Efficiency, fairness and incentives in resource allocation

Huzhang, Guangda

2018

Huzhang, G. (2018). Efficiency, fairness and incentives in resource allocation. Doctoral thesis, Nanyang Technological University, Singapore.

https://hdl.handle.net/10356/82541

https://doi.org/10.32657/10220/46645

Efficiency, Fairness and Incentives in Resource Allocation

Huzhang Guangda

Division of Mathematical Sciences
School of Physical and Mathematical Sciences

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Doctor of Philosophy

2018

Acknowledgements

First of all, I would like to thank my supervisor Bei Xiaohui for his guidance on research. We studied a lot of interesting topics and I learned a lot from his great ideas and suggestions. I also got lots of help in graduate study, academic presentation, academic writing and even future career. It has been my great honor and fortune to work with Xiaohui for several years.

I would like to thank my previous supervisor Chen Ning for providing me the opportunity to get the enrollment from NTU. I acquired basic knowledge from him. The book "Probability and Computing: Randomized Algorithms and Probabilistic Analysis" is one of the best books I've ever read, which is recommended by him.

Thanks to efforts of many different people from different places, I had the chance to produce several publications during the Ph.D. study. Thank Tao Biaoshuai for paper writing and poster design for our cake cutting paper. Thank Huang Xin and Zhang Shengyu for visiting NTU and working for online roommate allocation paper. Thank Warut Suksompong for visiting NTU and sharing us many novel and valuable ideas on cake cutting problem. Thank Zhang Jianfu for heavy programming and long-period experiments for our community detection paper.

I also would like to thank NTU to enroll me as a Ph.D. student, so that I was able to get close to the most forefront research. I received lots of help in both graduate study and daily life from my friends in school, thank all of them for good old days. Thank Chai Yuxi, Tian Yingtao, Xia Bin and Lin Chengyu for their suggestions in writings.

Last but not least, I would like to thank my family for their unconditional love and consistent support; especially to my wife, thanks for waiting so long.

Contents

Abstract

Resource allocation aims at allocating scarce resources to strategic agents in an efficient and fair manner. Due to its wide applications in real-life, finding such allocations satisfying specific properties is important for both theoretical research and industrial applications. In this thesis, we study three objectives: efficiency, fairness, and incentives, when allocating both divisible resource and indivisible resource.

First, we consider the fair division of a heterogeneous divisible resource, which is well known as the cake cutting problem. We combine fairness and incentive in this part. We focus on designing truthful and envy-free mechanisms with the presence of strategic agents. Most results established by previous studies in this setting all rely crucially on the free disposal assumption, meaning that the mechanism is allowed to throw away part of the resource at no cost. In the first part, we remove this assumption and focus on mechanisms that always allocate the entire resource. We exhibit a truthful envy-free mechanism for cake cutting and chore division for two agents with piecewise uniform valuations, and we complement our result by showing that such a mechanism does not exist when we further require the mechanism to be either anonymous, connected (the connected piece assumption), or position oblivious. Moreover, we give truthful mechanisms for multiple agents with restricted classes of valuations. In the second part, we explore cake cutting mechanism design with piecewise constant valuations. We again require the mechanism to be either connected, non-wasteful or position oblivious and show impossibility results even in the presence of the free disposal assumption.

Next, we study indivisible resource allocation. We consider three fairness no-

tions with indivisible resources: maximin share guarantee, envy-freeness up to one item, and envy-freeness up to the least item. We study whether a mechanism exists when we combine these fairness notions with the connected piece assumption, Pareto optimality and other moderate conditions. In the last part, we study a specific application, which we call the online roommate allocation problem. It is an online allocation problem under the roommate market model introduced in [Chan *et al.*, 2016]. Consider we have a fixed supply of n rooms and a list of $2n$ arriving agents in an online fashion. We have to assign one room to each agent upon his arrival. We show a polynomial-time online algorithm that achieves constant competitive ratio for social welfare maximization. Finally, we show both positive and negative results on the existence of an allocation satisfying different stability conditions in this online setting.

Chapter 1

Introduction

Resource allocation is the process of distributing a set of resources among a group of agents, such that the outcome is efficient for society [Burk, 1938] and reasonably fair to each participant [Steinhaus, 1948]. It is an interdisciplinary research topic involving mathematics, computer science, and economics. Resource allocation problems can be categorized into two groups: *divisible resource allocation* and *indivisible resources allocation*. Divisible resources are called *cakes* and related resource allocation problems are known *cake cutting* problems; Indivisible resources are called *goods* and the each resource is called an *item*. The related allocation problems are called *goods allocation* problems. For simplicity, we call a set of goods *a bundle*. A variant type of resource is a $chore$, such as workloads or costs. Agents may want to limit such resources. *Chore division* is an allocation problem that distributes a divisible chore among the agents, and *chores allocation* is a task that allocates indivisible chores to the agents. Below, we provide several illustrative examples of each type of resource:

- Consider a toy scenario in which a group of guests want to share a cake. Guests may have different preferences. For example, some guests only care about toppings, whereas some guests only care about the size of their pieces. We hope to cut the cake into pieces and allocate these pieces such that no guest thinks another guest's piece is better than his. Sometimes, preferences are private information, so the mechanism (the protocol which yields the allocation from preferences) should be designed to motivate participants to report their preferences truthfully. This example captures a large class of resource allocation scenarios in which participants want to divide a single heterogeneous resource. The cake cutting problem has been a central topic in resource allocation researches for many decades [Brams and Taylor, 1996; Robertson and Webb, 1998]. As Procaccia remarked, Cake cutting is not just child's play [Procaccia, 2013].

- Job scheduling [Graham *et al.*, 1979] is another type of divisible resource allocation problem. An example is a scenario with multiple servers and jobs, in which each job exhibits different efficiencies with different servers and has independent release times and due times. One constraint is that one server can process at most one job and one job can be processed by at most one server anytime. The goal is to assign servers to these jobs to optimise some objectives, such as minimizing total executing time or maximal lateness (completed time minus due time). In this example, we allocate to agents a certain amount of server time. Sometimes, we need to prevent a job from misreporting information to earn more executing time on servers.

- In universities, course allocation is a common problem. We want to design an allocation such that students are able to enrol in the courses they are interested in, but the number of students in each course does not exceed the capacity of the course. In this example, course slots are indivisible resources. There are other similar problems like seat arrangements or room assignments. See an application for courses allocation problem in a previous study [Budish and Cantillon, 2012].

- Organ allocation [Rais and Viana, 2011]is another important application of the resource allocation problem. Many patients need organ transplants, but there is a highly limited number of organs available globally. The proper allocation of organs maximizes the efficiency of saving patients, and there are many different conditions needed to be taken into consideration such as the transit time and the adaptivity of patients. This example also demonstrates that a single patient may have the incentive to misreport information under an improper protocol, with potentially dangerous results for society.

These four examples demonstrate that *efficiency*, *fairness* and *incentives* are three important foci of resource allocation problems. Efficiency benefits the whole society, whereas fairness is related to individuals' perspectives. Given the presence of strategic agents, incentive compatibility is needed to ensure that agents report their private valuations accurately. In the following sections, we introduce these notions separately.

1.1 Efficiency

Efficiency, a major objective of resource allocation problems, is commonly achieved by implementing the *Benthamite social welfare function*[Burk, 1938] (also known as the *utilitarian social welfare function*). It is defined as the sum of all of the individual utilities. Let $SW(A)$ represent the social welfare of an allocation $A = (A_1, A_2, \cdots, A_n)$ to agents $\{a_1, a_2, \cdots, a_n\}$, then

$$SW(A) = \sum_{A_i \in A} v_i(A_i),$$

where A_i is the allocation to a_i and $v_i(A_i)$ is the valuation of A_i to a_i. Allocation to one agent should not have any overlap with any other allocation. Generally we want to allocate all of the resources to agents, so the union of all A_i should be the union of all of the resources.

Another popular social welfare function is the *Nash social welfare function*[Kaneko and Nakamura, 1979], which is defined as the product of agents' utilities instead of the sum. Nash social welfare function takes some account of fairness, as intuitively we need to balance the utilities of the agents to obtain a large product. Formally, the

Nash social welfare function has the following form:

$$SW(A) = \prod_{A_i \in A} v_i(A_i).$$

The third social welfare function we want to consider is the *Rawlsian social welfare function* (or *max-min social welfare function*), which aims to maximize the minimal utility among agents. As it provides both efficiency and strong fairness, we consider the max-min allocation (the allocation maximizing Max-Min social welfare function) to be highly fair and efficient. It has the following form:

$$SW(A) = \min_{A_i \in A} v_i(A_i).$$

The Benthamite social welfare function and Rawlsian social welfare function are located at the two extremes of "the axis of fairness". The Benthamite social welfare function does not consider fairness at all, whereas the Rawlsian social welfare function will maximize the poorest person's happiness. The following interesting characterization connects these two extremes:

$$SW(A, x) = \sum_{A_i \in A} \frac{v_i(A_i)^{1-x}}{1-x},$$

where $x \in [0, +\infty)$ is a parameter. When $x = 0$, this becomes the Benthamite social welfare function; when $x \to \infty$, it becomes the Rawlsian social welfare function. Intuitively, we see that a larger x provides more fairness. Another observation is that the Nash social welfare function is in the middle of this range: when x approaches 1, the above social welfare function is the sum of $\ln(v_i(A_i))$ over a constant, which implies a Nash social welfare function. In this thesis, we focus on the Benthamite social welfare function. Therefore, when we discuss the social welfare function, we are referring to the Benthamite social welfare function unless stated otherwise.

Pareto optimality (or *Pareto efficiency*) is an alternative method for representing efficiency. We say an allocation is Pareto optimal if no other allocation makes no agent worse off and at least one agent better off.

1.2 Fairness

Fairness plays a significant role in resource allocation problems. In a *fair division problem*, we need to find an allocation that all of the agents perceive as fair. The fair cake cutting problem was first discussed as the *proportional* allocation problem[Steinhaus, 1948], in which an allocation is proportional if and only if every agent believes he can get at least $1/n$ of the whole cake (n is the number of participants). Formally, we say an allocation is proportional if and only if

$$v_i(A_i) \geq 1/n, \forall i \in \{1, 2, \cdots, n\},$$

where n is the number of agents. Here the valuation function v_i is normalized, which means that $v_i(R) = 1$ where R is the entire resource.

Finding a proportional allocation is not a difficult task, and we introduce this mechanism in Chapter 2. In addition to proportionality, another stronger fairness notion was proposed, which is called envy-freeness[Gamow and Stern, 1958]. Informally, an allocation is *envy-free* if all of the agents are satisfied with their assigned pieces, after comparing them with the assigned pieces of the other agents. Formally, an allocation is envy-free if

$$v_i(A_i) \geq v_i(A_j), \forall i \neq j.$$

Many studies have examined the envy-free cake cutting problem [Foley, 1967; Varian, 1974]. The cut-and-choose protocol is one of the most straightforward and most well-known methods for solving the cake cutting problem with two agents. The protocol operates by letting the first agent cut the cake into two parts of equal value, and then letting the second agent choose which part he prefers. However, the cut-and-choose protocol cannot be applied to cases with more than two agents, and many studies have tried to find envy-free mechanisms for scenarios with more agents. Although envy-free discrete mechanisms for three agents were independently found by multiple researchers in the 1960s, no algorithms were developed for scenarios with more than three agents. In 1995, a general cake-cutting mechanism was found [Brams and Taylor, 1995], and the authors developed an algorithm to compute the allocation among a group of agents. However, the algorithm has no certain bound on the number of steps. Recently, progress has been made in deriving an envy-free algorithm for four or more of agents with bounded steps [Aziz and Mackenzie, 2016b; Aziz and Mackenzie, 2016a].

There are also several fairness notions for indivisible resources, some of which are weak forms of proportionality and envy-freeness. For example, the maximin share guarantee (MMS guarantee) requires each agent to get a reasonably large value, which is similar to proportionality, but the value may be less than $1/n$. Envy-freeness up to one item (EF1) and envy-freeness up to the least valued item (EFX) require agents to be envy-free after removing an item from their opponents.

1.3 Incentive

There is a fundamental incentive issue in computing allocations: each participating agent is self-interested and wants to receive an allocation with as much utility as possible. Therefore, agents may misreport their valuations to increase the value of their allocations. This motivates the study of cake cutting from a game-theoretical point of view: is there a fair mechanism that could incentivize all of the agents to report their valuation truthfully?

Incentive compatibility [Hurwicz, 1972] is a property of mechanisms that ensures truthful reporting is a dominant strategy for all of the agents. Therefore, incentive compatibility is desired by the host, and it is commonly used in economic and game theory models, such as models of voting and auctions, that have strategic agents. We use the term truthfulness instead of incentive compatibility in this thesis. (so we call a mechanism *truthful* if truthful reporting is a dominant strategy for all of the agents)

As mentioned, the cut-and-choose protocol is a two-agent envy-free cake cutting algorithm. However, the protocol is not truthful. For example, if the first agent values the whole cake equally everywhere, then according to the protocol, he will divide the cake in half and get exactly half of the cake in his valuation. However, if he knows that the second agent only cares about the leftmost quarter of the cake, he can divide the cake into the leftmost quarter and the rest, knowing that the second agent will choose the leftmost quarter and leave him with three-quarters of the cake. Therefore, he may manipulate to move the cut point and gain more, which suggests the mechanism is not truthful.

The cake cutting problem in the presence of strategic agents was first addressed in [Chen *et al.*, 2013]. The authors developed a truthful cake cutting mechanism for piecewise uniform valuations. In their study, they left an open problem: can we have a truthful envy-free mechanism with piecewise constant valuation functions?

1.4 Organization of thesis

This thesis has two main chapters (Chapter 3 and 4) focusing on divisible and indivisible resource allocation problems, respectively.

Divisible resource allocation We study cake cutting problems in two directions in Chapter 3. In the first part, we study cake cutting problems where we regard the cake as an interval $[0, 1]$. We remove the free disposal assumption (which is a commonly used property in related researches) and restrict valuations to be piecewise uniform functions(refer Definition 1). We show that if we add certain requirements for the mechanism on top of being fair and truthful, then no desirable mechanism exists even for two agents. In particular, the impossibility holds when we make one of the following assumptions in addition to truthfulness and envy-freeness: (i) *anonymity*: the mechanism must treat all agents equally; (ii) *the connected piece assumption*: the mechanism must allocate a single interval to each agent; and (iii) *position obliviousness*: the values that the agents receive depend only on the lengths of the pieces desired by various subsets of agents and not on the positions of these pieces. Next, we will exhibit a truthful, envy-free and Pareto optimal cake cutting mechanism for two agents. With a simple reduction from chore division to cake cutting, we also derive a chore division mechanism for two agents with the same set of properties. Last, we consider the more general setting of multiple agents. We assume that each agent only values a single interval of the form $[0, x_i]$. We present a truthful, envy-free and Pareto optimal cake cutting mechanism and a truthful, proportional and Pareto optimal chore division mechanism for any number of agents with valuations in this class. In the second part, we present a family of impossibility results with piecewise constant functions (refer Definition 1), even if we adopt the free disposal assumption. We again consider three assumptions: (i) *the connected piece assumption*; (ii) *position obliviousness*; (iii) *non-wastefulness*. These results partially answer the open question raised in Chen et al's study [Chen *et al.*, 2013].

Indivisible resource allocation Chapter 4 studies indivisible resource allocation problems. In the first part, we study general indivisible resource allocation problems, which we call *goods allocation* and *chores allocation* (or *goods/chores assignment*). We use noun *item* and *chore* to represent one resource in goods and chores allocation. We study several novel fairness notions such as *maximin share(MMS) guarantee*, *envy-freeness up to one item (EF1)* and *envy-freeness up to any item (EFX)*. First, we show an efficient algorithm that satisfies a variation of maximin share guarantee. Next, we show an efficient algorithm to find an EF1 and Pareto optimal allocation for two agents. Last, we show algorithms which satisfy different variations of EFX. In the second part, we introduce a real-life application, which is based on the *roommate market problem* [Chan *et al.*, 2016]. We study the case where agents arrive in an online fashion. We focus on two objectives: (1) maximizing the social welfare, which is defined as the sum of valuations that applicants have for their rooms and the happiness valuation between each pair of roommates; (2) the allocation should satisfy certain stability conditions, such that no group of people would be willing to switch roommates or rooms. We show a polynomial-time online algorithm that achieves constant competitive ratio for social welfare maximization, and then generalize it in several directions. Then we show both positive and negative results in satisfying different stability conditions.

Chapter 2

Preliminaries

A resource allocation problem contains n *agents* $\{a_1, a_2, \cdots, a_n\}$ and m *resources* $\{R_1, R_2, \cdots, R_m\}$ that we need to allocate to agents. An *allocation* $(A_1, A_2, \cdots, A_n)$ is a partition of the resources that satisfies

$$\bigcup_{1 \le i \le n} A_i = \bigcup_{1 \le j \le m} R_j,$$

where $A_i \cap A_j = \emptyset, \forall i \neq j$. A_i is the resources allocated to a_i.

For divisible resources, A_i can contain fractions of the resources, and we often assume $\cup R_j = [0, 1]$. In general, we are allowed to divide the resource into a *finite many* number of pieces before the allocation. For indivisible resources, A_i can either contain all of R_j or none of R_j for each j. For some concrete applications, the allocation may have specific constraints. For instance, if we want to metch n agents with n items, the agents may be restricted to taking exactly one item each. In some scenarios, we can throw away some resources, and do not have to allocate all of the resources to agents.

2.1 Divisible resource allocation

Under the divisibility condition, if there are two or more divisible resources, we can combine all of the resources into a single heterogeneous resource, producing a single “cake” that can be arbitrarily divided. Therefore, the divisible resource allocation problem is known as the cake cutting problem. In a cake cutting problem, there are n agents and a heterogeneous divisible cake. For simplicity, we regard the cake as an interval $[0, 1]$. As the cake can be divided arbitrarily, we need to use a density function to describe the valuation of agents. Each agent a_i has an individual density function f_i, where

$$f_i : [0, 1] \to [0, \infty).$$

In some cases, we use normalized density functions, which implies that each f_i should satisfy

$$\int_0^1 f_i(x)dx = 1.$$

With these density functions, an agent's valuation of any cake segment can be computed. Valuation of cake pieces S to a_i is $v_i(S) = \int_S f_i(x)dx$. For allocating a cake, we generally have $f_i \geq 0$, so getting an additional part does not decrease one's utility. For allocating a chore, f_i is always non-negative, and agents may want to get the smallest possible portion of the chore. We will state the class of the resource before we use.

