






Maximize Egalitarian Welfare for Cake Cutting

Xiaolin Bu  and Jiaxin Song  

Shanghai Jiao Tong University, Shanghai, China
{lin_bu, sjtu_xiaosong}@sjtu.edu.cn

Abstract. A major problem in *cake-cutting* is how to both *fairly* and *efficiently* allocate the cake. *Egalitarian welfare*, which prioritizes agents with the worst utilities, is a compelling notion that provides guarantees for both fairness and efficiency. In this paper, we investigate the complexity of finding a maximized egalitarian welfare (MEW) allocation when all the value density functions are *piecewise-constant*. We design an FPT (fixed-parameter tractable) algorithm (with respect to the number of the agents) for computing an MEW allocation when all the bundles are requested to be contiguous. Furthermore, we show that this problem is NP-hard to approximate to within any constant factor.

Keywords: Cake-cutting · Egalitarian Welfare · Fair Division

1 Introduction

The *cake-cutting* problem is one of the most fundamental research problems explored in social science, economics, computer science, and other related fields (Steinhaus 1948, 1949; Aumann and Dombb 2015; Brams et al. 2012; Cohler et al. 2011; Bei et al. 2012, 2017; Tao 2022). Its primary objective is to *fairly* distribute a *heterogeneous* and *divisible* resource (also called a *cake*) among a group of agents, each with their own preferences for different portions of the resource. Although the setting seems simple, the problem becomes interesting and nontrivial when specific properties (*e.g.*, fairness, efficiency, truthfulness, etc.) must be met by the output allocation.

How to design an allocation that obtains a high *efficiency* when *fairness* is guaranteed is an important research question of cake-cutting and has garnered significant interest due to its wide range of applications (*e.g.*, network routing, public traffic, etc.). Many previous studies have attempted to investigate the correlation between fairness and efficiency. For example, there are some topics like the price of fairness (Bertsimas et al. 2011; Roughgarden 2010; Caragiannis et al. 2012), the complexity of computing the most efficient allocation subject to fairness constraints (Cohler et al. 2011; Bei et al. 2012; Aumann et al. 2013). The term “efficiency” mentioned above mostly refers to *social welfare*, which is defined as the sum of the agents’ utilities, and the above-mentioned papers

formulate the problem as a constrained optimization problem where a fairness notion is served as a constraint and the social welfare is the object that we would like to optimize.

Other than considering a combination of two separate notions (one for efficiency and one for fairness), there are other allocation criteria/measurements that embed both fairness and efficiency. *Egalitarian welfare*, defined as the minimum utility among all agents, is a measurement of both efficiency and fairness. It is a natural measurement of allocation efficiency. Moreover, from a fairness perspective, acquiring excess value for an arbitrary individual may not contribute to the improvement of egalitarian welfare, as it primarily focuses on the agent with the lowest value, so we tend to allocate the remaining resources to this specific agent to make the allocation fairer. In addition, an allocation that maximizes the egalitarian welfare also satisfies some common fairness criteria such as *proportionality*, where an allocation is proportional if each agent receives a share that is worth at least $1/n$ fraction of her total value of the entire cake (where n is the number of the agents).¹ In our work, we study the complexity of computing an allocation with *maximum egalitarian welfare* (MEW) in the setting where all the value density functions are piecewise-constant and all the agents are hungry.

Comparison with Other Notions. Other than egalitarian welfare, other notable notions concerning both efficiency and fairness are *Nash welfare* and *leximin*.

Nash welfare is defined as the product of agents' utilities. It is known that an allocation maximizing the Nash welfare satisfies the fairness notion *envy-freeness* (Kelly 1997), which is stronger than the proportionality mentioned earlier. Compared with Nash welfare, egalitarian welfare puts more focus on the least happy agent.

An allocation is *leximin* if it maximizes the utility to the least happy agent, and, subject to this, it maximizes the utility to the second least happy agent, and so on. Clearly, a leximin allocation always maximizes egalitarian welfare, and the notion of leximin places further requirements on the other agents. However, in many settings, maximizing egalitarian welfare is already NP-hard. We can then study the design of *approximation algorithms* for maximizing egalitarian welfare. Unlike egalitarian welfare which has a single objective to maximize, it is unclear how to define the approximation version of leximin.

1.1 Related Work

Fairness Notions. Apart from egalitarian welfare, a well-studied fairness notion is *proportionality*, which states that each agent receives at least an average value from her perspective. When there are two agents, proportionality can be easily achieved through the “*I cut, you choose*” algorithm, where the first agent cuts

¹ It is well known that a proportional allocation always exists even if we require each agent to receive a connected piece (see, e.g., (Dubins and Spanier 1961)). Therefore, there exists an allocation with egalitarian welfare of at least $1/n$, and the allocation with optimal egalitarian welfare is proportional.

the cake into two pieces that she thinks to be equal, then the second agent chooses one piece that she thinks of higher value. For any number of agents, two well-known algorithms to guarantee proportional are Dubins-Spanier Dubins and Spanier (1961) devised in 1961 and Even-Paz Even and Paz (1984) in 1984.

Another fairness notion is *envy-freeness*, which means each agent prefers her bundle over any other agent. It is a stronger notion than proportionality, and an envy-free allocation is also proportional. For two agents, the “I cut, you choose” still works. However, Dubins-Spanier and Even-Paz algorithms for general number of agents are not envy-free. For three agents, an elegant algorithm to compute an envy-free allocation is Selfridge-Conway (Brams and Taylor 1995). Furthermore, it has been confirmed that for any number of agents, an envy-free allocation always exists Aziz and Mackenzie (2017), even if only $n - 1$ cuts are allowed (Brams and Taylor 1995).

The third wide-studied notion is equitability. Incomparable to envy-freeness or proportionality, each agent is assigned a piece of the same value. An equitable allocation can be achieved by Austin moving-knife procedure for two agents, and if only one cut is allowed, it can be calculated with full knowledge of the partners’ valuations Jones (2002); Brams et al. (2006). Further, if more than two cuts are allowed, we can achieve an equitable allocation that is also envy-free Barbanel and Brams (2011). Austin moving-knife and full revelation procedure can also be extended to any number of agents, while the latter one still works under contiguous constraints. However, equitability and envy-freeness are not compatible under such constraints for three or more agents Brams et al. (2006).

Price of Fairness. The price of fairness (POF) is defined as the worst-case ratio between the optimal welfare obtained and the maximum welfare obtained by a fair allocation. In Caragiannis et al. (2012)’s work, they have shown, the price of envy-freeness and proportionality for two agents is $8 - 4\sqrt{3}$. For n agents, they show the price of proportionality is $\Theta(\sqrt{n})$ and the price of equitability is $\Theta(n)$. The price of proportionality directly implies that the lower bound of the price of envy-freeness is $\Omega(\sqrt{n})$ as envy-freeness implies proportionality. Bertsimas et al. (2011) further shows the upper bound of the price of envy-freeness is $O(\sqrt{n})$, concluding the price of envy-freeness is also $\Theta(\sqrt{n})$.

