j items between the agents of S, where the last item j is given to agent k. Therefore, the maximum welfare is $\max_{S,k}\left\{u_m^{(S,k)}\right\}$.

It therefore remains to show that the values in U can be computed in polynomial time. We begin with the first column, in which we consider only the first item. For every $k \in [n]$, we have $u_1^{(\{k\},k)} = v_k(a_1)$; for any $S \ne \{k\}$ the pair (S, k) is invalid and we set $u_1^{(S,k)} = -\infty$. Now, suppose that we have filled in the values in the first j columns, and consider $u_{j+1}^{(S,k)}$. If $k \notin S$, again (S, k) is invalid, and we set $u_{j+1}^{(S,k)} = -\infty$. If $k \in S$, then the maximum value is obtained by either extending k's piece in the best division of the items 1 through j in which k gets item j, or by taking the best division of items 1 through j between the agents $S \setminus \{k\}$, and giving item $j+1$ to k. Formally,

$$u_{j+1}^{(S,k)} = \max \left\{ u_j^{(S,k)} + v_k(a_{j+1}) \; , \; \max_{i \in S \setminus \{k\}} \left\{ u_j^{(S \setminus \{k\}, i)} \right\} + v_k(a_{j+1}) \right\} .$$

We get that the value of each table entry can be computed in time $O(n)$, and therefore filling the entire table U and finding the utilitarian optimum can be done in time $2^n \cdot poly(n,m)$, as stated. Once we have the optimal welfare, the actual division can be easily computed by backtracking. $\square$

4. HARDNESS

THEOREM 3. *Both* CUO *and* DISCRETE-CUO *are NP-complete in the strong sense.*
This holds even if the valuation functions of the agents are piecewise-uniform, and are given explicitly to the algorithm.

PROOF. We show a polynomial-time reduction from 3DM to CUO. In the problem of 3DM one is given three sets X, Y, Z of the same cardinality n, and a set $E \subseteq X \times Y \times Z$, and needs to decide whether there is some $E' \subseteq E$ that covers every element of $X \cup Y \cup Z$ exactly once. Given X, Y, Z and E, we construct a set of piecewise-uniform valuations and a bound B as an input for CUO.

For convenience, we represent the cake by the interval $\big[0, 4|E|\big]$ rather than $[0,1]$. We will think of the cake as being sectioned into $|E|$ "sections" of length 4, where the right half of each section is used for separation from the next section.[3] The set of agents we create has agents of three types: "triplet agents", "ground sets agents" and "separation agents". In what follows we describe the valuation functions of all the agents, by their type; for the bound, we set $B = |E| + 2 - \frac{n}{|E|}$.

- **Triplet Agents:** For each $z \in Z$ we create an agent. Let $e_i = (x_j, y_k, z_\ell) \in E$ be a triplet. The agent created for z_ℓ has value $\frac{1}{|E|}$ for the left half of the i-th section $\big(4(i-1), 4(i-1)+2\big)$, and value 0 for the remainder of this section. All other agents $z_{\ell'}$ with $\ell' \neq \ell$ have value $\frac{1}{|E|}$ for the right half of the i-th section $\big(4(i-1)+2, 4i\big)$ and value 0 for the remainder of this section.

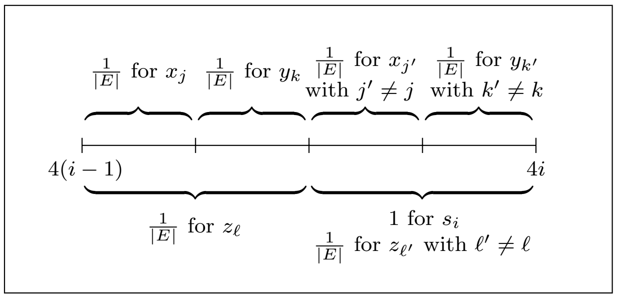
- **Ground Sets Agents:** For each $x \in X$, let m_x be the number of triplets in E in which x appears; we create $m_x - 1$ identical agents for x. Let $e_i = (x_j, y_k, z_\ell) \in E$ be a triplet. The agents created for x_j have value $\frac{1}{|E|}$ for first quarter $\big(4(i-1), 4(i-1)+1\big)$ of the i-th section, and value 0 for the remainder of this section. The agents created for $x_{j'}$ with $j' \neq j$ all have value $\frac{1}{|E|}$ for the third quarter $\big(4(i-1)+2, 4(i-1)+3\big)$ of the i-th section, and value 0 for the remainder of this section.