The Robertson-Webb cake cutting model[Robertson and Webb, 1998] is commonly used when the complexity of the mechanism needs to be taken into consideration. It summarizes the ways to acquire information from agents, and it allows the following two kinds of queries:

- *EVALUATE*(i, x, y), which returns how much a_i values an interval $[x, y]$; and
- *CUT*(i, x, w), which returns the minimal y such that $v_i([x, y]) = w$ or reports that y does not exist.

Let us see how this model works if we apply the cut-and-choose protocol. First, we cut the cake at $x =$ *CUT*$(1, 0, 1/2)$. If *EVALUATE*$(2, 0, x) \geq 1/2$, let a_2 take $[0, x]$, otherwise let a_2 take $[x, 1]$. Then a_1 takes the other part. We can see the protocol requires only two queries.

In this thesis, we focus on the fairness and incentives of cake cutting mechanisms. Each valuation function is either a piecewise uniform function or a piecewise constant function defined as follows.

Definition 1. *A valuation function f is* piecewise constant *if and only if it is able to partition $[0,1]$ into finite many intervals $(I_1, I_2, \cdots, I_m)$, and for each I_i we have*

$$f(x) = f(x'), \forall x, x' \in I_i.$$

A valuation function f is piecewise uniform *if and only if f is piecewise constant and there exists a positive number c such that for all $x \in [0,1]$, we have $f(x) = 0$ or $f(x) = c$.*

Next, we introduce the definitions of fairness and truthfulness used in cake cutting models.

Definition 2. *We say an allocation $(A_1, A_2, \cdots, A_n)$ is proportional if and only if*

$$v_i(A_i) \geq 1/n, \forall i \in \{1, 2, \cdots, n\},$$

where n is the number of agents.

Similarly, we say an allocation $(A_1, A_2, \cdots, A_n)$ is envy-free if and only if

$$v_i(A_i) \geq v_i(A_j), \forall i \neq j.$$

Proportionality and envy-freeness in the chore setting are defined similarly by reversing the direction of the inequality.

A mechanism *requires agents to input the valuation functions and it outputs an allocation. A mechanism can be represented by the function*

$$\mathcal{M} : (f_1, f_2, \cdots, f_n) \to (A_1, A_2, \cdots, A_n).$$

We say a mechanism is proportional (envy-free) if it always outputs a proportional (envy-free) allocation.

A truthful mechanism ensures that truthfully reporting one's true density function is the best strategy for every agent. The formal definition is as follows.

Definition 3. *A mechanism $\mathcal{M}$ is* truthful*, if the following holds: let*

$$(A_1, A_2, \cdots, A_i, \cdots, A_n) = \mathcal{M}(v_1, v_2, \cdots, v_i, \cdots, v_n),$$

where all of the agents report their valuations truthfully, and

$$(A'_1, A'_2, \cdots, A'_i, \cdots, A'_n) = \mathcal{M}(v_1, v_2, \cdots, v'_i, \cdots, v_n),$$

where a_i manipulates the valuation function v_i to be v'_i (while other agents remain unchanged). Then we always have

$$v_i(A_i) \geq v_i(A'_i).$$

According to this definition, a mechanism is truthful if an agent cannot get more by misreporting his valuation, when all of the other agents report their true valuation functions.

Under the *free disposal assumption*, we can dispose of every part of the cake that does not have a positive value for any agent. Note that in the chore setting, there is no assumption similar to the free disposal assumption, because disposing of parts of a chore does not make sense. We must allocate the whole chore and cannot disposal of any part, even if that part has no value for anyone.

As we mentioned above, Pareto optimality is another popular property in resource allocation problems, and it is intuitively a local optimality. Here, we give the formal definition.

Definition 4. *An allocation $A = (A_1, A_2, \cdots, A_n)$ is* Pareto optimal*, if and only if for any other allocation $A' = (A'_1, A'_2, \cdots, A'_n)$, we have*

$$\exists i, v_i(A_i) < v_i(A'_i) \Rightarrow \exists j, v_j(A_j) > v_j(A'_j).$$

An equivalent statement is that no allocation A' exists such that

$$\forall i, v_i(A_i) \leq v_i(A'_i) \text{ and } \exists j, v_j(A_j) < v_j(A'_j).$$

Similarly, we define Pareto optimality for chores by reversing the direction of all the inequalities.

Note that we only consider *valid allocations*: for a divisible resource, an allocation can involve an arbitrary partition of the resource; for indivisible resources, a resource must be entirely allocated to a single agent.

Previous studies There are many useful cake cutting mechanisms. In addition to the cut-and-choose protocol, there are interesting mechanisms that can solve various problems in simple settings.

- The two-agent exact allocation problem. In this task, we need to divide the cake into pieces so that all of the agents have the same certain value for each piece. For example, we want to cut the cake into three pieces that are $0.2, 0.3$ and 0.5 of the entire cake for anyone. First, let us see how to divide the cake into two equal parts for two agents. The cut-and-choose protocol will not work in this case: the agent who implements the "choose" operation may feel two pieces are not the same. A possible solution is the Austin moving-knife protocol [Austin, 1982]: let a_1 hold two knives, k_1 and k_2, which start at $k_1 = 0$ and $k_2 = x$ such that $v_1([0, x]) = 1/2$, Then a_1 continuously moves the knifes from left to right, maintaining $v_1([k_1, k_2]) = 1/2$. When a_2 finds $v_2([k_1, k_2]) = 1/2$, he shouts "stop" and the cake can be cut at k_1 and k_2. As $v_2([k_1, k_2])$ changes smoothly and $v_2([0, x]) + v_2([x, 1]) = v_2([0, 1]) = 1$, we have $v_2([0, x]) \leq 1/2 \leq v_2([x, 1])$ or $v_2([0, x]) \geq 1/2 \geq v_2([x, 1])$. Thus, we can use the intermediate value theorem to prove there must be such a moment. As $v_1([k_1, k_2]) = v_2([k_1, k_2]) = 1/2$, then $v_1([0, 1]\backslash[k_1, k_2]) = v_2([0, 1]\backslash[k_1, k_2]) = 1/2$. Although the Austin moving-knife protocol provides a valid solution for the exact allocation among two agents, it cannot process a finite number of queries under the Robertson-Webb model. The moving-knife protocol can be generalized to obtain exact allocations in scenarios with multiple parts as follows. Suppose we want to produce pieces with proportions $(c_1, c_2, \cdots, c_m)$ for two agents. In the i-th round we solve a subproblem that divides the remaining cake into two parts, where one part is valued as exactly c_i for two agents.

- The n-agent proportional allocation problem. We use a variation of the cut-and-choose protocol to solve this problem. Let each agent report the least x that he is satisfied with proportionality if he obtains cake $[0, x]$, and then let the agent who reports the lowest x take $[0, x]$. Then it becomes a subproblem with fewer agents, and we repeat the above step until no agents are left. Proportionality is straightforward: when there are n agents and one agent quits, the remaining agents will value the allocated part as less than $1/n$, so the proportionality of the subproblem will be stronger than that of the original problem. Everyone will quit eventually, and each agent gets a proportional allocation when he quits.

- The three-agent envy-free allocation problem. One solution is to use the Austin moving-knife protocol to create three parts that are considered equivalent by the first two agents (this is the two-agent exact allocation problem) and then let the last agent pick the part he likes; the first two agents can pick any part because the parts are equivalent. However, this protocol is a continuous process and it does not work well under the Robertson-Webb model. Thus, we introduce the Selfridge-Conway discrete procedure. It was developed by John L. Selfridge about 1960, and independently by John H. Conway in 1993. It works as follows:

    1. Let the agents be Alice, Bob and Carol. First, Carol divides the cake into three pieces that he considers to have the same value.
    2. Bob trims the largest piece, to make its value equal to the value of the second largest piece (we call the piece that was cut off a *trimming*). Then, from Bob's perspective, there will be two choices for the largest piece.
    3. Let Carol, Bob and Alice, in that order, take the piece that they like the most. An additional constraint is that Bob must take the trimmed piece if Carol does not take it.
    4. Then the allocation of the entire cake will be envy-free, except for the trimming. One observation is that Alice will not envy the agent who gets the trimmed piece, even if he gets the whole trimming.
    5. (i). If Bob gets the trimmed piece, let Carol divide the trimming into three pieces so that each piece has the same value to her, then let Bob, Alice and

Carol, in that order, choose the piece that they like the most. (ii). If Carol gets the trimmed piece, then let Bob divide the trimming into three pieces so that each piece has the same value to him, then let Carol, Alice and Bob, in that order, choose the piece they want.

We can see from the above process that the Selfridge-Conway discrete procedure involves at most 5 cuts of the cake. In recent studies of fair cake cutting problems [Aziz and Mackenzie, 2016b; Aziz and Mackenzie, 2016a], they use a similar trimming idea.

2.2 Indivisible resource allocation

In indivisible resource allocation problems, we have n agents and m items (or chores) such that $I = (I_1, I_2, \cdots, I_m)$. We have a valuation function $v_i : 2^I \to [0, +\infty)$ for each a_i. In this thesis, we assume that the valuation functions are additive; therefore, $v_i(S) = \sum_{g \in S} v_i(\{g\})$, where S is a subset of I. For an item, we may use $v_i(x)$ to represent $v_i(\{x\})$.

Classic fairness notions, such as envy-freeness and proportionality, are impossible to satisfy in the worst case when resources are indivisible. They may not be satisfied even in trivial cases; for example, an allocation cannot be envy-free or proportional if we want to allocate one indivisible item to n agents. Therefore, researchers have proposed new fairness notions. The *maximin share guarantee* ([Budish, 2011]) is a fairness notions that has recently been proposed for indivisible resource allocation problems.

Definition 5. *Let $\pi_n(I)$ be the set of all n-partitions of goods set I. The n-*maximin share(n-MMS) guarantee *of a_i is computed as:*

$$MMS_i^{(n)}(I) = \max_{(S_1, S_2, \cdots, S_n) \in \pi_n(I)} \min_{j \in [n]} v_i(S_j).$$

The partition that generates the maximal result to a_i is called an n-MMS partition to a_i.

We say an allocation $A = (A_1, A_2, \cdots, A_n)$ satisfies the n-MMS guarantee (or A is an n-MMS allocation) if

$$v_i(A_i) \geq MMS_i^{(n)}(I),$$

for each a_i.

In other words, the n-MMS of a_i is the maximum value a_i can get if he partitions all of the goods into n subsets, and he can only get the set with the least value. We note that one recent study [Procaccia and Wang, 2014] has shown that the MMS guarantee cannot be satisfied in the worst case.

We cannot guarantee envy-freeness and proportionality, but we may relax the them: for example, we can assume that envy only occurs when one agent's allocation is worse than another agent's allocation even if the second agent disposes of one item. Budish defined this type of envy-freeness as *envy-freeness up to one good* [Budish, 2011]. In this thesis, we use *envy-freeness up to one item* instead of the original name, because "good" is not usually a singular proper noun in English.

Definition 6. *An allocation $A = (A_1, A_2, \cdots, A_n)$ is* envy-free up to one item (EF1 hereafter)*, if and only if for each pair of agents (i, j), we have*

$$v_i(A_i) \geq v_i(A_j \backslash \{g\}), \exists g \in A_j.$$

Similarly, an allocation is envy-free up to one chore if and only if for each (i, j), we have

$$v_i(A_i \backslash \{g\}) \leq v_i(A_j), \exists g \in A_i.$$

An agent is EF1 when he does not envy another agent, after that agent disposes of the best item in the first agent's valuation. Another way to achieve EF1 is to let the second agent drop an arbitrary item (in the worst case, it will be the least valued item). Formally, this has the following definition[Caragiannis *et al.*, 2016].

Definition 7. *An allocation $A = (A_1, A_2, \cdots, A_n)$ is* envy-free up to the least valued item (EFX hereafter)*, if and only if for each pair of agents (i, j), we have*

$$v_i(A_i) \geq v_i(A_j \backslash \{g\}), \forall g \in A_j.$$

Similarly, an allocation is envy-free up to the least valued chore if and only if for each (i, j)*, we have*

$$v_i(A_i \backslash \{g\}) \leq v_i(A_j), \forall g \in A_i.$$

From the definition, we can see that EFX is strictly stronger than EF1. Approximately satisfying EFX is another direction of further research.

Definition 8. *For goods, an allocation* $A = (A_1, A_2, \cdots, A_n)$ *is* α-EFX*, if and only if for each pair of agents* (i, j)*, we have*

$$v_i(A_i) \geq \alpha v_i(A_j \backslash \{g\}), \forall g \in A_j.$$

Similarly for chores, an allocation is α*-EFX if and only if for each* (i, j)*, we have*

$$v_i(A_i \backslash \{g\}) \leq \alpha v_i(A_j), \forall g \in A_i.$$

Envy-free graph We introduce an interesting algorithm for EFX allocation of goods, where each agent has an identical order of preference for the items. Several independent studies have found the solution to this problem using an envy-free graph. An envy-free graph is a graph which is generated by an allocation, where an edge (i, j) indicates that a_i envies a_j in the given allocation. One insight of the envy-free graph is that we can always adjust an allocation to make it acyclic by rotating the agents' allocations. If there is a cycle in the envy-free graph, let every agent take the bundle of resources that is originally allocated to the agent he envies; then, the allocation becomes strictly better. The algorithm works as follows.

1. Sort the items by the value in the non-increasing order (remember that all of the agents have the same order of preference). Maintain the envy-free graph throughout the whole process, such that there is an edge from i to j if and only if $v_i(A_i) < v_i(A_j)$.

2. Pick any agent who is not envied by anyone (in other words, any vertex in the envy-free graph with zero indegree). Let him take the next item.

3. Repeatedly eliminate cycles by rotating the in-cycle agents' allocations until there is no cycle in the envy-free graph.

4. Go back to Step 2 until all of the items are allocated.

Proof. In an acyclic graph, there is always a vertex with an indegree of 0. Therefore, we can always find i in Step 2. Below, we show that after every Step 3, the allocation will be EFX. After we add one item to the allocation, the only option that may break EFX is that some a_j considers $v_j(A_i \backslash \{g\}) > v_j(A_j)$, where A_i is the allocation to a_i and g is the most recently added item (it is also the item with the least value). However, this would contradict the fact that no one envies a_i in Step 2, which implies that $v_j(A_i \backslash \{g\}) \leq v_j(A_j)$. Therefore, EFX is always guaranteed. □

This mechanism can also be used to solve the approximate MMS allocation[Amanatidis *et al.*, 2017] in both goods and chores settings.

2.3 Online roommate allocation

In this section, we consider a specific indivisible resource allocation problem: the online resource allocation problem. In a standard roommate market problem[Chan *et al.*, 2016], we have a set of $2n$ agents $I = \{1, 2, ..., 2n\}$, a set of n rooms $R = \{r_1, r_2, ..., r_n\}$, a happiness matrix $H = \{h_{ij} \mid i, j \in I, i \neq j\}$ in which h_{ij} denotes the happiness of a_i when he is assigned to live with a_j and a valuation matrix $V = \{v_{ir} \mid i \in I, r \in R\}$ in which v_{ir} denotes the valuation of a_i to room j. We assume that all of the happiness and valuation values are nonnegative. The outcome of the roommate market is an allocation $A = \{(i, j, r)\}$, which consists of n disjoint triples. The triple (i, j, r) indicates that a_i and a_j are assigned together to room r. We require that every agent be assigned to one room and that every room be assigned to exactly two agents. The social welfare of an allocation A is defined as

$$SW(A) = \sum_{(i,j,r) \in A} (h_{ij} + h_{ji} + v_{ir} + v_{jr}).$$

We assume an online setting in which all of the agents arrive online in an uniformly random order. When a_i arrives, his valuation v_{ir} of every room r, as well as his happiness valuation h_{ij} of all of the agents a_j that have already arrived are revealed to the algorithm. The algorithm needs to assign a_i to an available room immediately. As there are exactly $2n$ agents and n rooms, we are not allowed to leave any agent unassigned. Our goal is to find an allocation A that can maximize $\mathbb{E}[SW(A)]$, such that the expectation is taken over both the randomness of the algorithm and the random arrival order of the agents.

An online algorithm is said to be *c-competitive* (or to have *competitive ratio c*), if its output allocation has an expected social welfare of no less than $c \cdot SW(A_{opt})$, where A_{opt} is the optimal offline allocation.

In addition to efficiency, several other stabilities are considered in this thesis. It is possible to satisfy stabilities due to their bidirectional property. A few stabilities which were introduced in the study by Chan et al.[Chan *et al.*, 2016].

Definition 9. *An allocation is* 2-person stable *if for each a_i and a_j in two different rooms, switching their rooms cannot make both of them strictly increase their utility.*

An allocation is 4-person stable *if for any a_i and a_j in two different rooms, switching their rooms cannot make all four people in these two rooms strictly increase their utility.*

An allocation is room stable *if for any two agents a_i, a'_i in room r_i and two agents a_j, a'_j in another room r_j, switching their rooms cannot increase the sum of the two roommates' utilities for both rooms.*

Chan et al. proved that determining whether a roommate market has a 2-person stable solution is NP-hard, and there are polynomial-time algorithms for satisfying the 4-person stability and room stability (for details, see their study [Chan *et al.*, 2016]). In an online roommate allocation problem, satisfying stability is harder than in the offline model. Therefore, we do not consider the 2-person stability in the online roommate allocation problem. Both the 4-person stability and room stability are studied in this thesis.

Chapter 3

Divisible Resource Allocation

In the previous chapter, we introduce the cut-and-choose protocol and show that it is possible to design an envy-free cake cutting mechanism when there are only two agents. Although computing envy-free allocation for two agents is not complicated, envy-free cake cutting mechanism for three agents (such as the Selfridge-Conway discrete procedure) are not straightforward already. Until recently, there were no discrete envy-free cake cutting mechanisms even for four agents. Recently, an envy-free mechanism was found with a bounded number of cuts for an arbitrary number of agents [Aziz and Mackenzie, 2016a], but it is still unknown whether there is an allocation with a polynomial number of queries. In our study, we take truthfulness into consideration, to see whether a truthful fair mechanism exists under specific conditions. In fact, the existence of a solution to the cake cutting problem with two agents is an open question, even if we restrict the valuation functions to piecewise constant functions.

3.1 Truthful fair mechanism without the free disposal assumption

The search for a truthful and fair mechanism without the free disposal assumption for cake cutting problem is relatively new. The first study of *truthful and fair* cake cutting was done by Chen et al. [Chen *et al.*, 2013]. One of their most significant contributions is a truthful envy-free mechanism for an arbitrary number of agents with *piecewise uniform valuation functions*, and the mechanism additively needs *the free disposal assumption*. Although certain resources, such as cake or machine processing time, may be easy to get rid of, this is not the case for all resources. For instance, when we divide a piece of land among antagonistic agents or countries, we cannot simply throw away part of the land, and any piece of unallocated land constitutes a potential subject of future disputes. The free disposal assumption is even less reasonable when it comes to chore division—indeed, under this assumption, we might as well simply dispose of the whole chore!