MEW in Fair Division. In cake-cutting, the problem of computing MEW (Maximum Egalitarian Welfare) has been proven to have a 2-inapproximation ratio by Aumann et al. (2013).² With the number of agents being the parameter, they also developed an FPT PTAS (Polynomial Time Approximation Scheme) for egalitarian welfare objectives. For indivisible goods, the best-known polynomial-time algorithm can achieve an approximation factor of $O(\sqrt{n} \log^3 n)$ (Asadpour and Saberi 2010), and this problem has been proven to have a 2-inapproximation ratio (Aumann et al. 2013).

² The 2-inapproximation result is in the arXiv version of the paper Aumann et al. (2013).

1.2 Our Results

In our paper, we mainly study the complexity of computing MEW allocation in two settings. In the first setting, each bundle is not required to be *contiguous* (i.e. each agent could receive a lot of scattered intervals). This case is relatively straightforward, and we demonstrate that the MEW allocation can be directly obtained through linear programming. In the second setting, a stricter constraint is imposed where each bundle must be contiguous. Despite the moving-knife procedure being able to output an allocation with $\frac{1}{n}$ egalitarian welfare, it fails to extend for computing an MEW allocation. We design an FPT algorithm with respect to the number of agents for computing an MEW allocation, which improves the previous result of FPT PTAS by Aumann et al. (2013). Finally, we prove that this problem is NP-hard to approximate to within any constant factor by a reduction from 3-SAT, which significantly improves the 2-inapproximability by Aumann et al. (2013).

2 Preliminaries

Let $[n] = \{1, \dots, n\}$. In the cake-cutting problem, the cake is modeled as an interval $[0, 1]$ and is required to allocate to a set of $N = [n]$ agents. Each agent i has a non-negative value density function f_i over the cake $[0, 1]$. In particular, agent i 's utility v_i to a subset X of the cake is defined as $v_i(X) = \int_X f_i(x)dx$. Throughout this paper, we assume her value density function is piecewise-constant, that is, f_i is constant on each contiguous interval separated by a finite number of breakpoints. We also assume the agents are *hungry* and *normalized*, that is, f_i is strictly positive at any point of $[0, 1]$ and for any $i \in [n]$, $v_i([0, 1]) = 1$.

An allocation $\mathcal{A} = (A_1, \dots, A_n)$ is a partition of the cake, where $A_i \cap A_j = \emptyset$ for any i, j and $\cup_{i=1}^n A_i = [0, 1]$. Among those bundles, bundle A_i is allocated to agent i . We say an allocation $\mathcal{A} = (A_1, \dots, A_n)$ is *equitable* if all the agents get the exact same utility (by their own valuations). Formally, for each $i, j \in [n]$, $v_i(A_i) = v_j(A_j)$. Bundle A_i is called *contiguous* if it contains a single interval. An allocation with contiguous pieces requires that each bundle in the allocation is contiguous, so such an allocation contains only $n - 1$ cuts.

The egalitarian welfare (\mathcal{EW}) of an allocation \mathcal{A} is defined as

$$\mathcal{EW}(\mathcal{A}) \triangleq \min_{i \in [n]} v_i(A_i).$$

Clearly, if an allocation \mathcal{A} is *proportional* (i.e. $v_i(A_i) \geq \frac{1}{n}v_i([0, 1])$), then $\mathcal{EW}(\mathcal{A})$ is at least $\frac{1}{n}$. A common approach for computing a proportional allocation is called *moving-knife procedure*, which is defined as follows,

Definition 1 (Moving-knife Procedure). *We set a threshold $\theta = \frac{1}{n}$, and let a knife moves from the left of the cake to the right. An agent calls ‘stop’ when her value to the interval left to the knife reaches θ , and we cut the cake and allocate it to this agent.*

In our paper, we mainly study the problem of computing an allocation that maximizes egalitarian welfare (MEW). We respectively discuss two scenarios when the bundles could be non-contiguous and all of them must be contiguous.

Our technique for solving the above problem includes linear programming, and it is known that an optimal vertex solution can be found in polynomial time.

Lemma 1 (Güler et al. (1993)). *For a linear program $\max\{\mathbf{c}^\top \mathbf{x} : \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$, an optimal solution (if exists) can be found in polynomial time.*

To warm up, let us consider a simple scenario where each bundle may not be contiguous. In Theorem 1, we demonstrate that the MEW allocation can always be efficiently obtained via linear programming in polynomial time.

Theorem 1. *If all the bundles are not required to be contiguous, an MEW allocation can be found in polynomial time.*

Proof. First, assume all the different breakpoints of f_1, \dots, f_n are $0 = p_0 < p_1 < \dots < p_m = 1$, where $m \in \mathbb{Z}^+$. Let variable $x_{i,k}$ represent the fractional ratio that agent i receives from the interval $[p_{k-1}, p_k]$. Consider the following linear program:

$$\begin{aligned} & \max \quad \mathcal{EW}(\mathcal{A}) \\ & \text{subject to} \quad x_{1,k} + \dots + x_{n,k} = 1, & k \in [m] & \quad (1) \\ & \quad \sum_{k=1}^m x_{i,k} v_i([p_{k-1}, p_k]) \geq \mathcal{EW}(\mathcal{A}), & i \in [n] & \quad (2) \\ & \quad 0 \leq x_{i,k} \leq 1, & i \in [n], k \in [m] & \quad (3) \end{aligned}$$

Within the above linear program, the constraints (1) ensures each interval is exactly allocated and the constraints (2) ensure each agent's utility is no less than the objective function $\mathcal{EW}(\mathcal{A})$. Clearly, $\{x_{i,k}\}_{i \in [n], k \in [m]}$ could describe an allocation and it is MEW. Since the linear program can be solved in polynomial time, the theorem concludes. \square

3 Maximize Egalitarian Welfare with Contiguous Pieces

In this section, we will discuss our results when only $n - 1$ cuts are allowed (i.e. each agent's bundle should be contiguous). As mentioned before, the moving-knife procedure could find a proportional and contiguous allocation, whose egalitarian welfare is already $\frac{1}{n}$. Inspired by this, a natural idea to compute an MEW allocation is to first guess the value of the egalitarian welfare, then adopt the moving-knife algorithm where θ is set as the guessed value. If the moving-knife procedure works, we can trivially achieve a PTAS algorithm. Unfortunately, when the threshold θ exceeds $\frac{1}{n}$, moving-knife may fail to find an allocation with egalitarian welfare θ when such allocation exists.

$$f_1(x) = \begin{cases} 2 - \epsilon, & x \in [0, \frac{1}{4}] \cup [\frac{3}{4}, 1] \\ \epsilon, & x \in (\frac{1}{4}, \frac{3}{4}). \end{cases} \quad f_2(x) = f_3(x) = \begin{cases} \frac{4}{3} - \frac{2\epsilon}{3}, & x \in [0, \frac{3}{4}] \\ 2\epsilon, & \text{otherwise.} \end{cases}$$

Counter-Example of Moving-Knife. Consider a counter-example with three agents. Assume ϵ is sufficiently small, and we are given the value density functions as follows,

In the example, the optimal allocation is $A_1 = [\frac{3}{4}, 1]$, $A_2 = [0, \frac{3}{8}]$ and $A_3 = [\frac{3}{8}, \frac{3}{4}]$ with MEW as $\frac{1}{2} - \frac{\epsilon}{4}$. However, it cannot be found through the moving-knife algorithm, as agent 1 first calls stop at $\frac{1}{4}$, and assume we allocate the second interval with value $\frac{4}{3} - \frac{\epsilon}{3}$ to agent 2. When $\epsilon \rightarrow 0$, agent 3 can only receive the remaining part with value $(\frac{4}{3} - \frac{2\epsilon}{3}) \times (\frac{3}{4} - \frac{1}{4} - \frac{3}{8}) = \frac{1}{6} - \frac{\epsilon}{12} < \frac{1}{2} - \frac{\epsilon}{4}$. Further, in Sect. 3.2, we provide a constant-inapproximability result.