  Similarly, for each $y \in Y$, let m_y be the number of triplets in E in which x appears; we create $m_y - 1$ identical agents for y. Let $e_i = (x_j, y_k, z_\ell) \in E$ be a triplet. The agents created for y_k have value $\frac{1}{|E|}$ for the second quarter $\big(4(i-1)+1, 4(i-1)+2\big)$ of the i-th section, and value 0 for the remainder of this section. The agents created for $y_{k'}$ with $k' \neq k$ all have value $\frac{1}{|E|}$ for the fourth interval $\big(4(i-1)+3, 4i\big)$ of the i-th section, and value 0 for the remainder of this section.

- **Separation Agents:** We finally create $|E|$ separation agents $s_1, \ldots, s_{|E|}$. Each such agent s_i has value 1 for the right half of the i-th section $\big(4(i-1)+2, 4i\big)$, and value 0 for the remainder of the cake.

[3]Indeed, the last section needs not have such a "separation half"; however, we leave it there in order to treat it identically to all the other sections.

Figure 1 illustrates the structure of the preferences in one segment. In this example, we consider some triplet $e_i = (x_j, y_k, z_\ell) \in E$, and show the preferences of the agents for the section of the cake created for e_i.



$\frac{1}{|E|}$ for x_j
$\frac{1}{|E|}$ for y_k
$\frac{1}{|E|}$ for $x_{j'}$ with $j' \neq j$
$\frac{1}{|E|}$ for $y_{k'}$ with $k' \neq k$
$4(i-1)$
$4i$
$\frac{1}{|E|}$ for z_ℓ
1 for s_i
$\frac{1}{|E|}$ for $z_{\ell'}$ with $\ell' \neq \ell$

Figure 1: The valuations of the agents for the section created for $e_i = (x_j, y_k, z_\ell) \in E$. There are $m_{x_j} - 1$ identical agents for x_j and $m_{y_k} - 1$ identical agents for y_k.

Note that this construction indeed creates agents with piecewise-uniform valuations, and can be computed in polynomial time. Furthermore, all the numbers created in this instance can clearly be represented using a number of bits logarithmic in the input size.

Suppose first that $(X, Y, Z, E) \in$3DM, and let $E' \subseteq E$ be a cover of $X \cup Y \cup Z$. We can give the entire right half of each section i to the agent s_i. This contributes a sum of $|E|$ to the utilitarian welfare. Now, for each $z \in Z$ there is a unique $e_i \in E'$ in which z appears. Give the (unique) triplet agent z the left half of the i-th section, i.e. the interval $\big(4(i-1), 4(i-1)+2\big)$. This contributes $\frac{n}{|E|}$ more to the utilitarian welfare. Finally, consider the ground sets agents. For every $x \in X$, there is a unique $e_i \in E'$ in which x appears; hence, there are $m_x - 1$ triplets $e_{i'} \in E \setminus E'$ in which x appears. Since no piece of the left half of the corresponding sections was given so far, we can give the first quarter $\big(4(i'-1), 4(i'-1)+1\big)$ of one of these sections to each of x's agents. Similarly, we have $m_y - 1$ sections corresponding to triplets $e_{i'} \in E \setminus E'$ in which y appears; in all of these sections, the second quarter $\big(4(i'-1)+1, 4(i'-1)+2\big)$ is still available (and worth $\frac{1}{|E|}$ to all of y's agents). Thus, each ground sets agent can receive a piece of value $\frac{1}{|E|}$, and these agents contribute the missing amount $\frac{2(|E|-n)}{|E|}$ to the utilitarian welfare, which is now $|E| + \frac{2|E|-n}{|E|} = B$, as required.