With this motivation in mind, in the first section of this chapter we consider the problem of fairly and truthfully dividing heterogeneous resources without the free disposal assumption. Not having the option of throwing away part of the resource makes the task more complicated, as even if the mechanism is only allowed to throw away parts that are not valued by any agent, this still prevents agents from gaining by not reporting parts of the resource that no other agent values, in the hope of getting those parts for free along with a larger share of the remaining parts. Thus, as Chen et al. noted, getting rid of the free disposal assumption adds "significant complexity" to the problem, as under this condition the mechanism has to specify exactly how to allocate parts that no agent desires. The same group of authors also gave an example illustrating how removing the assumption can be problematic, even in the special case of two agents with very simple valuations.

3.1.1 Related works

For easier analysis, many recent studies have restricted the valuation functions to piecewise constant and piecewise uniform functions, which simplifies the description of valuations e.g. [Bei *et al.*, 2012], [Chen *et al.*, 2013], [Aziz and Ye, 2014], [Bei *et al.*, 2017a], [Menon and Larson, 2017]. One study has shown that in the Robertson-Webb model, finding an envy-free allocation with piecewise uniform valuation functions is equivalent to solving the problem without this property [Kurokawa *et al.*, 2013].

Chen et al. first studied truthful envy-free mechanisms with restricted valuations [Chen *et al.*, 2013]. They proposed a truthful and envy-free mechanism with piecewise uniform valuation function under the free disposal assumption. Maya and Nisan developed a truthful, proportional and Pareto optimal mechanism for two agents with piecewise uniform valuations [Maya and Nisan, 2012]. With piecewise constant valuations, Kurokawa et al. showed there is no truthful and envy-free mechanism in the Robertson-Webb model with bounded queries [Kurokawa *et al.*, 2013], and Aziz and Ye showed there is no truthful, proportional and Pareto optimal mechanism [Aziz and Ye, 2014]. In a setting in which every agent values only one interval, Alijani et al. showed efficient truthful envy-free mechanisms [Alijani *et al.*, 2017].

Most previous studies have adopted the free disposal assumption. In this section, we study the cake cutting problem without the free disposal assumption. This section is reproduced from our published paper [Bei *et al.*, 2018].

3.1.2 Anonymous mechanism

One might consider that "fair" mechanisms should treat the agents equally regardless of their identity.

Definition 10. *A mechanism $\mathcal{M}$ is* anonymous*, if the following holds:*

for any density functions $(f_1, f_2, \cdots, f_n)$ and any permutation σ of $(1, 2, \cdots, n)$, if

$$(A_1, A_2, \cdots, A_n) = \mathcal{M}(f_1, f_2, \cdots, f_n)$$

and

$$(A'_1, A'_2, \cdots, A'_n) = \mathcal{M}(f_{\sigma(1)}, f_{\sigma(2)}, \cdots, f_{\sigma(n)}),$$

then

$$v_i(A_i) = v_i(A'_{\sigma(i)})$$

for every i.

However, the following result shows that anonymity is incompatible with truthfulness and envy-freeness.

Theorem 1. *When the valuation functions are piecewise uniform, there is no truthful, envy-free, and anonymous cake cutting mechanism even for only two agents without the free disposal assumption.*

Proof. We are going to prove a stronger statement: even we restrict the valued part of valuations to be $[0, x_i]$, there is no truthful envy-free anonymous mechanism. We use $W_i = [0, x]$ to denote that a_i has value on $[0, x]$, and has no value on the rest part.

Suppose that such a mechanism exists. Let $x \in [0, 1)$ and $W_1 = W_2 = [0, x]$. Assume without loss of generality that in this instance, a_1 gets an interval containing point x and ending at point $x + f(x) > x$, possibly among other intervals. By envy-freeness, both agents must get half of the interval $[0, x]$.

If $W_1 = [0, x + \varepsilon]$ for some $\varepsilon \in [0, f(x)]$ and $W_2 = [0, x]$, then a_1 must get the entire interval $[x, x + \varepsilon]$ and half of the interval $[0, x]$. This is because a_2 must get at least half of the interval $[0, x]$, and if a_1 gets less than the whole interval $[x, x + \varepsilon]$, he can manipulate by reporting $W_1 = [0, x]$ and getting the whole interval $[x, x + \varepsilon]$. By anonymity, if $W_1 = [0, x]$ and $W_2 = [0, x + \varepsilon]$ for some $\varepsilon \in [0, f(x)]$, a_2 must also get the whole interval $[x, x + \varepsilon]$ and half of the interval $[0, x]$.

Now suppose that $W_1 = W_2 = [0, x+\varepsilon]$ for some $\varepsilon \in [0, f(x)]$. Both agents must get half of the interval $[0, x + \varepsilon]$. If a_1 gets more than half of the interval $[x, x + \varepsilon]$, then a_2 gets more than half of the interval $[0, x]$. In this case, if $W_2 = [0, x]$, a_2 can manipulate by reporting $W_2 = [0, x + \varepsilon]$. So a_1 cannot get more than half of the interval $[x, x + \varepsilon]$. By symmetry, neither can a_2. This means that both agents get exactly half of the interval $[x, x + \varepsilon]$. In other words, for any $y \in [x, x + f(x)]$, if $W_1 = W_2 = [0, y]$, then both agents get exactly half of the interval $[x, y]$.

Next, consider the set

$$A := \{(x, y) \in \mathbb{R}_{[0,1)} \times \mathbb{Q}_{[0,1]} \mid x < y < x + f(x)\}.$$

This set is uncountable, since for each of the uncountably many x's, there is at least one y such that $(x, y) \in A$. If for each y there only exist a finite number of x's such that $(x, y) \in A$, this set would be countable, which we know is not the case. Hence there exists a y such that $(x, y) \in A$ for infinitely many x's. Fix such a y.

Finally, suppose that $W_1 = W_2 = [0, y]$. For any of the infinitely many x's such that $(x, y) \in A$, both agents must receive exactly half of the interval $[x, y]$. However, if the mechanism divides the interval $[0, y]$ into k intervals in the allocation, then there can be at most one value of x per interval, and therefore at most k values in total, with this property. This gives us the desired contradiction that we have to divide the cake into infinite many intervals. □

In the rest of this thesis, the class of valuations that form $[0, x]$ appear many times. We say a valuation is *prefix valuation*, if the agent has the same positive value everywhere on $[0, x]$ for some x, and has no value on $[x, 1]$.

3.1.3 Connected piece assumption

When cutting a real cake, we intuitively know that we do not want to cut the cake into too many pieces. For example, if we are allocating a party cake among n guests, we try to allocate each guest one piece of cake rather than many small pieces of cake. Similarly, for some resources, such as time and land, it is better to have single piece, rather than lots of pieces. In a standard cake cutting model, we define the connected piece assumption as follows.

Definition 11. *An allocation satisfies* the connected piece assumption *if every agent gets a single subinterval of the cake. In other words, the cake will have* $n - 1$ *cutting points, and every agent will get one of these* n *intervals. A mechanism produces connected pieces if it always outputs allocations that satisfy the connected piece assumption.*

With the connected piece assumption, we prove there is no truthful envy-free cake cutting mechanism with piecewise uniform valuation functions for two agents. Similar results of impossibility were presented in previous studies [Menon and Larson, 2017; Bei *et al.*, 2017a], which depend on the free disposal assumption.

Theorem 2. *When the valuation functions are piecewise uniform, there does not exist a truthful and envy-free cake cutting mechanism for two agents without the free disposal assumption that satisfies the connected piece assumption, even the valuations are restricted to be prefix valuations.*

Proof. Suppose that such a mechanism exists. First, consider the instance where $W_1 = W_2 = [0, x]$ for some $x \in (0, 1)$. One agent will get the interval $[0, x/2]$ and the other agent the interval $[x/2, 1]$; assume without loss of generality that a_1 gets $[0, x/2]$ and a_2 gets $[x/2, 1]$. Next, consider the instance where $W_1 = [0, x]$ and $W_2 = [0, y]$ for some $y \in (x, 1)$. Then a_2 must still get the interval $[x/2, 1]$; otherwise he can report $W_2 = [0, x]$ instead.

Now, consider the instance where $W_1 = W_2 = [0, y]$. As before, one agent will get the interval $[0, y/2]$ and the other agent the interval $[y/2, 1]$. If a_1 gets $[0, y/2]$, then in the previous instance a_1 can gain more by reporting $W_1 = [0, y]$. Hence it must be that a_2 gets $[0, y/2]$ and a_1 gets $[y/2, 1]$. This means that in the instance where both agents report $[0, y]$, the ordering of the allocated pieces is reversed from the allocation in the instance where both agents report $[0, x]$. Since this holds for any $y > x$, if we take some $z > y$ (obviously $z > x$ also), we find that no allocation works when both agents report $[0, z]$, a contradiction. □

A remark is that we will have a mechanism under the free disposal assumption. If we have $W_1 = [0, x_1]$ and $W_2 = [0, x_2]$ and $x_1 < x_2$, let a_1 get $[0, x_2/2]$ and a_2 get $[x_2/2, x_2]$. There are four cases:

- a_1 may manipulate $x'_1 > x_2$, then he can get $x_2 - x'_1/2 < x_2/2$;
- a_1 may manipulate $x'_1 \leq x_2$, then nothing changes;
- a_2 may manipulate $x'_2 < x_2$, then he can get $x'_2/2 < x_2/2$;
- a_2 may manipulate $x'_2 > x_2$, then he can get $x_2 - x'_2/2 < x_2/2$.

We conclude that the mechanism is truthful and envy-free by checking these four cases.

3.1.4 Position oblivious mechanism

Position obliviousness is another mild assumption, which seems like the anonymity on cake segments. Let us see the formal definition first.

Definition 12. *Given a vector of n piecewise constant density functions* $\mathbf{f} = (f_1, f_2, \dots, f_n)$*, we define the indicator function* $L_{\mathbf{f}} : \mathbb{R}^n \mapsto \mathbb{R}$*, where*

$$L_{\mathbf{f}}(r_1, \dots, r_n) = |\{x \mid \forall i, f_i(x) = r_i\}|,$$

where $|S|$ means the summation of length of all intervals in the set S. Similarly, assume $A = (A_1, \dots, A_n)$ is an allocation produced by some mechanism with these density functions, define $L_{\mathbf{f}}^{A_i} : \mathbb{R}^n \mapsto \mathbb{R}$*, where*

$$L_{\mathbf{f}}^{A_i}(r_1, \dots, r_n) = |\{x \mid \forall i, f_i(x) = r_i\} \cap A_i|.$$

Definition 13. *A mechanism $\mathcal{M}$ is* position oblivious *if for any two cake cutting instances with density functions $\mathbf{f} = (f_1, f_2, ..., f_n)$ and $\mathbf{f}' = (f'_1, f'_2, ..., f'_n)$ such that $L_{\mathbf{f}} \equiv L_{\mathbf{f}'}$, the mechanism always outputs two allocations $\mathcal{M}(\mathbf{f}) = (A_1, \dots, A_n)$ and $\mathcal{M}(\mathbf{f}') = (A'_1, \dots, A'_n)$ such that $L_{\mathbf{f}}^{A_i} \equiv L_{\mathbf{f}'}^{A'_i}$ for every i.*

Intuitively, a position oblivious mechanism will decide the allocation only based on the set of segments $\{X_t \mid t = 1, \dots, m\}$ and each agent's values on these segments, but not on these segments' relative positions on $[0, 1]$.

Because in this section, we are considering piecewise uniform functions, we may use another simpler notation $L_f(S)$ to represent the length of segments where agents in S have value on it, while other agents don't have value on it. For example, $L_f(1, 0, 1)$ should be $L_f(\{1, 3\})$ in the new notation.

Theorem 3. *When the valuation functions are piecewise uniform, there does not exist a truthful, proportional, and position oblivious cake cutting mechanism for n agents without the free disposal assumption, where $n = 2k$ for some positive integer k.*

Proof. Suppose that such a mechanism exists. Assume that the cake is represented by the interval $[0, 4k^2 + k]$.

First, consider the instance where $W_{2i-1} = W_{2i} = [i-1, i]$ for $i = 1, 2, \ldots, k$. Since the interval $[k, 4k^2 + k]$ is of length $4k^2$ and there are $2k$ agents, some agent gets value more than $2k - 1$ from the interval. Assume without loss of generality that a_1 is one such agent, and that a_1 gets the interval $[k, 3k-1]$. Since the mechanism is proportional, a_1 must get value at least $1/2k$ from the interval $[0, 1]$ as well.

Next, consider the instance where $W_1 = [0, 1] \cup [k, 3k-1]$, $W_2 = [0, 1]$, and $W_{2i-1} = W_{2i} = [i-1, i]$ for $i = 2, 3, \ldots, k$. Agent a_1 must still get value at least $1/2k$ from the interval $[0, 1]$; otherwise he can report $W_1 = [0, 1]$ instead. This means that a_2 gets a total value of at most $1 - 1/2k$ in this instance.

Finally, consider the instance where $W_1 = W_2 = [0, 1] \cup [k, 3k-1]$ and $W_{2i-1} = W_{2i} = [i-1, i]$ for $i = 2, 3, \ldots, k$. By proportionality, a_2 must receive value at least 1; let $B_2 \subseteq [0, 1] \cup [k, 3k-1]$ be a piece of length 1 that a_2 receives. If $W_2 = B_2$ while the other W_i's remain fixed, then since the mechanism is position oblivious, a_2 must get a total value of at most $1 - 1/2k$. However, in that case a_2 can report $W_2 = [0, 1] \cup [k, 3k-1]$ and receive value 1. This implies that the mechanism is not truthful and yields the desired contradiction. $\square$

As with the connected piece assumption, Bei et al. showed a similar negative result for position obliviousness *under* the free disposal assumption but using the larger class of piecewise constant valuations[Bei *et al.*, 2017a]. For piecewise uniform valuations, Chen et al.'s mechanism is truthful, envy-free, and position oblivious under the free disposal assumption.

3.1.5 Pareto optimal mechanism for two agents

In this section, we focus on the case of two agents. We show that in this case, there exists a truthful, envy-free, and Pareto optimal mechanism for both cake cutting and chore division, for two agents with arbitrary piecewise uniform valuations.

We first describe the cake cutting mechanism.

Mechanism 1 (for cake cutting between two agents)

Step 1: Find the smallest value of $x \in [0,1]$ such that $v_1([0,x]) = v_2([x,1])$.

Step 2: Assign to a_1 the intervals in $[0,x]$ valued by a_1 and the intervals in $[x,1]$ *not* valued by a_2, and assign the rest of the cake to a_2.

While this is a succinct description of the mechanism, it turns out that the description is somewhat difficult to work with. We next provide an alternative formulation that is more intuitive and will help us in establishing the claimed properties of the mechanism.

Mechanism 1 (alternative formulation)

Phase 1: Let a_1 start at point 0 of the cake moving to the right and a_2 start at point 1 of the cake moving to the left. Let both agents "eat" the cake with the same constant speed, jumping over any interval for which they have no value according to their reported valuations. If the agents are at the same point while both are still eating, go to Phase 3. Else, one of the agents has no more valued interval to eat; go to Phase 2.

Phase 2: Assume that a_i is the agent who has no more valued interval to eat. Let a_i stop and a_{3-i} continue eating. If the agents are at the same point (either while a_{3-i} eats or while a_{3-i} jumps over an interval of zero value), go to Phase 3. Else, both agents have stopped, but there is still unallocated cake between their current points. In this case, let a_{3-i} continue eating the unallocated cake until he is at the same point as a_i, and go to Phase 3.

Phase 3: Assume that both agents are at point x of the cake. (It is possible that the two agents meet while both of them are jumping. In this case, we let a_2 jump first.) Assign any unallocated interval to the left of x to a_2 and any unallocated interval to the right of x to a_1.

Theorem 4. *When the valuation functions are piecewise uniform, Mechanism 1 is a truthful, envy-free, and Pareto optimal cake cutting mechanism for two agents without the free disposal assumption.*

Proof. We begin with truthfulness. Note that there is no incentive for an agent to report an interval that he has no value since this can only result in the agent wasting time eating such intervals. So the only potential deviation is for the agent to report a strict subset of the intervals that he has a positive value. If the agent does not report intervals that he has a positive value, then the intervals that he jumps over before the agents meet will be lost to the other agent, and the agent can use the extra time gained from not reporting these intervals to eat intervals of no more than the same length.

Next, for envy-freeness, it suffices to show that each agent gets at least half of his valued intervals allocated in each phase. In Phase 1, each agent only gains intervals that he has a positive value, and loses intervals that he has a positive value (due to the other agent's eating) at no more than the same speed. In Phase 2, the agent who continues eating can only gain more, while the agent who has stopped eating has no more interval that he has a positive value. In Phase 3, a_1 has no unallocated interval to the left of x that he has a positive value, so he cannot lose any unallocated interval that he has a positive value. The same argument holds for a_2.

Finally, our mechanism allocates any interval valued by at least one agent to an agent who has a positive value on it. This establishes Pareto optimality. □

Mechanism 1 gives rise to a dual mechanism for two-agent chore division that satisfies the same set of properties.

Mechanism 2 (for chore division between two agents)

Step 1: Use Mechanism 1 to find an initial allocation of the chore, treating the chore valuations as cake valuations.

Step 2: Swap the pieces of the two agents in the allocation from Step 1.

Theorem 5. *When the valuation functions are piecewise uniform, Mechanism 2 is a truthful, envy-free, and Pareto optimal chore division mechanism for two agents without the free disposal assumption.*

Proof. First, truthfulness holds because minimizing the chore in the swapped allocation is equivalent to maximizing the chore in the initial allocation, and Theorem 4 shows that this is exactly what Mechanism 1 incentivizes the agents to do. Next, envy-freeness holds again by Theorem 4 because getting at most half of the chore in the swapped allocation is equivalent to getting at least half of the chore in the initial allocation. Finally, in the initial allocation any interval of the chore valued by only one agent is allocated to that agent, so in the swapped allocation the interval is allocated to the other agent, implying that the mechanism is Pareto optimal. □

Besides truthfulness, envy-freeness, and Pareto optimality, how do Mechanisms 1 and 2 fare with respect to the other previous properties

- Mechanism 1 is not *anonymous*: If $W_1 = [0, 0.5]$ and $W_2 = [0, 1]$ then both agents get value 0.5, while if $W_1 = [0, 1]$ and $W_2 = [0, 0.5]$ then a_1 gets value 0.75 and a_2 gets value 0.25.
- It is also not *position oblivious*: If $W_1 = [0, 0.5]$ and $W_2 = [0, 1]$ then both agents get value 0.5, while if $W_1 = [0.5, 1]$ and $W_2 = [0, 1]$ then a_1 gets value 0.25 and a_2 gets value 0.75.
- The allocation when $W_1 = [0, 1]$ and $W_2 = [0, 0.5]$ shows that the mechanism does not satisfy the *the connected piece assumption*.