3.1 Agents with Fixed Permutation

We first consider an easy case when the order of agents receiving the cake is fixed, that is, for two agents i and j where $i < j$ in the permutation, i needs to receive a bundle before j . We show that the optimal allocation could be found in polynomial time. An observation is that we could directly extend it to the general case of constant n by enumerating all the permutations and handling each of them.

Next, we begin to present our algorithm for a fixed permutation. As shown in Algorithm 1, we design a linear programming-based algorithm to compute an optimal allocation. In our latter analysis, without loss of generality, we assume the fixed permutation is just $(1, 2, \dots, n)$. We define a series of knives $\{\delta_0, \delta_1, \dots, \delta_n\}$ where $\delta_i \in [0, 1]$, and let $\delta_0 = 0$. For each agent i , we consider allocating the interval $[\delta_{i-1}, \delta_i]$ to her. Denote all the different breakpoints of f_1, \dots, f_n by $0 < p_0 < p_1 < \dots < p_m = 1$, where $m \in \mathbb{Z}^+$. Let d_i be the distance between knife δ_i and its right-closest breakpoint $p_{\sigma(i)}$ (i.e. $d_i \triangleq \min_{p_k} (p_k - \delta_i)$ where $p_k > \delta_i$, and $p_{\sigma(i)} \triangleq \arg \min_{p_k} (p_k - \delta_i)$). Specifically, if $\delta_i = 1$, by this definition, $p_{\sigma(i)}$ and d_i would be 1 and 0, respectively.

$$\begin{aligned} \max \quad & \hat{y} & (4) \\ \text{subject to} \quad & 0 \leq x_i \leq d_i, & i \in [n] & (5) \\ & v_{1,\sigma(1)}x_1 = \hat{y}, & & (6) \\ & v_{i+1,\sigma(i+1)}x_{i+1} - v_{i+1,\sigma(i)}x_i = \hat{y}, & i \in [n-1]. & (7) \end{aligned}$$

Now, we move each knife δ_i from left to right within distance d_i to achieve a maximum increase in egalitarian welfare using linear program 4. In the linear program, we use \hat{y} to denote the increase of egalitarian welfare and use x_i to denote the distance that knife δ_i moves. For simplicity, we denote agent i 's value to an interval $[x_j, x_{j+1}]$ with a constant value density as $v_{i,j+1}$. The linear

Algorithm 1: Algorithm for computing maximum egalitarian welfare

Input: Each agent i 's value density function f_i with all the breakpoints $p_{i \in [m]}$, and a permutation of agents (i_1, i_2, \dots, i_n) .

Output: An allocation \mathcal{A} that maximizes egalitarian welfare.

- 1 Let $y \leftarrow 0$ denote the maximum egalitarian welfare under this permutation;
 - 2 Let $\delta_k \leftarrow 0$ denote the knife position for agent i_k for each $k \in [n]$;
 - 3 Let $\hat{y} \leftarrow \infty$ denote the increment of y within each iteration;
 - 4 **while** $\hat{y} \neq 0$ **do**
 - 5 Let $p_{\sigma(k)}$ denote the right-closest breakpoint of δ_k and $d_k \leftarrow p_{\sigma(k)} - \delta_k$;
 - 6 Run **linear program 4** to solve the optimal \hat{y} and x_k ;
 - 7 Update $\delta_k \leftarrow \delta_k + x_k$ and $y \leftarrow y + \hat{y}$;
 - 8 Let $A_i \leftarrow [\delta_{i-1}, \delta_i]$ for each $i \in [n]$;
 - 9 **return** $\mathcal{A} = (A_1, \dots, A_n)$
-

program is subject to, first, each agent's utility to her bundle has the same increase \hat{y} , hence, the output allocation is equitable. Second, to compute such an increase for each agent, we need to focus on the interval with constant value density, so the maximum distance that δ_i moves cannot exceed d_i . If δ_i moves and y increases, we update d_i and repeat the process. If $\hat{y} = 0$, we further prove y is the MEW value.

Theorem 2. *Algorithm 1 will output an MEW allocation for a fixed permutation of agents in polynomial time.*

Proof. Let $\text{OPT} = ([0, o_1], \dots, [o_{n-1}, o_n])$ represent the optimal allocation. We first claim that OPT is both equitable and unique via the following two lemmas.

Lemma 2. *OPT is equitable.*

Proof. For the sake of contradiction, we assume there exist $i \neq j \in [n]$ such that $v_i([o_{i-1}, o_i]) \neq v_j([o_{j-1}, o_j])$. Hence, there exist two adjacent agents such that one has the smallest utility and the other has a higher utility. Otherwise, the utilities of all the agents would be equal, which violates our assumption. Without loss of generality, we assume agent i and agent $i + 1$ are such two agents. Due to the intermediate value theorem, we could find o'_i such that $v_i([o_{i-1}, o'_i]) = v_{i+1}([o'_i, o_{i+1}])$. This operation clearly improves agent i 's utility and does not decrease agent $i + 1$'s utility to the original minimum. By repeating this operation, we can improve egalitarian welfare, which contradicts the assumption of the optimal solution. Note that the case where agent n has the smallest utility can be handled in a similar way. □

Lemma 3. *An equitable allocation is also unique.*

Proof. By contradiction, we assume there are two equitable allocations, and denote their cut points as $\{x_0 = 0, x_1, \dots, x_n = 1\}$ and $\{y_0 = 0, y_1, \dots, y_n = 1\}$ respectively. Since the allocations are different, we can find a minimal interval

$[x_i, x_j]$ and $[y_i, y_j]$ such that $x_i = y_i, x_j = y_j$, and for any $k \in [i + 1, j - 1]$, $x_k \neq y_k$. Without loss of generality, we assume $x_{i+1} < y_{i+1}$. We consider the following two cases.

1. For all $k \in [i + 2, j - 1]$, $x_k < y_k$. Since $[x_i, x_{i+1}]$ is a subset of $[y_i, y_{i+1}]$ and each agent is hungry, we have $v_{i+1}([x_i, x_{i+1}]) < v_{i+1}([y_i, y_{i+1}])$. Similarly, $v_j([x_{j-1}, x_j]) > v_j([y_{j-1}, y_j])$. However, since the two allocations are equitable, we have $v_{i+1}([x_i, x_{i+1}]) = v_j([x_{j-1}, x_j])$ and $v_{i+1}([y_i, y_{i+1}]) = v_j([y_{j-1}, y_j])$, which leads to a contradiction.
2. Otherwise, there exists at least one index $k \in [i + 2, j - 1]$ such that $x_k > y_k$. We further assume k is the smallest index, i.e. for all $\ell \in [i + 2, k - 1], x_\ell < y_\ell$. We already know $v_{i+1}([x_i, x_{i+1}]) < v_{i+1}([y_i, y_{i+1}])$. As $[x_{k-1}, x_k]$ is a superset of $[y_{k-1}, y_k]$, we have $v_j([x_{k-1}, x_k]) > v_j([y_{k-1}, y_k])$. Same to the analysis in the above case, this leads to a contradiction.