Conversely, suppose that $(X, Y, Z, E) \notin$3DM; we wish to prove that in this case, no division has utilitarian welfare

$\geq |E| + 2 - \frac{n}{|E|}$. Consider a utilitarian welfare maximizing division. Such a division must clearly give each agent s_i her entire desired piece, as the cost of giving (some of) it to another agent will always exceed the gain. Thus, in such a division, the entire right half of each section is given to a separation agent, and no other agent can receive a piece intersecting two or more sections. The utilitarian welfare so far is $|E|$, and to get it to $B = |E| + \frac{2|E|-n}{|E|}$, we must give each of the remaining agents a piece of value exactly $\frac{1}{|E|}$, as none of them can now get a piece with larger value. Let $E' \subset E$ be the set of triplets/sections from which the triplet agents receive their pieces of value $\frac{1}{|E|}$; it must be that the entire left half of each of these sections is now fully consumed. Now, note that since $(X, Y, Z, E) \notin$3DM, it has to be that in every $E' \subseteq E$ of cardinality n that covers all of Z, there is either some $x \in X$ or some $y \in Y$ that appears twice; w.l.o.g. assume that it is $x \in X$. This implies that the pieces given to the triplet agents contain at least two interval desired by the agents created for this x, out of the total m_x such intervals found in the left halves of the sections. Since we have $m_x - 1$ agents for this x, it follows that we cannot give each of them a piece of value $\frac{1}{|E|}$, and therefore we cannot achieve a utilitarian welfare of $B = |E| + \frac{2|E|-n}{|E|}$.

The proof for Discrete-CUO is analogous, and can easily be obtained by a straightforward partitioning of the cake created in the reduction into discrete indivisible chunks. □

The strong NP-hardness of CUO and Discrete-CUO implies the following corollary:

Corollary 4. *There is no FPTAS for neither* CUO *nor* Discrete-CUO.

5. MAXIMIZING WELFARE WITH NON-CONNECTED PIECES

We now turn to analyze the problem of welfare maximization when each agent may get a *collection* of intervals. We begin by noting that when the agents' valuations are given explicitly, the problem of finding the division that maximizes the utilitarian welfare is easily solvable.

Proposition 5. *In the explicit data model, it is possible to find a division maximizing the utilitarian welfare in polynomial time.*

Proof. Again, we can use the discretization method described in Section 3.1 to obtain a discretized instance A in which the optimal welfare is just as high as that of the original instance. In the discretized instance A, the division that gives each item to the agent who values it the most clearly maximizes the welfare. □

In contrast to this positive result, we show that maximizing welfare is *impossible* if instead of receiving the explicit valuation functions, we only get oracle access to the valuations. Note that by simply giving the entire cake to one agent, we can achieve an approximation of factor n to the utilitarian optimum; in what follows, we show that this is actually the best that can be done, as no deterministic algorithm can guarantee a better approximation. We stress that this result holds for any finite algorithm, and thus *does not depend* on any complexity assumptions.

Theorem 6. *For any $\epsilon > 0$, no deterministic algorithm working in the oracle input model can approximate the utilitarian welfare to a factor of $n - \epsilon$, when non-connected pieces are allowed.*

Proof. Let B be a deterministic cake division algorithm working in the oracle input model, and fix some $n \in \mathbb{N}$ and $\epsilon > 0$. Consider the operation of the algorithm on the set of preferences in which all agents value the entire cake uniformly. In this case, the utilitarian welfare obtained cannot exceed 1. We will now show that for any $\epsilon' > 0$ we can construct a different set of preferences on which B must output the same division (with the same welfare), but for which there exists a division achieving utilitarian welfare of $(1 - \epsilon')n$. The theorem will follow by choosing $\epsilon' = \epsilon/n$.