The same examples demonstrate that Mechanism 2 likewise satisfies none of the three properties.

3.1.6 Pareto optimal mechanism for n agents

In this section, we consider the general setting where we allocate the resource among any number of agents. We assume that each agent i has a prefix valuation (remind it implies that a_i only values the interval $[0, x_i]$). Prefix valuations may appear in a scenario where the agents are dividing machine processing time: a_i has a deadline x_i for his jobs, so he would like to maximize the processing time he gets before x_i but has no value for any processing time after x_i. We also remark that the example used to illustrate that removing the free disposal assumption can be problematic consists of *two* agents whose valuations belong to this class [Chen *et al.*, 2013, p. 296]. Hence, designing a fair and truthful algorithm is by no means an easy problem even for this valuation class.

We first describe the cake cutting mechanism.

Mechanism 3 (for cake cutting among n agents)

Step 1: If there is one agent left, the agent gets the entire remaining cake. Else, assume that there are $k \geq 2$ agents and length l of the cake left. Find the maximum $x \in [0, l]$ such that agent i values the entire interval $[(i-1)x, ix]$ for all $i = 1, 2, \ldots, k$, and allocate the interval $[(i-1)x, ix]$ to agent i.

Step 2: The agent whose right endpoint of his allocated interval coincides with the right endpoint of his valued piece exits the process. (If there are more than one such agent, choose the one with the lowest number.)

Step 3: Renumber the remaining agents in the same order starting from 1, and relabel the left endpoint of the remaining cake as point 0. Return to Step 1.

Theorem 6. *Let n be any positive integer. When the valuation functions are piecewise uniform, Mechanism 3 is a truthful, envy-free, and Pareto optimal cake cutting mechanism for n agents without the free disposal assumption, if valuations are restricted to be prefix valuations.*

Proof. First, for truthfulness, there are two types of manipulation: moving x_i to the left and to the right. Moving x_i to the left can only cause a_i to quit the process early when he could have gained more by staying on. On the other hand, if moving x_i to the right causes the allocation to change in some round of Step 1, the agent can only get less value from the allocated interval as its right endpoint moves past x_i. Moreover, since he has no more valued intervals to the right, he cannot make up for the loss.

Next, for envy-freeness, if an agent is no longer in the process, he has no more part that he has a positive value on it. During the process, all remaining agents receive an interval of the same length in each round. Since each agent values the entire interval that he receives, he does not envy any other agent.

Finally, our mechanism allocates any interval to an agent who has a positive value on it (if at least one agents have a positive value on it). This establishes Pareto optimality.

□

In the chore setting, prefix valuations may also appear in the case where agent i has another task and will be busy before the deadline x_i, therefore he wants to minimize the chore that he takes before x_i, and after x_i everything will be easy to deal with (so it valued as 0). Unlike in the case of two agents, there is no simple reduction between cake cutting and chore division in the general case. Nevertheless, our next result shows a truthful and proportional chore division mechanism for any number of agents. We were not able to strengthen the proportionality guarantee to envy-freeness and leave it as an interesting open question for future research.

Mechanism 4 (for chore division among n agents)

Step 1: Let a_1 take the piece $[0, x_1/n] \cup [x_1, 1]$. If some other agent has no value on parts of the interval $[0, x_1/n]$, give those parts to the agent. (If there are several such agents, allocate the parts arbitrarily.)

Step 2: Repeat Step 1 with the next agent up to a_{n-1} and the remaining chore; a_i takes the leftmost interval with value x_i/n as well as any piece for which he has no value. (If a_i has value less than x_i/n left, he takes the entire remaining chore.)

Step 3: Agent a_n takes all of the remaining chore.

Theorem 7. *Let n be any positive integer. When the valuation functions are piece-wise uniform, Mechanism 4 is a truthful, proportional, and Pareto optimal chore division mechanism for n agents without the free disposal assumption, if valuations are restricted to be prefix valuations.*

Proof. We begin with truthfulness. First, an agent who has no value on some piece that the mechanism initially allocates to another agent has no incentive not to take the piece. Apart from this, agent a_n has no control over his allocation, so the mechanism is truthful for his. For any other agent, there are two types of manipulation: moving x_i to the left and to the right. Moving x_i to the right can only increase the value of the piece that a_i has to take. If a_i moves x_i to the left by an amount y, he can save a value of at most y/n but has to take a piece of value y at the end. So a_i does not have a profitable manipulation.

We now consider proportionality. Each agent up to a_{n-1} gets a piece of value at most x_i/n. For a_n, we consider two cases. Let $x = \min(x_1, x_2, \ldots, x_{n-1})$. If $x_n \leq x$, then each of the first $n-1$ agents takes at least $1/n$ of the interval $[0, x_n]$, so at most $1/n$ of this interval is left for a_n. Else, we have $x_n > x$. The intervals $[0, (n-1)x/n]$ and $[x, 1]$ will not be left to a_n, meaning that a_n receives value at most $x/n < x_n/n$.

Finally, our mechanism allocates any interval for which some agent has no value to one such agent. This establishes Pareto optimality.

□

3.2 Cake cutting with piecewise constant valuation functions

The piecewise constant function is known as the step function in real analysis. It can be used to approximately approach a continuous function for any given accuracy, which provides a way to improve the analysis of the functions. In this section, we follow Chen et al. [Chen *et al.*, 2013] and study fair division with piecewise constant valuation functions instead of piecewise uniform valuation functions.

3.2.1 Related works

There is no truthful, proportional and Pareto optimal mechanisms for piecewise constant piecewise constant valuations [Schummer, 1996; Aziz and Ye, 2014]. Truthful mechanisms for two agents have been characterized [Maya and Nisan, 2012]. A recent work studied non-wastefulness and the connected piece assumption [Menon and Larson, 2017], which are also studied in our paper [Bei *et al.*, 2017a]. Menon and Larson developed an approximation algorithm that is proportional and partially satisfies truthfulness with piecewise uniform valuations. We have already introduced some of the related studies in the previous section, as these two sections significantly overlap.

This section is reproduced from our published paper [Bei *et al.*, 2017a].

3.2.2 Connected piece assumption

In Section 3.1.3, we learn there is no truthful envy-free mechanism that satisfyies the connected piece assumption for two agents without the free disposal assumption when the valuations are piecewise uniform. In this subsection, we have two differences from the setting in Section 3.1.3: the valuations are piecewise constant and we adopt the free disposal assumption. We are going to prove there is no truthful envy-free mechanism that can satisfy the connected piece assumption for any number of agents.

Theorem 8. *Under the free disposal assumption, no truthful envy-free cake cutting mechanism satisfies the connected piece assumption for any number of agents when valuation functions are piecewise constant.*

Proof. Let $\varepsilon > 0$ be a sufficiently small constant. The cake is represented by $[0, 7 + n]$. The density functions are defined as follows.

$$\begin{aligned} f_1(x) &= 1, && \text{for } x \in [1, 2+\varepsilon] \cup [5, 6] \\ f_2(x) &= 1, && \text{for } x \in [3, 4] \cup [7-\varepsilon, 8] \\ \text{For } i = 3, \ldots, n: & && \\ f_i(x) &= 1, && \text{for } x \in [6+i-\varepsilon, 6+i+\varepsilon] \end{aligned}$$

The valuation of all density functions is zero on the unspecified intervals. Notice that the cake is $[0, 9]$ when $n = 2$, in which case only f_1 and f_2 are defined.

Under the connected piece constraint, it is easy to see that either a_1 or a_2 will get a value of at most $1 + \varepsilon$. Without loss of generality, assume it is a_1. Considering the scenario where a_1 misreports his function to be

$$f_1'(x) = \begin{cases} 1 & x \in [1, 2+\varepsilon] \cup [5, 6] \\ 2 & x \in [8, 7+n] \end{cases},$$

we next show that a_1 can get an allocation with a value of at least 2 (with respect to f_1).

Given the condition that one has to receive a consecutive piece, we know that a_1 cannot get a value of more than 2 from the interval $[8, 7 + n]$. For each $x \in [8, 7 + n - 1]$, the interval $[x, x+1]$ will cover at least half of some $[6+i-\varepsilon, 6+i+\varepsilon]$, thus the envy-freeness must be broken. It holds trivially for $n = 2$, as a_1 has value exactly 2 on $[8, 7+n] = [8, 9]$. However, receiving value 2 is not enough for proportionality (as $\int_{[0,7+n]} f_1'(x)\, dx = 2n + \varepsilon$), and thus, it is not enough for envy-freeness. Also, it can be seen that a_1 cannot get a superset of $[7 - \varepsilon, 8]$ on which a_2 has more than half of his total valuation. Therefore, to receive a value of more than 2, a_1 has to take almost the entire interval $[1, 6]$; this results in a value of more than 2 with respect to the true function f_1. Compared with the upper bound $1 + \varepsilon$ that a_1 receives when reporting f_1 truthfully, his obtained value is increased from manipulation.

Therefore, no truthful mechanism exists, and the theorem follows. $\square$

3.2.3 Non-wasteful mechanism

The non-wastefulness implies that we are not willing to allocate a piece to someone who does not like it at all. Formally, we have the following definition.

Definition 14. *A mechanism is* non-wasteful *if and only if an agent will never get a cake segment, where he has no value on this segment.*

With this mild assumption, we have the following negative conclusion.

Theorem 9. *Under the free disposal assumption, there is no truthful, envy-free, and non-wasteful cake cutting mechanism when valuation functions are piecewise constant.*

Proof. Suppose otherwise there is a non-wasteful truthful envy-free mechanism $\mathcal{M}$. Consider the cake cutting instance with two agents whose density functions are $f_1(x) = 1$ and $f_2(x) = 1$ on the whole cake. The allocation $A = (A_1, A_2)$ given by $\mathcal{M}$ must satisfy $|A_1| = |A_2| = 0.5$. Consider another cake cutting instance with two agents whose valuation density functions are

$$g_1(x) = \begin{cases} 1 & x \in A_1 \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad g_2(x) = 1.$$

For this instance (g_1, g_2), a_1 will get the whole A_1 if $\mathcal{M}$ is truthful, as otherwise a_1 can bid $g_1'(x) = 1$ and get A_1. Moreover, a_1 cannot get more than A_1, because otherwise a_2 will envy a_1. Thus, $A = (A_1, A_2)$ is the only possible allocation generated by $\mathcal{M}$ for the instance (g_1, g_2). However, by taking advantage of the non-wasteful condition, a_2 can misreport his density function and get a better allocation than A_2. For example, a_2 can bid the following function g_2':

$$g_2'(x) = \begin{cases} 1 & x \in A_1 \\ 0.5 & \text{otherwise} \end{cases} .$$

For the instance (g_1, g_2'), a_2 will receive the entire A_2 according to the non-wasteful condition. In addition, he will receive some of A_1 to guarantee envy-freeness. Thus, a_2 will receive a strictly larger value from manipulation, which implies that $\mathcal{M}$ cannot be truthful. □

3.2.4 Position oblivious mechanism

For piecewise constant valuation functions, we cannot use a set of agents to uniquely denote the type of subintervals as we did with piecewise uniform valuation functions. Instead of that, we need to use n real numbers to denote the density of valuation to every agent. Below we show that position oblivious, truthful and envy-free mechanism does not exist.

First, we consider a special piecewise uniform case.

Lemma 1. *For any $I_1, I_2 \subseteq [0,1]$ such that $I_1 \cap I_2 = \emptyset$ and $|I_1| = |I_2|$, given two density functions*

$$f_1(x) = \begin{cases} 1 & \textit{if } x \in I_1 \cup I_2 \\ 0 & \textit{otherwise} \end{cases} \quad \textit{and } f_2(x) = \begin{cases} 1 & \textit{if } x \in I_2 \\ 0 & \textit{otherwise} \end{cases},$$

any mechanism $\mathcal{M}$ that is truthful, envy-free and position oblivious would produce an allocation $\mathcal{M}(f_1, f_2) = (A_1, A_2)$ such that $I_2 \subseteq A_2$.

Proof. First consider the case where both agents have density function f_1. By the envy-free condition, both agents will get half of $I_1 \cup I_2$. That is, we have $\mathcal{M}(f_1, f_1) = (A_1, A_2)$ with $|A_1 \cap (I_1 \cup I_2)| = |A_2 \cap (I_1 \cup I_2)| = |I_1| = |I_2|$.

Next, consider another case where a_2 has a density function

$$g_2(x) = \begin{cases} 1 & \text{if } x \in A_2 \cap (I_1 \cup I_2) \\ 0 & \text{otherwise} \end{cases}$$

Because $\mathcal{M}$ is truthful, one must have $\mathcal{M}(f_1, g_2) = (A'_1, A'_2)$ with $A_2 \subseteq A'_2$, since otherwise a_2 can bid $g_2 = f_1$ to receive whole A_2.

Finally, due to the position oblivious property and the fact that $L_{(f_1,f_2)} = L_{(f_1,g_2)}$, we know with input (f_1, f_2) mechanism should also give the whole I_2 to a_2. This proves the lemma. $\square$

Theorem 10. *Under the free disposal assumption, there is no truthful, envy-free, and position oblivious mechanism even for two agents when valuation functions are piecewise constant.*

Proof. Assume by contradiction that such mechanism $\mathcal{M}$ exists. Consider the cake cutting instance (f_1, f_2) with

$$f_1(x) = \begin{cases} 1 & \text{if } x \in [0, \frac{1}{3}] \\ 0 & \text{otherwise} \end{cases} \text{ and } f_2(x) = \begin{cases} 1 & \text{if } x \in [0, \frac{1}{3}] \\ \varepsilon & \text{if } x \in (\frac{1}{3}, 1] \end{cases}$$

with some small $\varepsilon > 0$.

Assume that with these inputs $\mathcal{M}$ produces the allocation $A = (A_1, A_2)$. By the envy-free condition of a_1, we know $|A_1 \cap [0, \frac{1}{3}]| \geq \frac{1}{6}$, which implies $|A_2 \cap [0, \frac{1}{3}]| \leq \frac{1}{6}$, and by the envy-freeness of a_2 we have $\frac{1}{6} \geq \left|A_2 \cap \left[0, \frac{1}{3}\right]\right| \geq \frac{1}{6}(1 - \frac{4}{3}\varepsilon))$, and $\left|A_2 \cap \left(\frac{1}{3}, 1\right]\right| \geq \frac{1}{3}$.

Next, consider another cake cutting instance (g_1, g_2) with

$$g_1 = f_1 \text{ and } g_2(x) = \begin{cases} 1 & \text{if } x \in [0, \frac{1}{3}] \cup I \\ 0 & \text{otherwise} \end{cases},$$

where we have picked $I \subseteq A_2 \cap (1/3, 1]$ such that $|I| = \frac{1}{3}$.

By Lemma 1, when reporting the true valuations, $\mathcal{M}$ would produce the allocation $\mathcal{M}(g_1, g_2) = (A_1, A_2)$ with $[0, \frac{1}{3}] \subseteq A_1$. Thus a_2's utility will be no more than $|I| = \frac{1}{3}$. On the other hand, by reporting his density function as $g_2 = f_2$, a_2 will receive the whole A_2 (A_2 is the allocation to a_2 with instance (f_1, f_2)), and his utility will be at least $\frac{1}{6}(1 - \frac{4}{3}\varepsilon) + \frac{1}{3} = \frac{1}{2} - \frac{2}{9}\varepsilon$. With ε small enough, this value is strictly larger than $\frac{1}{3}$, which implies that $\mathcal{M}$ cannot be truthful. $\square$

3.3 Discussion

There are many possible directions for future study. We could try to approximately satisfy the requirements for a truthful and envy-free mechanism. The approximation could be defined as follows.

Definition 15. *An allocation A is* α-envy-free *if for each pair of a_i and a_j, we have $\alpha v_i(A_i) \geq v_i(A_j)$. Similarly, a mechanism is* β-truthful *if a manipulation does not increase one's utility to β times. When $\alpha = \beta = 1$, the definition is exactly envy-free and truthful.*

Bei et al. provided the proof for a mechanism maximizing the Nash social welfare function that is 2-truthful and envy-free [Bei *et al.*, 2017a]. We plot (α, β) to show whether a mechanism can guarantee α-truthful and β-envy-free. Currently, we know that three points exist, namely, $(+\infty, 1)$ and$(2, 1)$ and $(1, +\infty)$, where the $+\infty$-truthful and envy-free mechanism is exactly an envy-free mechanism, and the truthful and $+\infty$-envy-free mechanism can be designed by allocating the whole cake to any fixed agent.

Another recent study [Bei *et al.*, 2017b] examined the *networked fairness*. In real-life scenarios, one agent envies another only if he knows that agent. Networked envy-freeness does not reduce the difficulty of designing mechanism, as a fully connected graph is the same as the general cake cutting problem. The study provided solutions for special cases and left the existence of mechanisms for networks that are directed circles or undirected circles as an open question. We successfully find a networked envy-free mechanism for the directed circle case.

Fact 1. *There is a mechanism that can always output an envy-free allocation on a directed circle network.*

Proof. Let the first agent divide the cake evenly into n pieces, and let the other agents each pick the preferred piece one by one in the circular order (every agent knows the next agent). After $n-1$ agents are done, let the first agent pick the remaining piece. We can see that every agent except the first agent will choose the piece before the agent that he might envy. The first agent will not envy any of the others, as the pieces are all the same in his view. $\square$

Chapter 4

Indivisible Resource Allocation

In the first part of this chapter, we study the general indivisible resource allocation problem. Several reasonable fairness notions are proposed for the cases of indivisible resources such as the *maximin share(MMS) guarantee*, *envy-freeness up to one item (EF1)*, and *envy-freeness up to the least valued item (EFX)*[Budish, 2011; Caragiannis *et al.*, 2016]. We present several results related to two open questions: the existence of an EF1 and Pareto optimal mechanism that is computable in polynomial time and the existence of an EFX mechanism.

In the second part, we introduce a real-life application, specifically, the online version of the *roommate market problem*, which we call the *online roommate allocation problem*. The online roommate allocation problem is characterized as follows. In the beginning, we have empty double-bed rooms; then agents arrive one by one in an online fashion. As soon as an agent arrives, we need to assign him a room. After all of the agents have arrived, we want our allocation to be efficient and fair. The model can be used, for example, to organize university accommodation or small working groups in companies.