Combining the two above cases, the lemma is concluded. □

Back to the original proof. Due to our previous description, the utilities of these agents keep the same during running Algorithm 1. If $\delta_n = 1$ when the algorithm terminates, according to Lemma 2 and Lemma 3, the output allocation must be the unique optimal solution. Otherwise, there will be an interval of cake left, and \hat{y} will be zero. We claim this case cannot happen and δ_n will reach 1 when Algorithm 1 terminates.

Lemma 4. *If $\hat{y} = 0$, then $\delta_n = 1$.*

Proof. According to the definition of linear program 4, if $\hat{y} = 0$, that implies $x_1 = \dots = x_n = 0$. If the proposition is false (i.e. $\delta_n < 1$), our main idea is to add a sufficiently small constant to each of x_i and \hat{y} so that the solution remains feasible but has a larger objective value. Let x'_i (for $i = 1, \dots, n$) and \hat{y}' be the new values of x_i and \hat{y} . Consider the following three constants λ, μ, ϵ and the new objective value \hat{y}' ,

$$\lambda = \max_{i \in [n-1]} \left\{ \frac{v_{i+1, \sigma(i)}}{v_{i+1, \sigma(i+1)}} \right\}, \mu = \max_{i \in [n-1]} \left\{ \frac{1}{v_{i+1, \sigma(i+1)}} \right\},$$

$$\epsilon = \min_{i \in [n]} \left\{ \frac{d_i}{2\lambda^{i-1}}, \frac{1}{v_{1, \sigma(1)}} \cdot \frac{d_i(1 - \lambda)}{2\mu(1 - \lambda^{n-1})} \right\}, \hat{y}' = v_{1, \sigma(1)}\epsilon.$$

Next, we set $x'_1 = \epsilon$ and $x'_{i+1} = \frac{v_{i+1, \sigma(i)}}{v_{i+1, \sigma(i+1)}} x'_i + \frac{\hat{y}'}{v_{i+1, \sigma(i+1)}}$ for $i = 1, \dots, n - 1$. Clearly, \hat{y}' is strictly positive. After that, we claim that $(x'_1, \dots, x'_n, \hat{y}')$ is also an feasible solution. It is not hard to verify the constraints (6) and (7) are satisfied by the definition of x'_i and \hat{y}' . Additionally, since $x'_1 = \epsilon \leq \frac{1}{2} \min_{i \in [n]} \left(\frac{d_i}{\lambda^{i-1}} \right) \leq \frac{1}{2} d_1 \leq d_1$. For $i = 1, \dots, n - 1$, constraints (5) can be verified by the following inequality:

$$\begin{aligned}
 x'_{i+1} &= \frac{v_{i+1, \sigma(i)}}{v_{i+1, \sigma(i+1)}} x'_i + \frac{\hat{y}'}{v_{i+1, \sigma(i+1)}} \leq \lambda x'_i + \mu \hat{y}' \leq \dots \\
 &\leq \lambda^i x'_1 + \mu \left(1 + \dots + a_0^{i-1}\right) \hat{y}' && \text{(By the definition of } a_0 \text{ and } b_0) \\
 &\leq \frac{1}{2} d_{i+1} + \mu \left(1 + \dots + \lambda^{i-1}\right) \hat{y}' && \text{(By the first part of the definition of } \epsilon) \\
 &\leq \frac{1}{2} d_{i+1} + \frac{1}{2} d_{i+1} = d_{i+1}. && \text{(By the second part of the definition of } \epsilon)
 \end{aligned}$$

Thus, $(x'_1, \dots, x'_n, \hat{y}')$ is also feasible and has a larger objective value, leading to a contradiction. \square

Due to Lemma 4 and our prior statements, the output allocation is equitable. Since the optimal allocation is unique and also equitable, we conclude that our algorithm achieves the optimal allocation.

Finally, we show this algorithm also runs in polynomial time. Within each iteration, we will find an optimal solution at a vertex of the feasible region. Since there are $3n$ constraints of the linear program and the vertex has $n + 1$ dimensions, then at least one of the constraints (5) is tight. Suppose x_i satisfies $x_i = 0$ or d_i . If $x_i = 0$, then x_i, \dots, x_1 would be all zero, and \hat{y} would also be zero, which contradicts to the assumption of $\hat{y} \neq 0$. Hence, $x_i = d_i$ and δ_i will be updated to $p_{\sigma(i)}$ at the end of this iteration.

We observe that all the knives would keep moving rightward during the algorithm and at least one would encounter a breakpoint within each iteration. Since there are only m breakpoints, the number of total iterations is at most $n \cdot m$. Within each iteration, the time complexity of solving the linear program is also polynomial. Thus, the overall time complexity is polynomial. \square

Note that for general number of agents, we can still enumerate all the permutations and adopt Algorithm 1. This directly leads to the following result.

Corollary 1. *For general cases without any constraint on the permutation of the agents, the problem of computing an MEW allocation can be solved within a complexity of fixed-parameter tractable with respect to the number of agents n .*

3.2 Agents Without Permutation Constraints

In this section, we present the inapproximability results for a general number of agents and no constraints on the permutation of the agents. To illustrate our reduction, we first provide proof of the inapproximability of 2 in Theorem 3. Although it has been demonstrated by Aumann et al. (2013), we also give our proof here. Following that, we expand upon this theorem and adapt it to demonstrate a c -inapproximation ratio in Theorem 4, where c can be any constant.

Theorem 3 (Aumann et al. (2013)). *The problem of computing an MEW allocation with contiguous pieces is NP-hard to approximate to within factor 2 for a general number of agents with no constraint on permutation.*

We present a reduction from the 3-SAT problem. Given a 3-SAT instance Φ with m clauses and $3m$ literals, we construct an instance of maximizing egalitarian welfare with contiguous pieces with four types of agents: *literal agents*, *clause agents*, *logic agents*, and *blocking agents*. Let $\Phi_{i \in [m]}$ denote the i -th clause and $\Phi^{j \in [3m]}$ denote the j -th literal in Φ . Before presenting the formal construction of our reduction. We first show some high-level ideas as follows.

Ideas of Construction. For each clause, we introduce a clause agent c_i and three literal agents $\ell_{3i-2}, \ell_{3i-1}, \ell_{3i}$ for the literals within it. Then, for each literal agent $j \in \{3i + 1, 3i + 2, 3i + 3\}$, let her have three disjoint valued pieces of cake $I_j^{(1)}, I_j^{(2)}, I_j^{(3)}$ such that there is a spacing between every two pieces. We denote those spacing by $s_{3i-2}^{(1)}, s_{3i-2}^{(2)}, s_{3i-1}^{(1)}, s_{3i-1}^{(2)}, s_{3i}^{(1)}, s_{3i}^{(2)}$. Within those spacing, we design to put other agents' valued intervals.