Let $0 = p_0 < p_1 < \ldots < p_{k-1} < p_k = 1$ be the set of (distinct) points that appear in the operation of B on the input above. In other words, $\{p_i\}_{i=0}^{k}$ is the set of all points a, b for which the algorithm makes a query $v_i(a, b)$ or receives an answer $b = v_i^{-1}(a, x)$, and all the points c in which the algorithm makes cuts in its output division. We create a new instance in which the total value each agent assigns to every interval (p_j, p_{j+1}) remains the same, but the division of this value *within* the interval is "rearranged". For each agent, the value of each such interval in the original instance (as well as in the new instance) is $\ell_j = p_{j+1} - p_j$. We divide this interval into $n + 1$ "slivers": the i-th sliver $(1 \leq i \leq n)$ is worth $\ell_j - \frac{\epsilon'}{k}$ to agent i, and zero to everyone else. The $(n+1)$-th sliver of the interval is worth $\frac{\epsilon'}{k}$ for all the agents. Formally, for each agent i and each $0 \leq j \leq k$, set

$$v'_i\left(p_j + \frac{i-1}{n+1} \cdot \ell_j \ , \ p_j + \frac{i}{n+1} \cdot \ell_j\right) = v_i(p_j, p_{j+1}) - \frac{\epsilon'}{k}$$

$$v'_i\left(p_j + \frac{n}{n+1} \cdot \ell_j \ , \ p_{j+1}\right) = \frac{\epsilon'}{k}$$

and have agent i give value of 0 to any other piece.

We now have that for every $i \in [n]$ and every $0 \leq j \leq j' \leq k$ it holds that $v'_i(p_j, p_{j'}) = v_i(p_j, p_{j'})$; furthermore, for every $i \in [n]$, every $0 \leq j \leq k$ and every $x \in \mathbb{R}$ such that B makes a query $v_i^{-1}(p_j, x)$ when executed on the preferences $\vec{v}$ we have $v'^{-1}_i(p_j, x) = v_i^{-1}(p_j, x)$. Consider the operation of the algorithm B on the set of valuations $\vec{v'}$. The first query of B on this instance is clearly identical to its first query on the instance $\vec{v}$, since before any queries are asked B cannot distinguish between the two instances. However, as we have observed, the answer to B's first query with the new instance $\vec{v'}$ is identical to the answer with the original instance $\vec{v}$. Since B is deterministic, this implies that the next query asked by B on $\vec{v'}$ is identical to that asked on $\vec{v}$. Continuing in this manner, we get that the entire operation of B is identical on both instances.

In particular, this implies that the cut points in the division produced for $\vec{v'}$ are all from the set $\{p_j\}_{j=0}^{k}$; in such a division the utilitarian welfare is always exactly 1. However, any division giving each agent all of the "slivers" that are only desired by her yields utilitarian welfare $> (1 - \epsilon')n$, and the theorem follows. □

6. OPEN PROBLEMS

In this work we have taken the first steps in studying the problem of maximizing the utilitarian welfare in cake cutting with connected pieces. However, some interesting related questions remain open. For example, we conjecture that the approximation ratio for maximizing utilitarian welfare can be improved; it may also be interesting to see if stronger inapproximability results can be shown. Other interesting extensions include:

- **Finding a Utilitarian-Optimal Proportional Division:** Bei et al. [4] have shown that finding the division maximizing welfare among all *proportional* divisions is hard to approximate within a factor of $O(\sqrt{n})$.[4] However, [4] do not provide an approximation algorithm for this problem. Here, we show that without the proportionality restriction, the optimal welfare can be approximated within a constant factor. Can our algorithm be adapted to find socially-efficient *proportional* divisions that are not too far from the optimum?