4.1 General indivisible resource allocation

The most general indivisible resource allocation problems is similar to the cake cutting problem, but the resources cannot be cut, and must be allocated to agents based on the pre-divided pieces (goods). Spliddit [Goldman and Procaccia, 2015] is an application of indivisible resources allocation theories that can help to solve real-life division tasks in several different scenarios, such as allocating goods or chores, or advanced topics such as rent division and taxi fare division.

4.1.1 Related works

If we want to achieve fairness in indivisible resource allocation, we first need new notions of fairness. In this section, we introduce three popular types of fairness notions. In 2011, Budish proposed the maximin share(MMS) guarantee [Budish, 2011], but could not prove the existence of the MMS allocation. A few years later, Ariel and Junxing proved there is no MMS mechanism [Procaccia and Wang, 2014]. In the same paper, they proposed an algorithm that can approximately satisfy the MMS guarantee. More recent studies have improved the approximation ratio, and one has successfully improved the approximation ratio of MMS to $\frac{7}{8} - \varepsilon$ for three agents[Amanatidis *et al.*, 2017]. In another strand of research, researchers have studied the MMS guarantee where every agent needs to get a connected component in the goods graph[Bouveret *et al.*, 2017]. As an alternative to the MMS guarantee, *envy-freeness up to one item (EF1)* has been proposed for solving the indivisible resource allocation problem[Budish, 2011]. Recently, Caragianni el at. showed that, surprisingly, an allocation maximizing the Nash social welfare function would be EF1 and Pareto optimal[Caragiannis *et al.*, 2016]; that study left the open question of whether we can find a polynomially computable allocation that satisfies EF1 and Pareto optimality. In the same paper, they also proposed *Envy-freeness up to the least valued item (EFX)* as well. Building on this research, Plaut and Roughgarden studied EFX direction and showed a series of results in different classes of valuation functions [Plaut and Roughgarden, 2018].

4.1.2 Contiguous maximin share guarantee

We want to satisfy the MMS guarantee in another way instead of an approximation. In some cases, for convenience, we want to make the allocation contiguous to agents, as in the connected piece assumption in the cake cutting problem. We consider a situation in which every agent can only receive a contiguous sequence from the goods list. As in the cake cutting problems with the connected piece assumption, we can make $n-1$ breakpoints on this list and allocate n bundles of items to n agents. Below, we give a formal definition for contiguous maximin share guarantee.

Definition 16. *We say a partition π of a list of goods I is a contiguous partition, if each $S \in \pi$ can be indicated by two numbers i and j, where S will contain item I_k for $i \leq k \leq j$ exactly.*

*The n-*contiguous maximin share guarantee *of a_i is computed as:*

$$CMMS_i^{(n)}(I) = \max_{(S_1,S_2,\cdots,S_n)\in c\pi_n(I)} \min_{j\in[n]} v_i(S_j),$$

where $c\pi_n(I)$ are the set of all of the contiguous n-partitions. The partition that generates the maximal result to a_i is called the n-CMMS partition to a_i.

We say an allocation satisfies the n-CMMS guarantee (or an allocation is an n-CMMS allocation) if we have

$$v_i(A_i) \geq CMMS_i^{(n)}(I).$$

Bouveret et al. independently studied a more general case than ours[Bouveret *et al.*, 2017]. The connected fair division problem contains a graph of goods, and each agent needs to acquire a subset of goods that form a connected component in the graph. They showed that when the graph of goods is a tree, it is always possible to find an allocation. The CMMS allocation can be reduced to a connected fair division problem where the goods graph is a path.

Our result after computing the CMMS allocation is as follows.

Theorem 11. *There is a polynomial-time algorithm for the indivisible resources allocation problem that can always yield an allocation that satisfies the CMMS guarantee.*

Proof. We compute the n-CMMS partition to a_i, totally we get n partitions. Let the breakpoints for the i-th partition be $B_i = (B_i^1 = 0, B_i^1, B_i^2, \cdots, B_i^{n-1}, B_i^n = m)$, where the j-th set of the partition contains items with indices that are between $B_i^{j-1} + 1$ and B_i^j.

Our algorithm works as follows. In the i-th step, we pick the leftmost breakpoint among $\{B_j^i\}$ for all of the agents j who have not been allocated. We can see that everyone gets an interval in which the right endpoint is exactly the same breakpoint as in his original CMMS allocation, but the left endpoint may move to the left of the original left endpoint, which means he can get a bundle that satisfies CMMS. Therefore, this algorithm outputs an allocation that meets the CMMS guarantee.

Each B_i can be calculated within $O(m \log(m^2))$ operations, as there are less than m^2 different possible sums of intervals. We use a binary search to find n-CMMS, where determining the existence of an allocation in which each bundle should be valued as no less than a fix number can be done in $O(m)$. Therefore, we need $O(nm \log m)$ time to find all values of B_i and the time complexity of the main procedure is $O(n^2)$.

□

4.1.3 EF1 and Pareto optimal allocation

Envy-freeness up to one item is a natural generalization of the envy-freeness. One way to satisfy EF1 is to use a Round Robin protocol that lets each agent choose the item he likes most, finally the allocation will be EF1. Consider a pair of agents, someone of them (call him a_1) always pick items before the other (call him a_2). a_1 does not envy a_2 clearly; if we remove the first chosen item from a_1's items, a_2 always picks items before a_1, thus establishing EF1. Until recently, an open question about EF1 was whether there is an allocation that is simultaneously EF1 and Pareto optimal. A recent study showed the surprising fact that any allocation maximizing the Nash social welfare function will be EF1 and Pareto optimal[Caragiannis *et al.*, 2016]. However, finding an allocation that maximizes Nash social welfare function is NP-hard, so it is unclear whether EF1 and Pareto optimal allocations can be computed in polynomial time.

In our study, we study a small case and show a polynomial-time algorithm to find an EF1 and Pareto optimal allocation for two agents. First, we introduce a notion that is used to make the proofs easier.

Definition 17. *A* snapshot *of an allocation* $(A_1, A_2, \cdots, A_n)$ *is a point in* n *dimension space with a coordinate of* $(S_1, S_2, \cdots, S_n)$*, where* $S_i = v_i(A_i)$*.*

Snapshots map allocations onto high dimensional space. The following lemma connects snapshots and Pareto optimality.

Lemma 2. *All of the allocations with a snapshot that has the farthest Euclid distance to any plane* $c_1x_1 + c_2x_2 + \cdots + c_nx_n = 0$ *(*$c_i > 0$ *for all* i*) are Pareto optimal.*

Proof. Maximizing the distance from snapshot $(x'_1, x'_2, \cdots, x'_n)$ to $c_1x_1 + c_2x_2 + \cdots + c_nx_n = 0$ is equivalent to maximizing

$$S = c_1x'_1 + c_2x'_2 + \cdots + c_nx'_n.$$

Suppose the snapshot $(x'_1, x'_2, \cdots, x'_n)$ has maximized S. As $c_i > 0$, if there is another snapshot in which all of the x'_i are not decreasing and some x'_j are becoming strictly larger, there is the implied contradiction that there is an allocation with a larger S than the maximal one. □

With this lemma, we can prove our main result.

Theorem 12. *Given two agents and their valuations of* m *items, an EF1 and Pareto optimal allocation can be found in polynomial time.*

Proof. Assume there are m items. We sort all of the items in the non-increasing order of $\frac{v_2}{v_1}$. For each i, we let $c_1 = v_2(g_i)$ and $c_2 = v_1(g_i)$ (g_i is the i-th item in the sorted goods list), and let a_1 take the first i items and let a_2 take the last $m - i$ items. Each allocation is Pareto optimal by Lemma 2, as it maximizes the value of $c_1x_1 + c_2x_2$. Then we choose the smallest k such that when a_1 gets the first k items, he will not envy a_2 up to one item. We claim this allocation is EF1 and Pareto optimal.

We first show why a_2 will not envy a_1 up to one item. We first take away k-th item, so a_1 gets the first $k-1$ items and a_2 gets the last $m-k$ items. It is also a Pareto optimal allocation by Lemma 2. As k is the smallest number that a_1 will not envy a_2 up to one item, a_1 must envy a_2 before he gets k-th item. At this moment, a_2 must not envy a_1, otherwise we swap their allocations and both agents' utilities will strictly increase, which contradicts the Pareto optimality of the allocation. Therefore, after a_1 receives the k-th item, a_2 will not envy him after a_1 disposes of the k-th item. Therefore, EF1 and Pareto optimality holds. The time complexity is $O(m \log m)$. $\square$

Although the above algorithm works well for the case of two agents, it is impossible to generalize to more agents as the following result shows.

Fact 2. *Although we have Lemma 2, it is NP-hard to determine whether there is an EF1 allocation among the Pareto optimal allocations generated by maximizing* $c_1x_1 + c_2x_2 + \cdots + c_nx_n$.

Proof. The partition problem is a standard NP complete problem where it wants to divide a set of positive integers into two subsets that havethe same sum. We can reduce the partition problem to our task: in a partition problem instance, we have n numbers $\{s_i\}$. Let $c_1 = c_2 = c_3 = 1$, $v_1(g_i) = v_2(g_i) = s_i$ and $v_3(g_i) = 0$ for $1 \le i \le n$. We set up two virtual items:

$$v_1(g_{n+j}) = v_2(g_{n+j}) = \frac{\sum_i s_i}{2}, v_3(g_{n+j}) = +\infty, 1 \le j \le 2.$$

As a result, a_3 takes items g_{n+1} and g_{n+2}, and a_1 and a_2 will take the first n items. If there is an algorithm that can determine whether an EF1 allocation exists in this case, it also solves the partition problem as well, as an EF1 allocation implies $v_1(A_1) = v_1(A_2)$, where n items with value $v_1(g_i) = s_i$ are allocated to a_1 and a_2. $\square$

4.1.4 Approximate EFX allocation

Unlike EF1 allocations, it is unclear whether EFX allocations exist. There are two possible answers to this open question:

- No EFX allocation exists. Pairwise MMS guarantee is a stronger notion in which each pair of agents satisfies the MMS guarantee with the union of items in their allocations. It is clear that pairwise MMS allocation is EFX. However, it is unclear whether pairwise MMS allocations exist currently.
- An EFX allocation always exists. In our study, we make an attempt on the variations of EFX. We impose some additional conditions, such as approximating or throwing away some resources.

In Chapter 2, we introduce the envy-free graph. It can be used to find EFX allocations efficiently when the agents value the items in the same order. Furthermore, it can also be used to approximate an MMS allocation. There is an algorithm that can achieve $\frac{1}{2}$-EFX [Plaut and Roughgarden, 2018] based on the envy-free graph. However, that algorithm may incur exponential running time in the worst case. In this subsection, we propose a polynomial-time algorithm for three agents. For convenience, we normalize the sum of the valuations of all of the items to 1.

Theorem 13. *There is a polynomial-time algorithm for three agents that can always output an $\frac{1}{2}$-EFX allocation.*

We first present several lemmas.

Lemma 3. *There is a polynomial-time algorithm to compute an EFX allocation for two agents.*

Proof. For two agents, an EFX allocation can be achieved using the cut-and-choose protocol: one agent divides the items into two bundles by repeatedly adding the highest valued item to the least valued bundle (in the beginning, there are two empty bundles), until there is nothing left; after that, the other agent chooses the bundle he prefers. The first agent is EFX whatever he gets, and clearly the second agent does not envy the first agent. □

Lemma 4. *If one agent gets a subset of items that is valued as no less than $1/3$, then no matter how the other items are allocated, he will be $\frac{1}{2}$-EFX.*

Proof. As he gets a value of at least $1/3$, the total value of all of the other items is at most $2/3$, which is no more than twice the value he already has. Therefore, it establishes $\frac{1}{2}$-EFX. $\square$

Lemma 5. *If there is an item that is valued as no less than $1/3$ by some a_i, there is an $\frac{1}{2}$-EFX allocation.*

Proof. We allocate the item g, which is valued as no less than $1/3$, to a_i, and then compute an EFX allocation with all items except g for the other two agents by Lemma 3. The other two agents will not envy a_i after we have removed the only item a_i has. By Lemma 4, a_i is $\frac{1}{2}$-EFX, so this is is an $\frac{1}{2}$-EFX allocation to all of the agents. $\square$

Lemma 6. *Assume that each item is valued as at no more than $1/3$ by the agents. If there is a subset of items that an agent values as no less than $1/3$ and the other two agents values as no more than $1/3$, an $\frac{1}{2}$-EFX allocation can be found.*

Proof. Assume that a_1 values a subset of items S as no less than $1/3$ then it is $\frac{1}{2}$-EFX to a_1 by Lemma 4. We compute an EFX allocation with the remaining items for a_2 and a_3 by Lemma 3, and we claim both a_2 and a_3 can get a value of at least $1/6$. As every item is valued as no more than $1/3$ and the total value of the remaining items is at least $2/3$, the valuations of the two bundles in the most unbalanced EFX allocation are $1/6$ and $3/6$. Finally, $1/6 \geq \frac{v_i(S)}{2}$ for $i \in \{2, 3\}$ implies $\frac{1}{2}$-EFX.

$\square$

Lemma 7. *Assume each item is valued as no more than $1/3$ by the agents. If there is a subset of items that two agents value as no less than $1/3$ and the other agent values it as no more than $1/3$, then an $\frac{1}{2}$-EFX allocation can be found.*

Proof. Suppose a_1 and a_2 value the subset S as no less than $1/3$ and a_3 values S as no more than $1/3$. First, we process a refinement on S, in which we repeatedly remove the least valued item in S until S is valued as less than $1/3$ after we apply one more removal. Then, we do the same refinement for a_2. After that, there are two possible cases: if one of two agents values S as less than $1/3$, then we are done by

Lemma 6; otherwise, both agents value S as no less than $1/3$ and it must be less than $2/3$ (otherwise, the refinement will not stop, as an item is valued as no more than $1/3$).

Next, we let a_3 divide the $I \backslash S$ into two bundles that both satisfy EFX, which can be done by Lemma 3. Now we have three bundles, which we call b_1 (it is S), b_2 and b_3 (divided by a_3), and we want to let every agent take one bundle. We claim that the following events will not disobey $\frac{1}{2}$-EFX:

- a_1 (or a_2) takes b_1;
- a_1 (or a_2) takes the better of b_2 or b_3;
- a_3 takes either b_2 or b_3.

By Lemma 4, a_1 and a_2 are $\frac{1}{2}$-EFX if they get b_1. As b_1 is valued as no more than $2/3$, either b_2 or b_3 is valued as no less than $1/6$. Due to the refinement, after we remove the least valued item in b_1, its value will be less than $1/3$, which establishes $\frac{1}{2}$-EFX when a_1 or a_2 get a value of at least $1/6$.

Finally, we want to prove that a_3 can take one bundle of b_2 and b_3 to satisfy $\frac{1}{2}$-EFX. The least valued bundle is valued as no less than $1/6$ (in the worst case, the valuations of three bundles are $v_3(b_1) = 1/3$, $v_3(b_2) = 1/6$ and $v_3(b_3) = 1/2$), which is at least half of $v_3(b_1) = 1/3$.

Therefore, an $\frac{1}{2}$-EFX allocation can be found: we let a_1 take b_1, let a_2 choose the better of b_2 and b_3, and then let a_3 take the last bundle. $\square$

Next we prove our theorem. We also illustrate our algorithm in pseudocode.

Proof of Theorem 13. If there is any item that is valued as no less than $1/3$ to any agent, by Lemma 5 we are done. Otherwise, each item is valued as less than $1/3$. Whenever there is a bundle of items that exactly one agent values as no less than $1/3$, by Lemma 6 we are done. Similarly, if there is a bundle that exactly 2 agents who value it as no less than $1/3$, by Lemma 7 we are done. Then we only need to consider the cases where each bundle is valued as no less than $1/3$ to all or none of

Algorithm 1 $\frac{1}{2}$-EFX Allocation for 3 agents

```
function IdenticalOrderGoodsEFXAllocation(goods, v, n)
    // Any item has the same value to all agents
    Sort the items by the value v in the non-increasing order
    for i ∈ goods do
        Pick j with the least value v(A_j)
        A_j ← A_j ∪ {i}
    end for
    return A
end function
function SatisfyingGoodsSetForAgent1(goods, b_1, v)
    // a_1 must be satisfying with b_1
    // a_3 values b_1 as less than 1/3
    // Lemma 6 and Lemma 7 are considered together in this function
    (b_2, b_3) ←IdenticalOrderGoodsEFXAllocation(goods/b_1, v_C, 2)
    if a_2 prefers b_3 to b_2 then
        Swap b_2 and b_3
    end if
    return (b_1, b_2, b_3)
end function
```

function 1/2-EFXGOODSALLOCATIONFORTHREEAGENT(*goods*, v)
  if i and g exist such that $v_i(g) \geq \frac{1}{3}$ **then**
    $b_1 \leftarrow \{g\}$
    $(b_2, b_3) \leftarrow$IDENTICALORDERGOODSEFXALLOCATION(*goods*$\backslash\{g\}, v_C, 2)$
    if a_2 prefers b_3 to b_2 **then**
      Swap b_2 and b_3
    end if
    return (b_1, b_2, b_3)
  end if
  (b_1, b_2, b_3) = IDENTICALORDERGOODSEFXALLOCATION(*goods*, $v_1, 3)$
  if There exists $S \in \{b_1, b_2, b_3\}$, such that exactly one agent a_j satisfies $v_j(S) \geq 1/3$ **then**
    Let *perm* be any permutation that makes a_j become the first agent
    $(b_1, b_2, b_3) \leftarrow$ SATISFYINGGOODSSETFORAGENT1(*goods*, S, *perm*(v_A, v_B, v_C))
    return *perm*$^{-1}(b_1, b_2, b_3)$
  end if
  if There exists $S \in \{b_1, b_2, b_3\}$, such that exactly two agent a_j and a_k consider $(v_j(S) \geq 1/3$ and $v_k(S) \geq 1/3)$ or ($v_2(S\backslash\{g\}) \geq 1/3$ *or* $v_3(S\backslash\{g\}) \geq 1/3$ for some $g \in S$) **then**
    while There exists $g \in S$, such that $v_j(S\backslash\{g\}) \geq 1/3$ for some a_j **do**
      $S \leftarrow S\backslash\{g\}$
    end while
    if Exactly one agent a_j values S as $v_j(S) \geq 1/3$ **then**
      Let *perm* be any permutation that makes a_j become the first agent
      (b_1, b_2, b_3) = SATISFYINGGOODSSETFORAGENT1(*goods*, S, *perm*(v_1, v_2, v_3))
      return *perm*$^{-1}(b_1, b_2, b_3)$
    else// It implies that still two agents value this set as no less than $1/3$
      Assume a_j and a_k value S as greater than $1/3$
      Let *perm* be any permutation that a_j becomes the first agent and a_k becomes the second agent (b_1, b_2, b_3) = SATISFYINGGOODSSETFORAGENT1(*goods*, S, *perm*(v_1, v_2, v_3))
      return *perm*$^{-1}(b_1, b_2, b_3)$
    end if
  end if
  Reorder sets that $v_1(b_1) \leq v_1(b_2) \leq v_1(b_3)$
  if a_3 prefers b_2 more than b_3 **then**
    Swap b_2 and b_3
  end if
  return (b_1, b_2, b_3)
end function

the agents. Then, let a_1 divide the items into b_1, b_2 and b_3 in any way that achieves EFX for himself. We can do this by repeatedly adding the highest valued item to the least valued bundle, which is almost the same as the two bundles case in Lemma 3. Let b_1 be the best bundle to a_1, then $v_i(b_1) \geq 1/3$ for $1 \leq i \leq 3$. We claim that the following events will not disobey $\frac{1}{2}$-EFX:

- a_1 takes any bundle;
- a_2 (or a_3) takes b_1.
- a_2 (or a_3) takes the better of b_2 or b_3;

By Lemma 4, a_2 is $\frac{1}{2}$-EFX if he gets b_1. To prove that the better bundle of b_2 and b_3 is $\frac{1}{2}$-EFX to a_2, we discuss two cases:

- If $v_2(b_1 \backslash \{g\}) \geq 1/3$ where g is the least valued item in b_1. As a_1 first divides the items into three bundles in a way that achieves EFX for himself, $v_1(b_1 \backslash \{g\})$ is at most $1/3$. Then, it can be done by Lemma 6 or Lemma 7.
- If $v_2(b_1 \backslash \{g\}) < 1/3$. The better bundle of b_2 and b_3 is valued as no less than $\frac{1-v_2(b_1)}{2} \geq 1/6$, implying it is at least half of $v_2(b_1 \backslash \{g\}) < 1/3$.