First, as shown in Fig. 1, we let the clause agent c_i have three valued pieces within $s_{3i-2}^{(2)}, s_{3i-1}^{(2)}, s_{3i}^{(2)}$. For each of the three literal agents, we refer to the first two of her valued pieces $I_j^{(1)}, I_j^{(2)}$ including the spacing $s_j^{(1)}$ as her *true interval* and $I_j^{(2)}, I_j^{(3)}$ including the spacing $s_j^{(2)}$ as her *false interval*. If Φ is satisfiable, we aim to let each literal agent receive her true interval $I_j^{(1)}, s_j^{(1)}, I_j^{(2)}$ if the corresponding literal is true in the assignment and false interval $I_j^{(2)}, s_j^{(2)}, I_j^{(3)}$ if the corresponding literal is false. If Φ is unsatisfiable, there always exists a clause such that all three literals are assigned false under any assignment. If we insist that the allocation represents a valid boolean assignment for Φ , there will exist a clause such that the literal agents of it will all receive their false intervals at the same time. That will cause the value received by the corresponding clause agent to be zero, which will lead to egalitarian welfare being zero. Hence, we could no longer make the allocation represent a valid boolean assignment.

To guarantee the consistency and validity of the assignment, we further add some constraints by introducing logic agents. There are two types of logic agents. The first type of logic agent ensures any literal agent ℓ_j cannot get a complete interval including $I_j^{(1)}, I_j^{(2)}$ and $I_j^{(3)}$ (so that we cannot assign both “false” and “true” to a single literal). Otherwise, there will exist a logic agent receiving zero utility. The second type of logic agent guarantees the consistency of the literals. For example, if a variable x occurs twice as the form of x in two literals Φ^{j_1} and Φ^{j_2} , then we introduce two logic agents t_x^1 and t_x^2 . The logic agent t_x^1 has two valued pieces within $s_{j_1}^{(1)}$ and $s_{j_2}^{(2)}$, and t_x^2 has two valued pieces within $s_{j_1}^{(2)}$ and $s_{j_2}^{(1)}$. In this case, the literal agent ℓ_{j_2} cannot receive her false interval if ℓ_{j_1} receives her true interval, and the literal agent ℓ_{j_2} cannot receive her true interval if ℓ_{j_1} receives her false interval. Otherwise, t_x^1 or t_x^2 will receive a utility of zero. The case is similar when the literal agent ℓ_{j_1} receives her false interval.

Construction. The detailed construction is defined as follows:

- For each clause Φ_i , we construct a clause agent c_i .
- For each literal Φ^j , we construct a literal agent ℓ_j .

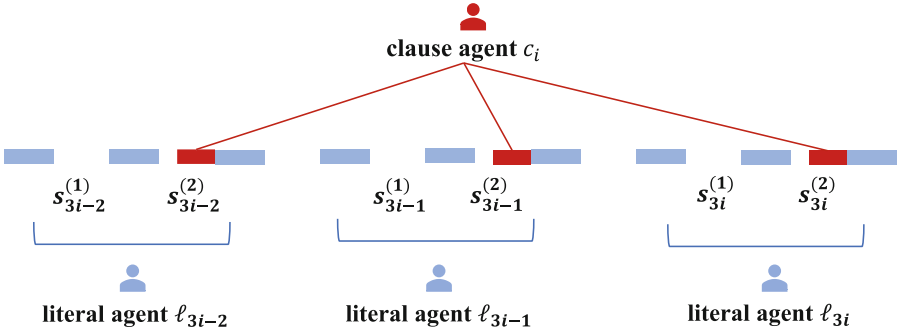


Fig. 1. Construction of clause agents and literal agents.

- Logic agents further contain two types: (1) For each literal Φ^j , we construct a logic agent t_j . (2) If a variable x as well as its negation \bar{x} appear totally k times where $k > 1$, we construct $2(k - 1)$ logic agents t_x^1, \dots, t_x^{2k-1} .
- Further, we construct $9m$ blocking agents d_1, \dots, d_{9m} .

For simplification, we extend the cake from interval $[0, 1]$ to $[0, 36]$ while each agent’s value to the cake remains normalized. We refer to the interval $[0, 18m]$ as the *actual cake*, and $[18m, 36m]$ as the *dummy cake*. Each agent’s value density function to the cake is constructed as follows.

Let $\epsilon > 0$ be a sufficiently small real number. In order to simplify the description of the value density function, we will only define the parts where the density is not equal to ϵ . In fact, in our later argument, we will regard ϵ as 0. It will not affect our proof of inapproximability ratio.

Figure 2 illustrates our construction of the value density functions for both literal agents and clause agents on the actual cake. For clarity, for each literal agent $\ell_{j \in [3m]}$, she has three disjoint valued intervals on the actual cake.

$$f_{\ell_j}(x) = \frac{1}{6}, \text{ for } x \in [6j - 6, 6j - 5] \cup [6j - 4, 6j - 3] \cup [6j - 2, 6j - 1].$$

Let \hat{k} be the maximum number of occurrences of any variable x in Φ , formally,

$$\hat{k} \triangleq \max_x \sum_{j \in [3m]} \mathbf{1}(\Phi^j = x \vee \Phi^j = \bar{x}).$$

Then, we evenly divide the spacing interval $[6j - 5, 6j - 4]$ into \hat{k} disjoint sub-intervals and $[6j - 3, 6j - 2]$ into $\hat{k} + 1$ disjoint sub-intervals. For each clause agent $c_{i \in [m]}$, she only has value to three disconnected intervals. When $x \in [18i - 14 - \frac{1}{k+1}, 18i - 14] \cup [18i - 8 - \frac{1}{k+1}, 18i - 8] \cup [18i - 2 - \frac{1}{k+1}, 18i - 2]$, $f_{c_i}(x) = \frac{\hat{k}+1}{3}$.

As shown in Fig. 3, for each logic agent t_j of the first type constructed from Φ^j , we define her value density function as follows,

$$f_{t_j}(x) = \begin{cases} \frac{\hat{k}}{2}, & x \in [6j - 5, 6j - 5 + \frac{1}{\hat{k}}] \\ \frac{\hat{k}+1}{2}, & x \in [6j - 3, 6j - 3 + \frac{1}{\hat{k}+1}]. \end{cases}$$

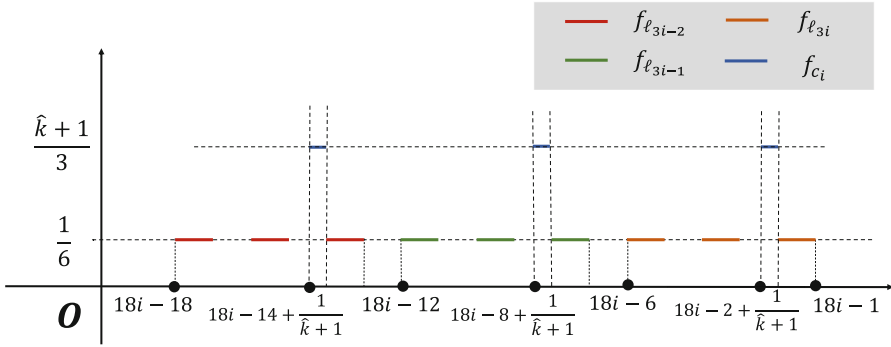


Fig. 2. Value density function of literal agents and clause agents.