- **Maximizing Egalitarian Welfare:** The egalitarian welfare measure $eg(x) = \min_{i \in [n]} u_i(x)$ considers the *minimum* utility of any agent in the division instead of summing these utilities. Maximizing the egalitarian welfare has been considered in allocation of discrete goods (e.g. [3, 1, 8]), and seems a difficult problem. Almost all of our results have analogues for egalitarian welfare as well, with the only exception being Theorem 1: the best approximation algorithms we have for egalitarian welfare achieve linear-factor approximations. Closing this significant gap is therefore an interesting (and possibly challenging) open question.

- **Strategic Behavior:** One implicit assumption in our work was that we have access to the (true) valuations of the agents. However, in reality the agents may have incentive to lie about their valuations. Maya and Nisan [17] study this problem for two agents and piecewise-uniform valuations. The question of what welfare guarantee can be achieved by a truthful mechanism in a more general setting is therefore still open.

- **2-Dimensional Cake:** The cake cutting literature has generally assumed a one-dimensional cake; indeed, for the purpose of maintaining fairness, which was its main focus, a 2-dimensional cake can be simply "projected" onto one dimension, and divided fairly according to the projection. However, this may result in a significant loss of welfare. Therefore, maximizing welfare in allocation of 2-dimensional cakes may require completely different tools and techniques.

7. ACKNOWLEDGEMENTS

The first two authors were supported in part by ISF grant 1401/09. The authors would also like to thank Zvi Gotthilf for many useful discussions.

8. REFERENCES

[1] A. Asadpour and A. Saberi. An approximation algorithm for max-min fair allocation of indivisible goods. In *STOC*, pages 114–121, 2007.

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

[3] N. Bansal and M. Sviridenko. The santa claus problem. In *STOC*, pages 31–40, 2006.

[4] X. Bei, N. Chen, X. Hua, B. Tao, and E. Yang. Optimal proportional cake cutting with connected pieces. In *AAAI*, 2012.

[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, New York, NY, USA, 1996.

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

[8] D. Chakrabarty, J. Chuzhoy, and S. Khanna. On allocating goods to maximize fairness. In *FOCS*, pages 107–116, 2009.

[9] Y. Chen, J. Lai, D. C. Parkes, and A. D. Procaccia. Truth, justice, and cake cutting. In *AAAI*, 2010.

[10] Y. Chevaleyre, P. E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J. A. Padget, S. Phelps, J. A. Rodríguez-Aguilar, and P. Sousa. Issues in multiagent resource allocation. *Informatica (Slovenia)*, 30(1):3–31, 2006.

[11] Y. J. Cohler, J. K. Lai, D. C. Parkes, and A. D. Procaccia. Optimal envy-free cake cutting. In *AAAI*, 2011.

[12] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer, 1999.

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

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

[15] M. Guo and V. Conitzer. Strategy-proof allocation of multiple items between two agents without payments or priors. In *AAMAS*, pages 881–888, 2010.

[16] L. Han, C. Su, L. Tang, and H. Zhang. On strategy-proof allocation without payments or priors. In *WINE*, pages 182–193, 2011.

[17] A. Maya and N. Nisan. Incentive compatible two player cake cutting. In *WINE*, pages 170–183, 2012.

[18] A. D. Procaccia. Thou shalt covet thy neighbor's cake. In *IJCAI*, pages 239–244, 2009.

[19] J. Robertson and W. Webb. *Cake-cutting algorithms: Be fair if you can.* A K Peters, Ltd., Natick, MA, USA, 1998.

[20] J. Sgall and G. J. Woeginger. A lower bound for cake cutting. In *ESA*, pages 459–469, 2003.

[21] H. Steinhaus. Sur la division pragmatique. *Econometrica*, 17(Supplement: Report of the Washington Meeting):315–319, Jul 1949.

[22] W. Stromquist. How to cut a cake fairly. *The American Mathematical Monthly*, 87(8):640–644, 1980.

[23] W. Stromquist. Envy-free cake divisions cannot be found by finite protocols. *Electronic Journal of Combinatorics*, 15(1), Jan 2008.

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

[4]A division is called *proportional* if each agent receives at least $\frac{1}{n}$ of the value of the entire cake, by her own valuation.