Therefore, we first let a_3 choose any bundle he prefers, then a_2 has a choice that satisfies $\frac{1}{2}$-EFX, and finally let a_1 take the last bundle. Let m be the number of items, the time complexity of our algorithm is $O(m \log m)$, which requires running function *IdenticalOrderGoodsEFXAllocation* for constant times.

□

4.1.5 EFX allocation of chores

In this subsection, we consider chores allocation problems. If we are allowed to dispose of a few chores, proportionality can be satisfied as follows.

Fact 3. *There is no proportional mechanism when we can dispose of no more than $n-2$ chores. There is a proportional mechanism when we can dispose of $n-1$ chores.*

Proof. First, consider a scenario with $n-1$ chores, where each chore is valued as $1/(n-1)$ by everyone. After we remove $n-2$ chores, there is still a chore with a value of $1/(n-1)$, which breaks proportionality for whoever takes this chore.

If we can dispose of $n-1$ chores, the result becomes positive. If there are only two agents, let a_1 divide the chores into two bundles that maximize the least valued bundle. Then let a_2 choose a bundle and let a_1 remove one chore from his bundle, which clearly creates a proportional allocation.

We assume that a proportional mechanism exists for $n-1$ agents. It is done by induction if we solve the case for n agents. Let C be the set of all of the chores and let set S be an empty set. We repeatedly add one chore to S until all agents value it as $v_i(S) \geq 1/n$. We dispose of the last added chore L and allocate $S\backslash\{L\}$ to a_x who satisfies $v_x(S\backslash\{L\}) \leq 1/n$. Then we are left with $n-1$ agents and $C\backslash S$ chores, which becomes the desired subproblem. As all agents value S as $v_i(S) \geq 1/n$, the proportionality in the subproblem implies $v_i(A_i) \leq \frac{1-v_i(S)}{n-1} \leq 1/n$, which finishes our proof. $\square$

Next, we consider EFX or proportional as a new fairness notion. The formal definition is as follows.

Definition 18. *An allocation is* EFX or proportional *(EFX or PR hereafter), if every agent satisfies at least one of the following two conditions:*

1. *the agent's allocated bundle is proportional; or*

2. *the agent's allocated bundle is EFX.*

Our main result in this subsection is as follows.

Theorem 14. *In the chores allocation stting, there exists an EFX or PR mechanism for three agents, if it is allowed to dispose of at most one chore. Moreover, the mechanism is polynomially computable.*

We first present several lemmas.

Lemma 8. *If there is a subset S of chores where exactly one agent values it at most $1/3$, then there is an EFX or PR allocation.*

Proof. Assume a_1 values S as no more than $1/3$, and a_2 and a_3 value S as no less than $1/3$. Then let a_2 divide all of the chores except S into two bundles in a way that achieves EFX for himself (as with goods, this can be done by repeatedly adding the heaviest chore to the least valued bundle). Let a_1 take S, a_3 choose the bundle he prefers, and a_2 take the last bundle. S is proportional to a_1, a_3 takes the bundle he likes the best (the value he gets is at most $1/3$). After we remove the least valued chore in a_2's bundle, the value must be less than $\frac{1-v_2(S)}{2} \leq 1/3$, which is also less than $v_2(S) \geq 1/3$. □

Lemma 9. *If there is subset S where exactly two agents value S as no more than $1/3$, there is an EFX or PR allocation.*

Proof. Assume a_1 and a_2 value S as no more than $1/3$, and a_3 values S as no less than $1/3$. Let a_3 divide all of the chores except S into b_2 and b_3 in a way that achieves EFX for himself. If we assume $v_3(b_2) \geq v_3(b_3)$, we have $v_3(b_3) \leq \frac{1-v_3(S)}{2} \leq 1/3$. If a_1 and a_2 value b_3 at least $1/3$, then we can find an EFX allocation by Lemma 8. Otherwise, without loss of generality, we assume that a_1 values b_3 at most $1/3$, thus he is satisfied to take S or b_3 by proportionality. Therefore, we let a_1 take b_3, and a_2 take S, then a_3 takes b_2. a_1 and a_2 are satisfied by proportionality, and for a_3, we have $v_3(b_2 \backslash \{c\}) \leq v_3(b_3) \leq v_3(b_1)$ as well, where c is the least valued chore in b_2. □

With these two lemmas, the proof for the following theorem can be done. We also provide pseudocode.

Proof of Theorem 14. By Lemma 8 and 9, if there is a set S that is valued as no more than $1/3$ by some agents but not by all agents, then we are done. Otherwise, if any agent values a bundle as no more than $1/3$, then all agents will value this bundle no more than $1/3$.

Let a_1 divide the chores into three bundles (b_1, b_2, b_3) in a way that achieves EFX for himself. Assume $v_1(b_1) \leq v_1(b_2) \leq v_1(b_3)$. As $v_1(b_1) \leq 1/3$, all of the agents value b_1 as no more than $1/3$. Our algorithm works as follows.

Algorithm 2 EFX or PR allocation for 3 agents with disposing of one chore

```
function IdenticalChoresEFXAllocation(chores, v, n)
    // Any chore has the same value to all agents
    Sort the chores by the value v in the non-decreasing order
    for i ∈ chores do
        Pick j with the least value v(A_j)
        A_j ← A_j ∪ i
    end for
    return A
end function
function SatisfyingChoresSetForAgent1(chores, b_1, v)
    // b_1 is proportional to a_1
    // b_1 is not proportional to a_2 and a_3
    (b_2, b_3) ←IdenticalOrderChoresEFXAllocation(chores/b_1, v_3, 2)
    if a_2 prefers b_3 more than b_2 then
        Swap b_2 and b_3
    end if
    return (b_1, b_2, b_3)
end function
function SatisfyingChoresSetForAgent12(chores, b_1, v)
    // b_1 is proportional to a_1 and a_2
    // b_1 is not proportional to a_3
    (b_2, b_3) ←IdenticalOrderChoresEFXAllocation(chores/b_1, v_3, 2)
    Let v_3(b_2) ≤ v_3(b_3)
    if v_1(b_2) ≤ 1/3 then
        return (b_2, b_1, b_3)
    end if
    if v_2(b_2) ≤ 1/3 then
        return (b_1, b_2, b_3)
    end if
    Let perm be any permutation that makes a_3 become the first agent
    return perm^{-1} (SatisfyingChoresSetForAgent1(chores, b_2, perm(v_1, v_2, v_3)))
end function
```

function EFXORPRCHOREALLOCATIONFORTHREEAGENTEXCEPTONECHORE(*chores*, v)
    (b_1, b_2, b_3) = IDENTICALORDERCHORESEFXALLOCATION(*chores*, v_1, 3)
    if There exists $S \in \{b_1, b_2, b_3\}$, such that exactly one agent a_j values S as $v_j(S) \leq 1/3$ **then**
        Let *perm* be any permutation that makes a_j become the first agent
        $(b_1, b_2, b_3) \leftarrow$SATISFYINGCHORESSETFORAGENT1(*chores*, S, *perm*(v_1, v_2, v_3));
        return *perm*$^{-1}(b_1, b_2, b_3)$
    end if
    if There exists $S \in \{b_1, b_2, b_3\}$, such that exactly two agents a_j and a_k value S as $v_j(S) \leq 1/3$ and $v_k(S) \leq 1/3$ **then**
        Let *perm* be any permutation that a_j becomes the first agent, a_k becomes the second agent
        $(b_1, b_2, b_3) \leftarrow$SATISFYINGCHORESSETFORAGENT12(*chores*, S, *perm*(v_1, v_2, v_3));
        return *perm*$^{-1}(b_1, b_2, b_3)$
    end if
    Reorder (b_1, b_2, b_3) such that $v_1(b_1) \leq v_1(b_2) \leq v_1(b_3)$
    Let *leastChore*$_B$ and *leastChore*$_C$ be the least valued chore in b_2 and b_3 to agent A
    if Agent A prefers $b_2 \backslash \{$*leastChore*$_2\}$ more than $b_3 \backslash \{$*leastChore*$_3\}$ **then**
        Swap b_2 and b_3, so *leastChore*$_2$ and *leastChore*$_3$ are also swapped
    end if
    $b_2 \leftarrow b_2 \backslash \{$*leastChore*$_2\}$, remove *leastChore*$_2$ from *chores*
    if $v_2(b_2) \leq 1/3$ **then**
        return (b_3, b_2, b_1)
    end if
    if $v_3(b_2) \leq 1/3$ **then**
        return (b_3, b_1, b_2)
    end if
    // Then, only a_1 values b_2 at most $1/3$
    $(b_1, b_2, b_3) \leftarrow$SATISFYINGCHORESSETFORAGENT1(*chores*, b_2, v);
    return (b_1, b_2, b_3)
end function

1. If $v_1(b_2) \leq 1/3$, let a_2 take b_2 , a_3 take b_1 and a_1 take b_3, and our algorithm stops.

2. If $v_1(b_2\backslash\{c_2\}) \geq v_1(b_3\backslash\{c_3\})$ where c_i is the least valued chore in b_i, dispose of c_2 from the set of chores; otherwise, dispose of c_3 from the set of chores and swap b_2 and b_3.

3. If b_2 is valued at least $1/3$ by all or none of the agents, go to Step 4; otherwise output an allocation based on Lemma 8 or 9.

4. Let a_2 take b_2, a_3 take b_1 and a_1 take b_3, and our algorithm stops.

If the algorithm stops at Step 1, b_1 and b_2 are valued as no more than $1/3$ by all of the agents, and b_3 is EFX to a_1; therefore, the allocation is EFX or PR. In Step 3, only b_1 is valued at most $1/3$ in the beginning. Due to EFX, $v_1(b_2\backslash\{c_2\}) \leq v_1(b_1) \leq 1/3$, where c_2 is the least valued chore in b_2 (it also holds for b_3). Therefore, b_2 is proportional to a_2 and b_1 is proportional to a_3; otherwise the case will break the property that one bundle is valued at least $1/3$ by all or none of the agents, which can be solved in Step 3 by Lemma 8 or 9. Last, a_1 is EFX when he takes b_3, as $v_1(b_2\backslash\{c_2\}) \geq v_1(b_3\backslash\{c_3\})$ in Step 2, which finishes our proof. □

4.2 Online roommate allocation problem

The roommate market problem is highly relevant to real life. In this problem, we expect to know all of the agents' valuations of rooms and other agents. However, in many practical scenarios, we may not be able to gather all of the information from all of the agents. We study the case where agents arrive in an online fashion. In this section, we propose an algorithm for online roommate allocation and promise a good constant competitive ratio of optimal social welfare. Moreover, we study generalizations of our model, such as some rooms can have a larger capacity than two agents, and rooms have an individual capacity. In addition to efficiency, we study several kinds of stability.

4.2.1 Related works

The roommate market is a generalization of the matching problem. In this sense, there are two strands of research related to our work: the stable matching and the online bipartite matching. The stable matching is a classic problem in graph theory, in which we need to find a match and satisfy some stability requirement. In the stable marriage problem, we need to select pairs of ladies and gentlemen and let them have a stable marriage. Similarly, our roommate matching model needs to assign two people to live in a room and to provide some stability. This problem has been studied for about 60 years. It was first proposed in 1962 [Gale and Shapley, 1962]. Several surveys of the history of stable matching are available [Knuth, 1997; Iwama and Miyazaki, 2008]. The basic stable matching problem has been extended in many dimensions. One direction is to consider general matchings [Irving, 1985; Irving and Manlove, 2002] and higher dimensional matchings [Ng and Hirschberg, 1991; Eriksson *et al.*, 2006; Huang, 2007]. Different stability notions have also been proposed, such as exchange stability [Cechlárová and Manlove, 2005] and popular matching [Biró *et al.*, 2010].

Online matching is another generalization of the secretary problem, see [Ferguson and others, 1989] for more details about the secretary problem. In 1990, Karp et al. provided an algorithm with the optimal competitive ratio for unweighted bipartite graph matching problems [Karp *et al.*, 1990]. In 2013, the algorithm was extended to the weighted bipartite matching [Kesselheim *et al.*, 2013]. Many variants have been proposed and analyzed, such as vertex-weighted matching [Aggarwal *et al.*, 2011] and online packing [Kesselheim *et al.*, 2014].

In a roommate matching scenario, we need to match two agents to one room, and the type of edge is different: an edge between people may indicate an interpersonal relation or an edge between a person and a room may denote the degree to which the person favours the room. Previous studies have developed complete answers to the roommate market problem[Chan *et al.*, 2016]. This thesis focus on the generalization of the online model.

This section is reproduced from our published paper[Huzhang *et al.*, 2017].

4.2.2 Online no-rejection bipartite matching

We are going to show our online algorithm for the online roommate allocation problem with a constant competitive ratio. To sketch our proof, we first introduce an special online bipartite matching problem: no-rejection online bipartite matching problem. In a no-rejection online bipartite matching problem, we have a weighted complete bipartite graph $G = (L, R, E)$, where $|L| = |R| = n$ and all (i, j) will be in E where $i \in L$ and $j \in R$. Node in L will come in an uniformly random order. Every time a node in L arrives, and we need to determine which node in R it should match. After all nodes in L arrived, we want to maximize the weight of matching in expectation. In comparison with the original online bipartite matching problem, we cannot dispose of any node here.

Our algorithm is similar to the original problem's solution provided by [Kesselheim *et al.*, 2013], and our proof also follows a similar analysis in that work. To understand it better, we present the algorithm below.

Lemma 10. *Algorithm 3 has approximation ratio c_b for online no-rejection bipartite graph matching problem, where $c_b = \ln(3)/8$ is a constant.*

Proof. We apply two steps:

1. We prove that if a vertex is matched according to the local maximum matching, then the expected weight of this matching is $1/n$ of maximum matching.
2. We lower bound the probability of successful matching by local maximum matching in each round.

Let M_{opt} be the weight of maximum matching between L and R. Let random variable w_k denote the weight of the edge chosen by a maximum matching in round k, i.e. the matching among first k left vertices and all right vertices. Now we prove the following claim.

Claim 1. *For $k > n/5$, $\mathbb{E}[w_k] \geq M_{opt}/n$.*

Algorithm 3 ONLINEMATCHING(n, R)

$counter \leftarrow 0$
$L \leftarrow \emptyset$
$A \leftarrow \emptyset$
for every person v that arrives **do**
  $L \leftarrow L \cup \{v\}$
  $counter \leftarrow counter + 1$
  if $counter \geq n/5$ **then**
    $M^v \leftarrow$ Optimal matching on $G[L \cup R]$
    $e^v \leftarrow$ The matching edge that contains v in M^v
    if $A \cup e^v$ is a matching **then**
      $A \leftarrow A \cup e^v$
    else
      Randomly choose an available vertex v'.
      $A \leftarrow A \cup (v, v')$
    end if
  else
    Randomly choose an available vertex v'.
    $A \leftarrow A \cup (v, v')$
  end if
end for
return A

Proof. Roughly speak, we need to prove that expected weight of matching will be increased by M_{opt}/n when we successfully add one edge to matching.

The k-th vertex can be viewed as picking by the following process:

1. First, choose k vertices uniformly random from n vertices;

2. Second, choose one vertex uniformly random from those k vertices.

Let M^k be the maximum matching between the first k vertices and R. Due to the property that left vertices are uniformly coming, the expected weight of M^k is at least $\frac{k}{n}M_{opt}$. Since the k-th left vertex is picked randomly, the expected weight of w_k is $1/k$ of M^k. Thus, we have

$$\mathbb{E}[w_k] \geq \frac{1}{k}\frac{k}{n}M_{opt} = \frac{1}{n}M_{opt}.$$

$\square$

Let B_k be the event that the k-th left vertex is matched according to the maximum matching in the k-th round, which means the right vertex it wants to match is unmatched. The following claim gives a lower bound of the probability $\Pr[B_k]$.

Claim 2. *For* $k > n/5$, $\Pr[B_k] \geq \frac{n/5}{k-1} \cdot \frac{n-k+1}{n}$.

Proof. Above claim indicates that the probability of successfully adding one edge to matching is large enough.

Let l be the k-th left vertex. Let r be the vertex matching with l in M^k. We first analysis the probability that r does not match to any previous vertex by a maximum matching.