For each logic agent of the second type, we consider her corresponding variable y . Assume y and \bar{y} appears k times totally in Φ . We only focus on the case where $k > 1$. Assume y first appears in literal Φ^a , and without loss of generality, assume its (including \bar{y}) s -th appearance ($2 \leq s \leq k$) is in literal Φ^b . We construct two literal agents of the second type t_y^{2s-3} and $t_{\bar{y}}^{2s-2}$.

If $\Phi^b = y$, we define their value density functions as

$$f_{t_y^{2s-3}}(x) = \begin{cases} \frac{\hat{k}}{2}, & x \in [6a - 5 + \frac{s-1}{k}, 6a - 5 + \frac{s}{k}] \\ \frac{\hat{k}+1}{2}, & x \in [6b - 3 + \frac{s-1}{k+1}, 6b - 3 + \frac{s}{k+1}], \end{cases}$$

$$f_{t_{\bar{y}}^{2s-2}}(x) = \begin{cases} \frac{\hat{k}+1}{2}, & x \in [6a - 3 + \frac{s-1}{k+1}, 6a - 3 + \frac{s}{k+1}] \\ \frac{\hat{k}}{2}, & x \in [6b - 5 + \frac{s-1}{k}, 6b - 5 + \frac{s}{k}]. \end{cases}$$

If $\Phi^b = \bar{y}$, we swap the second interval of the two agents.

$$f_{t_y^{2s-3}}(x) = \begin{cases} \frac{\hat{k}}{2}, & x \in [6a - 5 + \frac{s-1}{k}, 6a - 5 + \frac{s}{k}] \\ \frac{\hat{k}}{2}, & x \in [6b - 5 + \frac{s-1}{k}, 6b - 5 + \frac{s}{k}], \end{cases}$$

$$f_{t_{\bar{y}}^{2s-2}}(x) = \begin{cases} \frac{\hat{k}+1}{2}, & x \in [6a - 3 + \frac{s-1}{k+1}, 6a - 3 + \frac{s}{k+1}] \\ \frac{\hat{k}+1}{2}, & x \in [6b - 3 + \frac{s-1}{k+1}, 6b - 3 + \frac{s}{k+1}]. \end{cases}$$

In addition to the above intervals for literal agents, each literal agent also has three disconnected valued intervals on the dummy cake. When $x \in [18m + 6j - 6, 18m + 6j - 5] \cup [18m + 6j - 4, 18m + 6j - 3] \cup [18m + 6j - 2, 18m + 6j - 1]$, $f_{l_j}(x) = \frac{1}{6}$ (Fig. 4).

The blocking agents are designed to block the valuable intervals of the above agents. Each blocking agent has value to one piece between them:

$$f_{d_j}(x) = 1, \text{ for } x \in [18m + 2j - 1, 18m + 2j] \text{ and } j \in [1, 9m].$$

Under this construction, the integral of the value density function for each agent over the entire cake is equal to 1. Now we are ready to provide the formal proof.

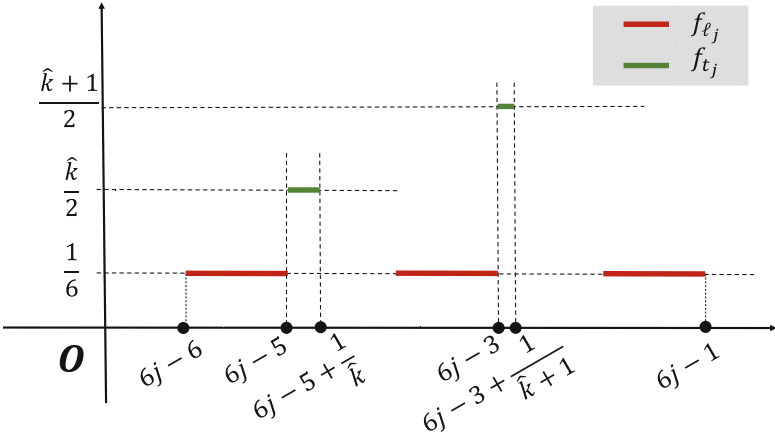


Fig. 3. The value density function of t_j . In this case, literal agent l_j (whose value density function is colored red) cannot receive the three intervals $[6j - 6, 6j - 5]$, $[6j - 4, 6j - 3]$ and $[6j - 2, 6j - 1]$ at the same time. Otherwise, the utility of logic agent t_j (whose value density function is colored green) will be zero. (Color figure online)

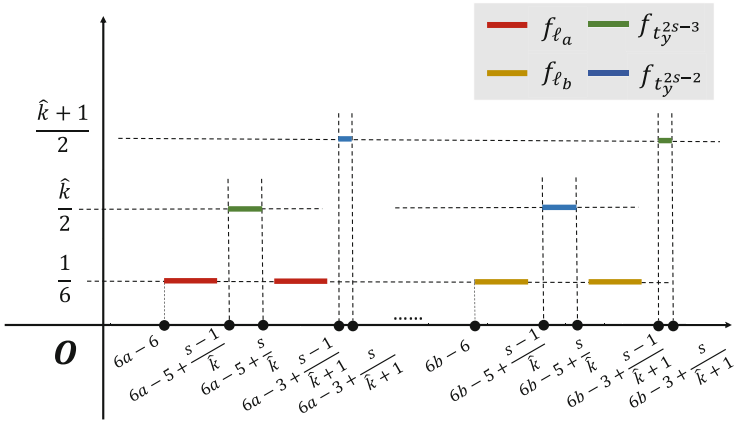


Fig. 4. The value density functions of t_y^{2s-3} and t_y^{2s-2} when literals Φ^a, Φ^b are both y . In this case, when literal agent l_a (whose value density function is colored red in the above figure) receives her true intervals, literal agent l_b (whose value density function is colored yellow) is forbidden to receive her false intervals. Otherwise, logic agent t_y^{2s-3} (whose value density function is colored green) will have zero utility. For a similar reason, literal agents l_a and l_b cannot respectively receive their true intervals and false intervals at the same time. Otherwise, logic agent t_y^{2s-2} (whose value density function is colored blue) will have zero utility. The construction of the case when $\Phi^a = y$ and $\Phi^b = \bar{y}$ is similar. (Color figure online)

Proof. Suppose the 3-SAT instance is a yes instance (i.e., there exists a valid assignment such that $\Phi = true$). We construct an allocation as follows.

In the assignment, if a literal Φ^j is assigned *true*, then we let the literal agent ℓ_j receive her true interval $[6j - 6, 6j - 3]$. If the clause agent $c_{\lceil j/3 \rceil}$ has not received any interval, we let her take the interval $\left[6j - 2 - \frac{1}{k+1}, 6j - 2\right]$. Further, we let all the logic agents that have value on $\left[6j - 3, 6j - 2 - \frac{1}{k+1}\right]$ receive their valued intervals. Otherwise, if Φ^j is assigned *false*, we let ℓ_j receive her false interval $[6j - 4, 6j - 1]$, and let all the logic agents receive their valued intervals on $[6j - 5, 6j - 4]$. Further, we let each blocking agent receive her valued interval. For each contiguous interval that remains unallocated, we allocate it to an arbitrary agent who receives the interval adjacent to it.