Let e^v be the right vertex which is matched to v in previous rounds as in algorithm. In any of the preceding rounds $j \in \{\lceil \frac{n}{5} \rceil, \ldots, k\}$, the vertex r was matched only if r is $e^{l'}$ for some l' which comes before l. We define E_k be the event that r becomes matched in round k. The last vertex in the order can be seen as being chosen uniformly at random from the j participating vertices on the left-hand side. Hence, the probability of r being matched in step j is at most $1/j$, i.e. $Pr[E_j] \leq \frac{1}{j}$. The order of the vertices $1, \ldots, j-1$ is irrelevant for this event. Therefore, also the respective events if some vertex $j' < j$ was matched to r can be regarded as independent.

For an easier analysis, we consider the following new model: we first label $k-1$ right vertices to be forbidden for matching, and when we want to randomly pick an edge for matching, we can randomly select a vertex in the forbidden set to match it. Thus it can be considered that we first randomly pick $k-1$ vertices and throw away them before considering the maximum matching edges, the probability, that one vertex is matched by a random picking step, will be easy to bound. Let B'_k be the event same as B_k in the new model. Because we will make more vertices forbidden for matching by this changing order operation, any vertices will have more probability to lose their maximum matching edge and also for k-th vertex, therefore $Pr[B'_k] \leq Pr[B_k]$.

We define C_k be the event that r is not matched by previous maximum matching edge in round k, we have

$$Pr[B_k] \geq Pr[B'_k] = Pr[C_k] * \frac{n-k+1}{n}.$$

Because of independent of E_k, we have

$$\begin{aligned}
\Pr\left[C_k\right] &= \Pr\left[\wedge_{j=\lceil \frac{n}{5} \rceil}^{k-1} \bar{E}_j\right] \\
&\geq \Pi_{j=\lceil \frac{n}{5} \rceil}^{k-1} \frac{j-1}{j} \\
&= \frac{\lceil \frac{n}{5} \rceil - 1}{k-1}.
\end{aligned}$$

Combining them together, the probability is

$$\Pr[B_k] \geq \frac{n/5}{k-1} \cdot \frac{n-k+1}{n}.$$

$\square$

(back to Lemma 10) Let random variable M denote the weight of resulting matching. By two claims, we have

$$\begin{aligned}
\mathbb{E}[M] &\geq \sum_{k=\lceil n/5 \rceil}^{n} \mathbb{E}[w_k] \cdot \Pr[B_k] \\
&\geq \frac{M_{opt}}{n} \cdot \sum_{k=\lceil n/5 \rceil}^{n} \frac{n/5}{k-1} \cdot \frac{n-k+1}{n} \\
&= \frac{M_{opt}}{n} \cdot \sum_{k=\lceil n/5 \rceil}^{n} \frac{1}{5} \left(\frac{n}{k-1} - 1 \right) \\
&= \frac{M_{opt}}{5} \cdot \left(\sum_{k=\lceil n/5 \rceil}^{n} \frac{1}{k-1} - \sum_{k=\lceil n/5 \rceil}^{n} \frac{1}{n} \right) \\
&\geq \frac{\ln(5) - 0.8}{5} \cdot M_{opt} \\
&\geq 0.1618 M_{opt}.
\end{aligned}$$

□

4.2.3 Online algorithm for online roommate allocation

We look at the happiness first. In the online roommate market problem, the relations between agents form a general graph instead of a bipartite graph. Because the incoming order is uniformly random, it is natural to pick up first half agents to be the left-hand side nodes and last half to be the right-hand side nodes, and then the social welfare will not be lost by a lot in expectation. With this idea, we introduce the following lemma.

Lemma 11. *In a complete general graph with $2n$ vertices and weight matrix W, we make it be a bipartite graph by randomly picking n vertices to be left vertices and other n vertices to be right vertices (so only the crossing edges will be considered in the new bipartite graph). Then, the maximum weight bipartite matching will be $\frac{1}{2}$ approximation for original maximum weight matching in expectation.*

Proof. Look at the edges that appear in original maximum weight matching. One edge will have probability $\frac{1}{2}$ that two vertices of it will be in opposing sides, so by linearity of expectation, the weight of maximum weight bipartite matching will be at least $\frac{1}{2}$ approximation to original maximum weight matching. $\square$

Because of the uniformly random arriving order, we see that we can get a good approximation if we divide the agents into two groups directly by the incoming order. Fortunately, the strategy to partition agents into groups can also make the value between person and room kept in a large approximation in expectation.

We now present our constant competitive ratio algorithm for the online roommate market problem. It uses the online no-rejection bipartite matching algorithm as a key ingredient. The high-level idea is that we first apply Algorithm ONLINEMATCHING on the first n agents arrived. After this stage, each room contains exactly one agent. Then we combine each room-person pair as one new "room", and apply Algorithm ONLINEMATCHING again on the last n agents with adjusted valuations to match them to the n room-person pairs.

Algorithm 4 ONLINEROOMMATE (n, H, V)

```
Run ONLINEMATCHING on the first n agents arrived.
Let M_1 be the output matching.
for every agent a_i arrived after the first n agents do
    for each room r ∈ R do
        Set v'_ir ← v_ir + (h_ij + h_ji) where (j, r) ∈ M_1
    end for
end for
Run ONLINEMATCHING on the last n agents with valuation matrix V'.
Let M_2 be the returned matching.
return M_1 ∪ M_2
```

Theorem 15. *Algorithm* ONLINEROOMMATE *is a polynomial-time and* $\frac{c_b}{4}$*-competitive algorithm for the online roommate market problem, where* $c_b = \frac{\ln 5 - 0.8}{5} \approx 0.1618$.

Proof. Let A_{opt} denote the optimal offline allocation with maximum social welfare. Let M_{pp} denote the maximum weight general graph matching between the $2n$ agents, where the weight between a_i and a_j is $h_{ij}+h_{ji}$. Let M_{pr} denote the maximum weight matching between $2n$ agents and n rooms where each room is duplicated into two vertices. By slight abuse of notations, in the following we use $SW(M)$ to denote the summation of the edge weights in matching M.

We can divide the social welfare $SW(A_{opt})$ into two parts: the first part is the happiness between roommates, which will not exceed $SW(M_{pp})$; the other part is the valuations between agents and the rooms, which will not exceed $SW(M_{pr})$. Hence we have

$$SW(M_{pp}) + SW(M_{pr}) \geq SW(A_{opt}).$$

Next we bound $SW(M_{pp})$ and $SW(M_{pr})$. Fix a particular agents arriving order. Let A_1 be the set of first n agents, A_2 be the set of last n agents, and E_{12} be the set of weighted edges between A_1 and A_2 (where again the weight of edge (i, j) between a_i and a_j is $h_{ij} + h_{ji}$). We further define the following notations:

- M_{pb}: the maximum weight matching in bipartite graph (A_1, A_2, E_{12}).
- M_{pr1}: the maximum weight matching between the first n agents and n rooms
- M_{pr2}: the maximum weight matching between the last n agents and n rooms.

We will show in the following that

$$\begin{aligned}&2\mathbb{E}[SW(M_{pr1}) + SW(M_{pr2}) + SW(M_{pb})]\\ \geq&SW(M_{pp}) + SW(M_{pr}),\end{aligned}$$

where the expection is over the random arriving order of the agents.

First we bound $SW(M_{pb})$. Since agents arrive in an uniformly random order, every edge in M_{pp} will be present in E_{12} with probability at least $\frac{1}{2}$, and these edges together form a matching. We therefore have

$$\mathbb{E}[SW(M_{pb})] \geq \sum_{(i,j)\in M_{pp}} \frac{1}{2}(h_{ij} + h_{ji}) = \frac{1}{2}SW(M_{pp}).$$

Now we bound $SW(M_{pr})$ by $SW(M_{pr1})$ and $SW(M_{pr2})$. Let M_{pr0} be the maximum weight bipartite matching between $2n$ people and n rooms and each room only has one slot. We have

$$SW(M_{pr1}) + SW(M_{pr2}) \geq SW(M_{pr0}) \geq \frac{1}{2} SW(M_{pr}).$$

The first inequality is because the edges in $M_{pr1} \cup M_{pr2}$ can at least cover all edges in M_{pr0}.

Together with

$$\mathbb{E}[SW(M_{pb})] \geq \frac{1}{2} SW(M_{pp}),$$

we have

$$\begin{aligned} & 2\mathbb{E}[SW(M_{pr1}) + SW(M_{pr2}) + SW(M_{pb})] \\ \geq & SW(M_{pp}) + SW(M_{pr}) \\ \geq & SW(A_{opt}). \end{aligned}$$

Back to our algorithm, the first call to ONLINEMATCHING gives us a matching with expected social welfare no less than $c_b \cdot SW(M_{pr1})$; the second call to ONLINEMATCHING gives us a matching with expected social welfare no less than $c_b \cdot \max\{SW(M_{pb}), SW(M_{pr2})\}$. Let A denote the allocation output by our algorithm. Together we have

$$\begin{aligned} & \mathbb{E}[SW(A)] \\ \geq & c_b \cdot \mathbb{E}[SW(M_{pr1})] + c_b \cdot \max\{\mathbb{E}[SW(M_{pb})], \mathbb{E}[SW(M_{pr2})]\} \\ \geq & c_b \cdot \mathbb{E}[SW(M_{pr1})] + \frac{c_b}{2} \mathbb{E}[SW(M_{pb}) + SW(M_{pr2})] \\ \geq & \frac{c_b}{2} \mathbb{E}[SW(M_{pr1}) + SW(M_{pb}) + SW(M_{pr2})] \\ \geq & \frac{c_b}{4} SW(A_{opt}). \end{aligned}$$

□

4.2.4 Generalization to c-beds

In real-life situations, one room may contain more than two agents sometimes. We consider the case where a room can contain $c > 2$ people, and take the advantage of our algorithm's flexibility, an algorithm can be developed for c-bed rooms.

Definition 19. *In* online roommate matching with c-beds rooms*, there will be cn agents and n rooms, we need to assign c agents per room. It need to maximize*

$$\sum_{(i_1,i_2,\ldots,i_c,r)\in A}\Big(\sum_{1\le j<k\le c} H_{i_j i_k} + \sum_{j=1}^{c} v_{i_j r}\Big)$$

For this kind of generalization, the idea is similar: we partition the agents into c groups and every group will match to the set of one slot of the rooms. But the analysis is harder because of more edges involved in the proof.

Algorithm 5 ONLINECBEDROOMMATE (n, H, V)

$V' \leftarrow V$
for $g = 1, 2, \ldots, c$ **do**
  Run ONLINEMATCHING on the next n arriving agents with valuation matrix V'.
  Let M_g be the returned matching.
  for every a_i that is yet to arrive **do**
    $v'_{ir} \leftarrow v'_{ir} + h_{ij} + h_{ji}$ where $(j, r) \in M_g$
  end for
end for
return $\cup M_g$

Theorem 16. ONLINECBEDROOMMATE *has competitive ratio $\frac{c_b(c-1)}{c^3}$ for the generalized online roommate market problem, where $c_b = \frac{\ln 5 - 0.8}{5} \approx 0.1618$.*

Proof. The proof is along similar lines of that for Theorem 15.

Let A_{opt} denote the optimal offline allocation with maximum social welfare. Let A_{pp} denote the allocation of cn agents into n rooms that with maximum total happiness valuation between agents, and let M_{pr} denote the maximum bipartite matching between cn agents and n rooms where each room is duplicated into c copies. Again it is straightforward to see that $SW(A_{opt}) \le SW(A_{pp}) + SW(M_{pr})$.

Note that for each room r, the c agents in this room will have total happiness contributed by $\frac{c(c-1)}{2}$ pairs of relations. Let C_r be the general graph *matching* with maximum total happiness among the c agents allocated to room r in allocation A_{pp}. Because each clique of c vertices can always be covered by c matchings via the round-robin tournament algorithm, we can see that $SW(A_{pp}^r) \leq c \cdot SW(C_r)$, where A_{pp}^r is the sum of happiness of all agents in room r in allocation A_{pp}.

Now go back to the online random arrival order. We call every n consecutively arriving vertices a block. The probability of any two vertices arrive in different blocks is $\frac{cn-n}{cn-1}$. Let M_{pp}^{ij} represent the maximum weight bipartite matching between agents in the i-th block and j-th block, and we can assume $c > 2$. We can distribute all matching edges in each C_r into some M_{pp}^{ij} if the two vertices of the edge are in different blocks, and every vertex will be involved in at most one matching edge. Hence by linearity, we have

$$\begin{aligned}\sum_{1 \leq i < j \leq c} \mathbb{E}[SW(M_{pp}^{ij})] &\geq \frac{cn-n}{cn-1} \sum_{r=1}^{n} SW(C_r) \\ &> \frac{c-1}{c} \sum_{r=1}^{n} SW(C_r).\end{aligned}$$

Put everything together, we have

$$\begin{aligned}SW(A_{pp}) = \sum_{r=1}^{n} SW(A_{pp}^r) &\leq \sum_{r=1}^{n} c \cdot SW(C_r) \\ &\leq \sum_{1 \leq i < j \leq c} \frac{c^2}{c-1} \cdot \mathbb{E}[SW(M_{pp}^{ij})].\end{aligned}$$

Next we look at M_{pr}. Let M_{pr}^1 be the matching induced from M_{pr} such that each room connects with only one agent who has the largest matching value. We have $SW(M_{pr}) \leq c \cdot SW(M_{pr}^1)$. Let M_{ir} be the maximum weight matching between agents in i-th block and one slot of each room.

We also have $SW(M_{pr}^1) \leq \sum_{i=1}^{c} SW(M_{ir})$. Combing these two inequalities gives us

$$SW(M_{pr}) \leq c \sum_{i=1}^{c} SW(M_{ir}) < \frac{c^2}{c-1} \sum_{i=1}^{c} SW(M_{ir}).$$

Together with A_{pp}, we have

$$
\begin{aligned}
SW(A_{opt}) &\leq SW(A_{pp}) + SW(M_{pr}) \\
&\leq \frac{c^2}{c-1} \cdot \mathbb{E}\left[\sum_{1\leq i<j\leq c} SW(M_{pp}^{ij}) + \sum_{i=1}^{c} SW(M_{ir})\right].
\end{aligned}
$$

Finally, let M_i be the maximum weight matching between the agents in i-th block and aggregated agent-room combinations according to the algorithm. We have

$$
SW(M_i) \geq \max\{SW(M_{ir}), \max_{1\leq j\leq i-1} SW(M_{pp}^{ij})\}.
$$

Since there are at most c items in the max bracket, we have $c \cdot SW(M_i) \geq SW(M_{ir}) + \sum_{j=1}^{i-1} SW(M_{pp}^{ij})$. Thus,

$$
\begin{aligned}
&\frac{c^2}{c-1} \cdot \mathbb{E}\left[\sum_{i=1}^{c} c \cdot SW(M_i)\right] \\
&\geq \frac{c^2}{c-1} \cdot \mathbb{E}\left[\sum_{1\leq i<j\leq c} SW(M_{pp}^{ij}) + \sum_{i=1}^{c} SW(M_{ir})\right] \\
&\geq SW(A_{pp}) + SW(M_{pr}) \geq SW(A_{opt}).
\end{aligned}
$$

By Lemma 10, the algorithm computes M_i^* which satisfies $\mathbb{E}[SW(M_i^*)] \geq c_b \cdot SW(M_i)$. Therefore, we have

$$
\mathbb{E}\left[\sum_{i=1}^{c} SW(M_i^*)\right] \geq \frac{c_b(c-1)}{c^3} SW(A_{opt}).
$$

This finishes the proof of this theorem. □

4.2.5 Rooms with different capacities

The model can be further generalized to allow rooms to have different capacities. Assume we have n agents and k rooms with capacities $\vec{c} = (c_1, c_2, \ldots, c_k)$ such that $\sum_i c_i = n$.

Corollary 12. ONLINEGENCBEDROOMMATE *achieves constant competitive ratio for the generalized online roommate market problem where every room has constant capacity.*

Algorithm 6 ONLINEGENCBEDROOMMATE $(\vec{c}, H, V)$

$c \leftarrow \max\{c_1, \ldots, c_k\}$
$V' \leftarrow V$
for $g = 1, 2, \ldots, c$ **do**
  Let n_g be the number of available rooms.
  Run ONLINEMATCHING on the next n_g arriving agents with valuation matrix V' (only rooms with open capacities are used).
  Let M_g be the returned matching.
  for every a_i that is yet to arrive **do**
    $v'_{ir} \leftarrow v'_{ir} + h_{ij} + h_{ji}$ where $(j, r) \in M_g$
  end for
  for every room i with $c_i > 0$ **do**
    $c_i \leftarrow c_i - 1$
  end for
end for
return $\cup M_i$

Proof. We use a similar algorithm as for generalized online roommate market. In fact, the algorithm also worked for regular rooms, it is a kind of generalized for original version: every time we pick all rooms with available slot, say r rooms in total, then let next r agents match with the rooms with the weight that sum up all happiness of agents who are already in the room and room valuation.

For the happiness, we calculate the maximal happiness without considering about room valuation, and we also divide every clique of agents by matching and select the maximum matching in every clique. Same as before, we have each sum of a clique is still not more than c times of maximal happiness sum by matching in the same clique, and combine all the matching we will have a constant competitive ratio for it in expectation.

For the valuation of rooms, we look at the first matching which is from n agents to n rooms. In expectation, it is $1/c$ of the maximum matching between all agents and all slots of rooms. As before, this matching is $1/c$ of optimal allocation which has maximum room valuation between all agents and rooms, notice here we will lose by $1/c$. So the ratio will still be a constant. □

4.2.6 Fairness

We will meet some cases that need to consider the individual mind, and the stable marriage problem is a good example. Previous study about roommate market also works with different kinds of stabilities. In the online setting, stability is harder to guarantee, and it is a challenge to have stability. In this section, we discuss different stability conditions in the online roommate market model.

In our work, we will only discuss stability notions that guarantee to exist in the offline setting, which are the 4-*person stability* and *room stability*. In addition, we will also consider a new notion – *weak room-stability*.

Our goal is to design online algorithms that can satisfy certain stability conditions, and further can gain high social welfare. We summarize our results in the following table.

Stability type	Achievable	Social welfare competitive ratio
4-person stable	no	-
Room stable	yes	unknown
Weakly room-stable	yes	constant

In the offline setting, Chan et al. gave an algorithm that can find a 4-person stable solution in $O(n^2)$ time[Chan *et al.*, 2016]. However, no algorithm can always guarantee a 4-person stable solution in the online setting.