Under this allocation, it is straightforward to find that each literal agent receives value $\frac{1}{3}$, while each blocking agent receives a value of 1. For each clause Φ_i , there exists at least one literal $\Phi^{j \in [3i-2, 3i]}$ assigned *true*. Therefore, the interval received by each clause agent c_i will be at least $\frac{1}{3}$. For each logic agent t_j of the first type, since ℓ_j only receives one of $[6j - 5, 6j - 4]$ and $[6j - 3, 6j - 2]$, she can always receive an interval with value $\frac{1}{2}$. For each logic agent of the second type t_y^j , denote the two intervals that she has value to by $J_{y_1}^j$ and $J_{y_2}^j$. Assume y appears k times, then $k - 1$ logic agents can receive $J_{y_1}^j$ and obtain value $\frac{1}{2}$ on the first appearance of y . Since the assignment is consistent, by our construction, the other $k - 1$ agents can always receive $J_{y_2}^j$ with value $\frac{1}{2}$.

Hence, the maximum egalitarian welfare under a yes instance is at least $\frac{1}{3}$.

Suppose the 3-SAT instance is a no instance, that is, there is no assignment such that $\Phi = true$. We prove the maximum egalitarian welfare will be at most $\frac{1}{6}$ by contradiction.

Assuming the maximum egalitarian welfare is more than $\frac{1}{6}$, then, each literal agent ℓ_j needs to receive a length of more than 2 on the cake (since the maximum density of f_{ℓ_j} is $\frac{1}{6}$). She cannot receive anything from the dummy cake, otherwise, at least one blocking agent will receive a utility of zero. Therefore, we only consider the case that each literal agent receives only part of the actual cake, hence she will surely receive an interval containing either $[6j - 5, 6j - 4]$ or $[6j - 3, 6j - 2]$.

Denote the corresponding variable of Φ^j as y . To ensure each logic agent of the second type receives more than $\frac{1}{6}$, all the literal agents ℓ_a where $\Phi^a = y$ cannot receive $[6j - 3, 6j - 2]$ and all the literal agents ℓ_b where $\Phi^b = \bar{y}$ cannot receive $[6j - 5, 6j - 4]$. Moreover, to ensure each clause agent receives more than $\frac{1}{6}$, at least one literal agent in each clause cannot receive $[6i - 3, 6i - 2]$. If such allocation exists, we can construct an assignment that if the literal agent ℓ_j receives $[6j - 5, 6j - 4]$, we assign $\Phi^j = true$. If the literal agent ℓ_j receives $[6j - 3, 6j - 2]$, we assign $\Phi^j = false$. The assignment is consistent according to the above analysis, and a literal is assigned *true* in each clause.

Hence, we conclude Φ is satisfiable, which leads to the contradiction. Therefore, the inapproximability is at least $\frac{\frac{1}{3}}{\frac{1}{6}} = 2$ and the theorem holds. \square

Theorem 4. *The problem of computing MEW with contiguous pieces is NP-hard to approximate within any constant factor for a general number of agents with no constraint on permutation.*

Construction. We follow the same idea as the construction in the above example on the actual cake while the dummy cake is not needed in the following construction. Let $r \in \mathbb{Z}^+$ be a constant integer.

- For literal Φ^j , we still construct a literal agent ℓ_j . Instead of constructing three disconnect intervals that she has value to, we construct $2r + 1$ such disconnect intervals. Denote these intervals by $\mathcal{I}_j = \{I_j^{(1)}, \dots, I_j^{(2r+1)}\}$, and the interval that splits $I_j^{(k)}$ and $I_j^{(k+1)}$ by $s_j^{(k)}$. All these intervals are disjoint with each other. In our new construction, we call the interval from $I_j^{(1)}$ to $I_j^{(r+1)}$ as the agent ℓ_j 's true interval T_j (which begins at the leftmost point of $I_j^{(1)}$ and ends at the rightmost point of $I_j^{(r+1)}$), and the interval from $I_j^{(r+1)}$ to $I_j^{(2r+1)}$ as false interval F_j (which begins at the leftmost point of $I_j^{(r+1)}$ and ends at the rightmost point of $I_j^{(2r+1)}$).

Our goal is to make a literal agent receive the entire true interval when the 3-SAT instance is a yes instance. In the case of a no instance, we want to limit her bundle to at most one interval of \mathcal{I}_j .

- For clause Φ_i , we construct r^3 clause agents instead of only one clause agent in the previous construction. In particular, for each of the r^3 combinations $\{s_{3i-2}^{(j_1)} \subset F_{3i-2}, s_{3i-1}^{(j_2)} \subset F_{3i-1}, s_{3i}^{(j_3)} \subset F_{3i}\}$, we construct a clause agent that has value to a sub-intervals of each of the three disconnect intervals. The design of the clause agent in this context is adapted from the previous proof. If the three literal agents of the same clause all receive their false interval, there will always exist a clause agent receiving zero utility.
- For each literal, we construct r^2 logic agents of the first type. In particular, for each of the r^2 combinations $\{s_i^{(j_1)} \subset T_i, s_i^{(j_2)} \subset F_i\}$, we construct a logic agent of the first type that has value to a sub-intervals of each of the two disconnected intervals. These agents are also used to prevent any literal agent from receiving her true and false intervals at the same time.
- For each variable z , assume it first appears in Φ^i . If it appears more than once, for each of its appearances in Φ , we construct $2r^2$ logic agents of the second type. In particular, if $\Phi^j = z$, the first r^2 agents have value to the sub-intervals of each combination $\{s_i^{(i')} \subset T_i, s_j^{(j')} \subset F_j\}$, and the latter r^2 agents have value to the sub-intervals of each combination $\{s_i^{(i')} \subset F_i, s_j^{(j')} \subset T_j\}$. Otherwise, if $\Phi^j = \bar{z}$, the first r^2 agents have value to the sub-intervals of each combination $\{s_i^{(i')} \subset T_i, s_j^{(j')} \subset T_j\}$, and the latter r^2 agents have value to the sub-intervals of each combination $\{s_i^{(i')} \subset F_i, s_j^{(j')} \subset F_j\}$.

After constructing these three types of agents, if there are n' agents that have value to a sub-interval of $s_i^{(j)}$, we divide $s_i^{(j)}$ into n' disjoint intervals, and

let each agent has value only on one sub-interval. Moreover, for each agent, if there are multiple intervals that she has value to, we let her value to each of these intervals equal and normalized to 1 in total.

Formal Proof. Assume Φ is a yes instance, then there exists a valid assignment such that $\Phi = true$. We construct an allocation \mathcal{A} from a satisfying assignment as follows.

We traverse j from 1 to $3m$. If the literal Φ^j is assigned *true*, then we let literal agent ℓ_j receive her true interval T_j (i.e., $\mathcal{A}_{\ell_j} = T_j$). If a corresponding clause agent or a logic agent has not received any interval, we let her receive a sub-interval of $s_j^{(k)} \in F_j$ that she has value to. If Φ^j is assigned *false*, we let ℓ_j receive her false interval F_j (i.e., $\mathcal{A}_{\ell_j} = F_j$). We also let each of the corresponding logic agents that has value on T_j and has not received any bundle receive an interval that she has value to. For each contiguous interval that remains unallocated, we allocate it to an arbitrary agent who receives the interval adjacent to it.