Lemma 13. *No algorithm can always find a* 4*-person stable allocation in the online roommate market setting.*

Proof. We prove this lemma by a simple example with four agents and two rooms. Assume that every agent has value 0 for every room. Hence we only need to consider the happiness values between them. We also assume that happiness values are symmetric, i.e., $h_{ij} = h_{ji}$ for every a_i and a_j. Consider the following sequence of agents' arrival: let 1 and 2 be the first and second arriving agents with $h_{12} = 1$. When a_2 arrives, any algorithm needs to make one of two choices:

- Assign a_1 and a_2 to the same room. In this case, assume that the next two arriving a_3 and a_4 have happiness values $h_{13} = h_{24} = 100$. All other unspecified happiness values are 0. It is easy to check that this already breaks the 4-person stable condition because swapping a_1 and a_3 would make every agent better off.

- Assign a_1 and a_2 to different rooms. In this case, assume that the next two arriving a_3 and a_4 have $h_{34} = 1$ and all other happiness values are 0. Here moving a_1 and a_2 to the same room can improve the utility of every agent.

Note that an online algorithm needs to make an assignment decision at each moment some agent arrives. This means regardless of what one algorithm does, it must be failed to output a 4-person stable solution in one of above two situations. □

When discussing this condition, the happiness valuation between roommates can be ignored because the roommate relation will not be changed. It turns out that this room-stable condition can be satisfied by an online algorithm.

Lemma 14. *There is an online algorithm that always gives a room-stable allocation.*

Proof. The simple serial dictatorship algorithm works as follows: For every arriving agent, assign this agent to his most preferred room.

Now we show that the simple dictatorship algorithm that assigns every arriving agent his most preferred available room can produce a room- stable allocation. Fixing any two rooms r_1 and r_2. Suppose the arriving order among them is a_1, a_2, a_3, a_4. If a_1 and a_2 both choose the same room, then they would not want to move to the other room. If a_1 and a_2 choose different rooms, without loss of generality, assume a_1 chooses room r_1 and a_2 chooses room r_2. If a_3 chooses room r_1, then a_1 and a_3 both prefer room r_1 more than r_2; If a_3 chooses r_2, both a_2 and a_3 prefer room r_2 to r_1. In either case, there is a room in which the two tenants do not want to switch.

□

We comment that the above dictatorship algorithm, while always preserving the room stability, does not have any competitive ratio guarantees on social welfare. It remains an open question to design an algorithm that can achieve both room stability and constant competitive ratio on social welfare. However, as we will show below, if we are willing to weaken the room stability condition, such goal indeed becomes achievable.

Definition 20. *An allocation is* weakly room-stable *if for any two agents i, i' in room r_i and two agents j, j' in another room r_j, switching their rooms cannot increase all four agents' utilities.*

Theorem 17. *There is an online algorithm that can always produce a weakly room-stable allocation with competitive ratio $c_b/8$ on social welfare, where $c_b = \frac{\ln 5 - 0.8}{5} \approx 0.1618$.*

Proof. Recall in the proof of Theorem 15, we showed

$$2\mathbb{E}[SW(M_{pr1}) + SW(M_{pb}) + SW(M_{pr2})] \geq SW(A_{opt}).$$

Note that we also have $\mathbb{E}[SW(M_{pr1})] = \mathbb{E}[SW(M_{pr2})]$. This means we can ignore $\mathbb{E}[SW(M_{pr1})$ and still get a constant competitive ratio solution. Thus we modify algorithm ONLINEROOMMATE as follows: in the first step, we easily let the first n arriving agents choose the best empty room as they want. For the next n agent we still follow algorithm ONLINEMATCHING. After this change, our new algorithm will have competitive ratio $c_b/8$. In addition, the output solution also satisfies weak room stability. This is because if we want to swap room r_i and r_j, and the first slot of r_i is assigned before r_j. Then the agent who is assigned to r_i will not want to switch because he prefers room r_i to r_j. Thus the output allocation is always weakly room-stable. □

4.3 Discussion

In this chapter, we first introduce general indivisible resource allocation problems. The existence of the polynomially computable, EF1 and Pareto optimal mechanism and the existence of an EFX allocation are two challenging open questions in this area. For the EF1 and Pareto optimal mechanism design, we show an efficient mechanism for two agents, but fail to generalize it to more agents. To approximate EFX mechanisms, we introduce a mechanism (which is not polynomially computable) that can satisfy $\frac{1}{2}$-EFX, and we believe there are better mechanisms with the better approximation ratio. For two agents, we provide an efficient $\frac{1}{2}$-EFX algorithm, and offer a possible direction for generalizing it to cases with more agents. For the chores setting, we present an EFX or proportional allocation that can be found in the polynomial time for three agents, where we are allowed to dispose of one chore. We are curious whether we can remove some of the conditions from our EFX or proportional mechanism.

There are a few directions for further research on the online roommate allocation problem. Finding a better competitive ratio is one possible direction. Although we currently have a constant competitive ratio, it could be improved, and we do not have a lower bound for the ratio. Another direction is to determine whether we can guarantee more stabilities. In the future, we may also run experiments with real data to test the performance of the mechanisms.

Chapter 5

Summary

In this thesis, we introduce many topics on divisible and indivisible resource allocation problems.

Chapter 3 introduces a set of divisible resource allocation problems. We classify them into cake cutting problems and chore division problems. We study them by restricting the valuation functions to piecewise uniform and piecewise constant functions. This chapter is divided into two parts. In the first part, we consider piecewise uniform valuations and do not adopt the free disposal assumption, which has been widely used in recent studies of truthful and fair cake cutting problems. We obtain several negative results when we add some mild assumptions (Theorem 1, 2 and 3) and show that a fair allocation can be found when the number of agents is small(Theorem 4, 5) or when more constraints are added to the valuation functions (Theorem 6, 7). In the second part of the chapter, we generalize the previous reserch [Chen *et al.*, 2013] and relax the piecewise uniform valuations to the piecewise constant valuations. We obtain some negative results when we add similar mild assumptions(Theorem 8, 9 and 10). We identify the following remaining open questions as interesting directions for further research.

- Do truthful envy-free anonymous mechanisms with piecewise uniform valuation functions for more than two agents without the free disposal assumption exist? In our study, we prove that no anonymous mechanism exists, as such a mechanism needs an infinite number of cuts (Theorem 1). However, our proof is difficult to generalize to cases with more agents, so we do not know whether there is an anonymous mechanism for even three agents. To have a complete proof, we need to prove the case for an arbitrary number of agents.

- Do truthful envy-free position oblivious mechanisms with piecewise uniform valuation functions for an odd number of agents without the free disposal assumption exist? We have shown a constructive proof that no position oblivious mechanism exists for an even number of agents (Theorem 3). In the proof, we need to partition agents into several pairs. If there are an odd number of agents, then the constructive proof no longer works. We cannot find a position oblivious mechanism for three agents, and cannot prove there is no such mechanism as well.

- Do truthful envy-free cake cutting (chore division) mechanisms with piecewise uniform valuation functions for n agents where every agent has a positive value on an interval of the cake, without the free disposal assumption, exist? In our study, we design mechanisms for a special case of this open question, i.e. the case in which valuations are prefix valuations (Theorem 6 and 7). We hope to relax this constraint to get closer to the setting of the general cake cutting problem.

- Do truthful envy-free cake cutting mechanisms with piecewise constant valuation functions for n agents exist under the free disposal assumption? For this open question, we prove some negative results with some natural assumptions added (Theorem 8, 9 and 10). Another direction is to allow the mechanism to throw away some parts of the cake and the resulting allocation should be envy-free and proportional (prevent too much cake from throwing away);

- Is there a mechanism that can approximate truthfulness and envy-freeness? Real-life application of indivisible resource allocation tasks may require us to weaken prefect fairness into "good enough" fairness. We have introduced notions such as approximately truthfulness and approximately envy-freeness, and this may be a potential direction for future research that aims to apply these mechanisms to real scenarios where reasonably good approximation of fairness are needed.

In Chapter 4, we first introduce standard indivisible resource allocation problems with the maximin share guarantee, envy-freeness up to one item (chore), envy-freeness up to the least valued item (chore) and Pareto optimality. We propose CMM-S (Theorem 11) and several mechanisms, most of them based on cases with a small number of agents (Theorem 12, 13 and 14). Then we introduce a real-life application of indivisible resource allocation, which we call the online roommate allocation problem. We present an efficient online algorithm (Theorem 15) and generalize it to many situations (Theorem 16 and Corollary 12). We consider stabilities at the same time (Theorem 17 and Lemma 13, 14). We list the open questions that we want to examine in future studies below.

- Do polynomial-time algorithms to find EF1 and Pareto optimal allocations exist? Currently, we know that the allocation that maximizes Nash social welfare is EF1 and Pareto optimal, but no polynomial-time mechanism can be found. This could potentially be applied in many real-life situations.

- Do EFX allocations for goods or chores exist? EFX represents strong fairness, but we can neither find an EFX mechanism nor prove it does not exist for either the goods or chores settings. It is an open question, even for cases with only three agents.

- Is there a better competitive ratio and achievable stability in the online roommate allocation problem? Our polynomial-time online algorithm is not proved to have a tight constant competitive ratio, so it is possible that there is a better algorithm for this problem.

List of work Below is a list of works that have been done during my Ph.D. study.

- Cake Cutting: Envy and Truth[1], with Xiaohui Bei, Ning Chen, Biaoshuai Tao, and Jiajun Wu.
- Online Roommate Allocation Problem[2], with Xin Huang, Shengyu Zhang, and Xiaohui Bei.
- Truthful Fair Division without Free Disposal[3], with Xiaohui Bei and Warut Suksompong.

[1][Bei *et al.*, 2017a], copyright holder IJCAI's link: http://www.ijcai.org/

[2][Huzhang *et al.*, 2017], copyright holder IJCAI's link: http://www.ijcai.org/

[3][Bei *et al.*, 2018] copyright holder IJCAI's link: http://www.ijcai.org/

Bibliography

[Aggarwal *et al.*, 2011] Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1253–1264. SIAM, 2011.

[Alijani *et al.*, 2017] Reza Alijani, Majid Farhadi, Mohammad Ghodsi, Masoud Seddighin, and Ahmad S Tajik. Envy-free mechanisms with minimum number of cuts. In *AAAI*, pages 312–318, 2017.

[Amanatidis *et al.*, 2017] Georgios Amanatidis, Evangelos Markakis, Afshin Nikzad, and Amin Saberi. Approximation algorithms for computing maximin share allocations. *ACM Transactions on Algorithms (TALG)*, 13(4):52, 2017.

[Austin, 1982] AK Austin. Sharing a cake. *The Mathematical Gazette*, 66(437):212–215, 1982.

[Aziz and Mackenzie, 2016a] Haris Aziz and Simon Mackenzie. A discrete and bounded envy-free cake cutting protocol for any number of agents. *FOCS*, 2016.

[Aziz and Mackenzie, 2016b] Haris Aziz and Simon Mackenzie. A discrete and bounded envy-free cake cutting protocol for four agents. *STOC*, 2016.

[Aziz and Ye, 2014] Haris Aziz and Chun Ye. Cake cutting algorithms for piecewise constant and piecewise uniform valuations. In *WINE*, 2014.

[Bei *et al.*, 2012] Xiaohui Bei, Ning Chen, Xia Hua, Biaoshuai Tao, and Endong Yang. Optimal proportional cake cutting with connected pieces. In *AAAI*, 2012.

[Bei *et al.*, 2017a] Xiaohui Bei, Ning Chen, Guangda Huzhang, Biaoshuai Tao, and Jiajun Wu. Cake cutting: envy and truth. In *IJCAI*, pages 3625–3631. AAAI Press, 2017.

[Bei *et al.*, 2017b] Xiaohui Bei, Youming Qiao, and Shengyu Zhang. Networked fairness in cake cutting. 2017.

[Bei *et al.*, 2018] Xiaohui Bei, Guangda Huzhang, and Warut Suksompong. Truthful fair division without free disposal. In *IJCAI*, 2018.

[Biró *et al.*, 2010] Péter Biró, Robert W Irving, and David F Manlove. Popular matchings in the marriage and roommates problems. In *Algorithms and Complexity*, pages 97–108. Springer, 2010.

[Bouveret *et al.*, 2017] Sylvain Bouveret, Katarína Cechlárová, Edith Elkind, Ayumi Igarashi, and Dominik Peters. Fair division of a graph. In *IJCAI*, 2017.

[Brams and Taylor, 1995] Steven J Brams and Alan D Taylor. An envy-free cake division protocol. *American Mathematical Monthly*, 102(1), 1995.

[Brams and Taylor, 1996] Steven J Brams and Alan D Taylor. *Fair Division: From cake-cutting to dispute resolution*. Cambridge University Press, 1996.

[Budish and Cantillon, 2012] Eric Budish and Estelle Cantillon. The multi-unit assignment problem: Theory and evidence from course allocation at harvard. *American Economic Review*, 102(5):2237–71, 2012.

[Budish, 2011] Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.

[Burk, 1938] Abram Burk. A reformulation of certain aspects of welfare economics. *The Quarterly Journal of Economics*, 52(2):310–334, 1938.

[Caragiannis *et al.*, 2016] Ioannis Caragiannis, David Kurokawa, Hervé Moulin, Ariel D Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum nash welfare. In *Proceedings of the 2016 ACM Conference on Economics and Computation*. ACM, 2016.

[Cechlárová and Manlove, 2005] Katarína Cechlárová and David F Manlove. The exchange-stable marriage problem. *Discrete Applied Mathematics*, 152(1):109–122, 2005.

[Chan *et al.*, 2016] Pak Hay Chan, Xin Huang, Zhengyang Liu, Chihao Zhang, and Shengyu Zhang. Assignment and pricing in roommate market. In *Thirtieth AAAI Conference on Artificial Intelligence*, pages 446–452, 2016.

[Chen *et al.*, 2013] Yiling Chen, John Lai, David Parkes, and Ariel D. Procaccia. Truth, justice, and cake cutting. *Games and Economic Behavior*, 77:284–297, 2013.

[Eriksson *et al.*, 2006] Kimmo Eriksson, Jonas Sjöstrand, and Pontus Strimling. Three-dimensional stable matching with cyclic preferences. *Mathematical Social Sciences*, 52(1):77–87, 2006.

[Ferguson and others, 1989] Thomas S Ferguson et al. Who solved the secretary problem? *Statistical science*, pages 282–289, 1989.

[Foley, 1967] Duncan Foley. Resource allocation and the public sector. *Yale Econ Essays*, 7(1):45–98, 1967.

[Gale and Shapley, 1962] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *American mathematical monthly*, pages 9–15, 1962.

[Gamow and Stern, 1958] George Gamow and Marvin Stern. *Puzzle-math*. Viking Press, 1958.

[Goldman and Procaccia, 2015] Jonathan Goldman and Ariel D Procaccia. Spliddit: Unleashing fair division algorithms. *ACM SIGecom Exchanges*, 13(2):41–46, 2015.

[Graham *et al.*, 1979] Ronald L Graham, Eugene L Lawler, Jan Karel Lenstra, and AHG Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. In *Annals of discrete mathematics*, volume 5, pages 287–326. Elsevier, 1979.

[Huang, 2007] Chien-Chung Huang. Two's company, three'sa crowd: Stable family and threesome roommates problems. In *Algorithms–ESA 2007*, pages 558–569. Springer, 2007.

[Hurwicz, 1972] Leonid Hurwicz. On informationally decentralized systems. *Decision and organization*, 1972.

[Huzhang *et al.*, 2017] Guangda Huzhang, Xin Huang, Shengyu Zhang, and Xiaohui Bei. Online roommate allocation problem. In *IJCAI*, 2017.

[Irving and Manlove, 2002] Robert W Irving and David F Manlove. The stable roommates problem with ties. *Journal of Algorithms*, 43(1):85–105, 2002.

[Irving, 1985] Robert W Irving. An efficient algorithm for the "stable roommates" problem. *Journal of Algorithms*, 6(4):577–595, 1985.

[Iwama and Miyazaki, 2008] Kazuo Iwama and Shuichi Miyazaki. A survey of the stable marriage problem and its variants. In *Informatics Education and Research for Knowledge-Circulating Society, 2008. ICKS 2008. International Conference on*, pages 131–136. IEEE, 2008.

[Kaneko and Nakamura, 1979] Mamoru Kaneko and Kenjiro Nakamura. The nash social welfare function. *Econometrica: Journal of the Econometric Society*, pages 423–435, 1979.

[Karp *et al.*, 1990] Richard M Karp, Umesh V Vazirani, and Vijay V Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 352–358. ACM, 1990.

[Kesselheim *et al.*, 2013] Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In *Algorithms–ESA 2013*, pages 589–600. Springer, 2013.

[Kesselheim *et al.*, 2014] Thomas Kesselheim, Andreas Tönnis, Klaus Radke, and Berthold Vöcking. Primal beats dual on online packing lps in the random-order model. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 303–312. ACM, 2014.

[Knuth, 1997] Donald Ervin Knuth. *Stable marriage and its relation to other combinatorial problems: An introduction to the mathematical analysis of algorithms*, volume 10. American Mathematical Soc., 1997.

[Kurokawa *et al.*, 2013] David Kurokawa, John K Lai, and Ariel D Procaccia. How to cut a cake before the party ends. In *AAAI*, 2013.

[Maya and Nisan, 2012] Avishay Maya and Noam Nisan. Incentive compatible two player cake cutting. In *WINE*, 2012.

[Menon and Larson, 2017] Vijay Menon and Kate Larson. Deterministic, strategyproof, and fair cake cutting. In *IJCAI*, 2017.

[Ng and Hirschberg, 1991] Cheng Ng and Daniel S Hirschberg. Three-dimensional stable matching problems. *SIAM Journal on Discrete Mathematics*, 4(2):245–252, 1991.

[Plaut and Roughgarden, 2018] Benjamin Plaut and Tim Roughgarden. Almost envy-freeness with general valuations. In *SIAM*, 2018.

[Procaccia and Wang, 2014] Ariel D Procaccia and Junxing Wang. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the 2014 ACM conference on Economics and computation*. ACM, 2014.

[Procaccia, 2013] Ariel D Procaccia. Cake cutting: Not just child's play. *Communications of the ACM*, 2013.

[Rais and Viana, 2011] Abdur Rais and Ana Viana. Operations research in healthcare: a survey. *International transactions in operational research*, 18(1):1–31, 2011.

[Robertson and Webb, 1998] Jack Robertson and William Webb. *Cake-cutting algorithms: Be fair if you can*. AK Peters/CRC Press, 1998.

[Schummer, 1996] James Schummer. Strategy-proofness versus efficiency on restricted domains of exchange economies. *Social Choice and Welfare*, 14(1), 1996.

[Steinhaus, 1948] Hugo Steinhaus. The problem of fair division. *Econometrica*, 16(1):101–104, 1948.

[Varian, 1974] Hal R Varian. Equity, envy, and efficiency. *Journal of economic theory*, 9(1):63–91, 1974.