Then we show that $\mathcal{E}\mathcal{W}(\mathcal{A})$ is at least $\frac{1}{3}$. Since each literal agent takes either her true interval or false interval, she will receive $\frac{1}{2}$ in \mathcal{A} . Since $\Phi = true$ under this assignment, in each clause, there is at least one literal that is assigned *true*, so all the r^3 clause agents can receive an interval from this literal agent's false interval and obtain value $\frac{1}{3}$. Each logic agent of the first type can receive value $\frac{1}{2}$ as the corresponding literal agent can only receive either her true interval or her false interval. Since the assignment is consistent, each logic agent of the second type can also receive $\frac{1}{2}$ for the same reason as we have shown in the 2-inapproximation result. Therefore, the maximum egalitarian welfare under a yes instance is $\frac{1}{3}$.

Now we consider the case that Φ is a no instance, so no assignment can satisfy $\Phi = true$. We prove $\mathcal{E}\mathcal{W}(\mathcal{A})$ will be no more than $\frac{1}{2r+1}$ for any allocation \mathcal{A} by contradiction.

Suppose the maximum egalitarian welfare is more than $\frac{1}{2r+1}$, we show that there will exist a satisfying assignment. Assume for each literal agent ℓ_i , $|I_i^{(j)}| = |s_i^{(j)}| = 1$. Then, each literal agent needs to receive an interval J_i with $|J_i| > 2$, $|J_i \cap \mathcal{I}_i| > 1$ and $|J_i \setminus \mathcal{I}_i| \geq 1$. J_i cannot intersect with both T_i and F_i , otherwise, there will be at least one logic agent of the first type that receives 0. We assume $\Phi^i = z$ and, without loss of generality, $J_i \subseteq T_i$. Then, to avoid any logic agent of the second type receiving 0, for any j such that $\Phi^j = z$, we need to allocate an interval $J_j \subseteq T_j$ to ℓ_j using the similar analysis in the 2-inapproximation result. If $\Phi^j = \bar{z}$, ℓ_j needs to receive an interval $J_j \subseteq F_j$. Further, to avoid any clause agent receiving 0, at least one literal agent corresponding to this clause cannot receive any sub-interval from $s_j^{(r+1)}$ to $s_j^{(2r)}$. Hence, she needs to receive a bundle on the true interval. Then, we consider the following assignment. For a literal agent ℓ_i , if $J_i \subseteq T_i$, we assign Φ^i as *true*. If $J_i \subseteq F_i$, we assign Φ^i as *false*. From the above analysis, the assignment is consistent, and there is a *true* literal in each clause, indicating $\Phi = true$ under this assignment and leading to a contradiction.

We have shown that the inapproximability factor is $\frac{1/3}{1/(2r+1)} = \frac{2r+1}{3}$. Since r can be any constant, Theorem 4 holds. \square

4 Conclusion and Future Work

In this paper, we consider the problem of maximizing egalitarian welfare for hungry agents with and without contiguous constraints respectively. In the absence of contiguous constraints, the problem can be readily solved using linear programming. With contiguous constraints, there exists a non-trivial algorithm to output an optimal allocation for agents with fixed permutation in polynomial time, which indicates a fixed parameter tractable algorithm with respect to the number of agents n . Moreover, if a fixed permutation is not required, we provide a constant-inapproximability result.

As we consider hungry agents in our work, an interesting future direction is to generalize the setting to agents without hungry assumption. An intuitive idea is to first apply our algorithm, however, as an agent may have a value of 0 to the next interval, the egalitarian welfare we calculate may not increase and the knives may not move. In this case, we may deploy another linear program to maximize the summation of the distance of each knife's position while maintaining that each agent's utility to her contiguous piece does not decrease. Further, we may consider a more general case where the constraints for contiguous are subjective to the agents, which means some agents can receive a non-contiguous bundle while others need to receive a contiguous one.

References

- Asadpour, A., Saberi, A.: An approximation algorithm for max-min fair allocation of indivisible goods. *SIAM J. Comput.* **39**(7), 2970–2989 (2010). <https://doi.org/10.1137/080723491>
- Aumann, Y., Dombb, Y.: The Efficiency of Fair Division with Connected Pieces. *ACM Trans. Econ. Comput.* **3**(4), 1–16 (2015)
- Aumann, Y., Dombb, Y., Hassidim, A.: Computing socially-efficient cake divisions. In: *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 343–350 (2013)
- Aziz, H., Mackenzie, S.: A discrete and bounded envy-free cake cutting protocol for any number of agents. [arXiv:1604.03655](https://arxiv.org/abs/1604.03655) [cs.DS] (2017)
- Barbanel, J.B., Brams, S.J.: Two-person cake-cutting: the optimal number of cuts. Available at SSRN 1946895 (2011)
- Bei, X., Chen, N., Hua, X., Tao, B., Yang, E.: Optimal proportional cake cutting with connected pieces. In: *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pp. 1263–1269 (2012)
- Bei, X., Chen, N., Huzhang, G., Tao, B., Wu, J.: Cake cutting: envy and truth. In: *IJCAI*, pp. 3625–3631 (2017)
- Bertsimas, D., Farias, V.F., Trichakis, N.: The price of fairness. *Oper. Res.* **59**(1), 17–31 (2011)

- Brams, S.J., Feldman, M., Lai, J., Morgenstern, J., Procaccia, A.D.: On maxsum fair cake divisions. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), pp. 1285–1291 (2012)
- Brams, S.J., Jones, M.A., Klamler, C., et al.: Better ways to cut a cake. *Not. AMS* **53**(11), 1314–1321 (2006)
- Brams, S.J., Taylor, A.D.: An envy-free cake division protocol. *Am. Math. Monthly* **102**(1), 9–18 (1995). <http://www.jstor.org/stable/2974850>
- Caragiannis, I., Kaklamanis, C., Kanellopoulos, P., Kyropoulou, M.: The efficiency of fair division. *Theory Comput. Syst.* **50**(4), 589–610 (2012)
- Cohler, Y.J., Lai, J.K., Parkes, D.C., Procaccia, A.D.: Optimal envy-free cake cutting. In: Proceedings of AAAI Conference on Artificial Intelligence (AAAI), pp. 626–631 (2011)
- Dubins, L.E., Spanier, E.H.: How to cut a cake fairly. *Am. Math. Monthly* **68**(1P1), 1–17 (1961)
- Even, S., Paz, A.: A note on cake cutting. *Discret. Appl. Math.* **7**(3), 285–296 (1984)
- Güler, O., den Hertog, D., Roos, C., Terlaky, T., Tsuchiya, T.: Degeneracy in interior point methods for linear programming: a survey. *Ann. Oper. Res.* **46**(1), 107–138 (1993)
- Jones, M.A.: Equitable, envy-free, and efficient cake cutting for two people and its application to divisible goods. *Math. Mag.* **75**(4), 275–283 (2002)
- Kelly, F.: Charging and rate control for elastic traffic. *Eur. Trans. Telecommun.* **8**(1), 33–37 (1997)
- Roughgarden, T.: Algorithmic game theory. *Commun. ACM* **53**(7), 78–86 (2010)
- Steinhaus, H.: The problem of fair division. *Econometrica* **16**(1), 101–104 (1948)
- Steinhaus, H.: Sur la division pragmatique. *Econometrica* **17**(1949), 315–319 (1949)
- Tao, B.: On existence of truthful fair cake cutting mechanisms. In: Proceedings of the 23rd ACM Conference on Economics and Computation, pp. 404–434 (2022)