Robust Rent Division

Anonymous Author(s)
Affiliation
Address
email

Abstract

In fair rent division, the problem is to assign rooms to roommates and fairly split the rent based on roommates' reported valuations for the rooms. Envy-free rent division is the most popular application on the fair division website Spliddit. The standard model assumes that agents can correctly report their valuations for each room. In practice, agents may be unsure about their valuations, for example because they have had only limited time to inspect the rooms. Our goal is to find a robust rent division that remains fair even if agent valuations are slightly different from the reported ones. We introduce the lexislack solution, which selects a rent division that remains envy-free for valuations within as large a radius as possible of the reported valuations. We also consider robustness notions for valuations that come from a probability distribution, and use results from learning theory to show how we can find rent divisions that (almost) maximize the probability of being envy-free, or that minimize the expected envy. We show that an almost optimal allocation can be identified based on polynomially many samples from the valuation distribution. Finding the best allocation given these samples is NP-hard, but in practice such an allocation can be found using integer linear programming.

1 Introduction

The literature on fair division of resources has produced allocation mechanisms for many domains, such as course allocation, indivisible goods, chores, house assignment, and the selection of citizens' assemblies [Budish, 2011, Caragiannis et al., 2019, Moulin, 2019, Flanigan et al., 2021]. But arguably the most widely *used* example is *rent division*: this is the most popular application on the fair division website spliddit.org [Goldman and Procaccia, 2014], where it has been used more than 30,000 times since its launch in 2014.

Rent division deals with the common situation where a group of n future roommates are planning to move into a house or apartment which has n rooms, one for each roommate. They will split the rent payments among themselves. The roommates may differ in how much they are willing to pay for different rooms. Given the room valuations of each roommate, our task is to assign the rooms, and to decide how to split the rent. We wish to do this fairly, and so we will choose an allocation that is *envy-free*: no roommate would strictly prefer to get another room, given the prices we have assigned to those rooms. Such an allocation is guaranteed to exist [Svensson, 1983].

Let us consider an example with $n = 3$ roommates, and let the total rent be \$1000. Table 1 shows the valuation that each agent assigns to each room. Given this information, the algorithm in use on Spliddit will assign Room 1 to Alice, Room 2 to Bob, and Room 3 to Charlie, charging them \$100, \$500, and \$400 respectively. This allocation is envy-free under the assumption (which we will make throughout) that agents have *quasilinear utilities*: their utility under an allocation is the value of their room minus its price. For example, Alice has utility $300 - 100 = 200$. She does not envy the others: Bob's room would give her only utility $400 - 500 < 200$, and Charlie's room $300 - 400 < 200$.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

	Room 1	Room 2	Room 3
Alice	**300**	400	300
Bob	300	**700**	0
Charlie	300	100	**600**
Spliddit	100	500	400
Lexislack	200	450	350

Table 1: Example of valuations. In any envy-free allocation, Alice gets room 1, Bob gets room 2, and Charlie gets room 3. The lower rows display the price vectors selected by Spliddit's rule (maximin) and by our lexislack rule.

On a typical instance, there are infinitely many allocations that are envy-free. Spliddit's algorithm chooses the one that maximizes the utility of the worst-off agent, subject to envy-freeness [Gal et al., 2017]. This is known as the *maximin* rule. In optimizing this objective, Spliddit might choose an outcome that is only barely envy-free. In the example, Bob has utility $700 - 500 = 200$, but he would gain the same utility from having Alice's room: $300 - 100$. If, upon moving in, Bob discovers a defect in his room and now only values it at 600, say, then he would envy Alice. Thus, the envy-freeness of Spliddit's allocation is not robust.

We study the rent division problem with the goal of finding allocations that are robustly envy-free, in the sense that they remain envy-free even if valuations change slightly. For this, we introduce the *lexislack rule*, which selects an envy-free allocation where the minimum "slack" (the amount by which agent i prefers her allocation to agent j's) is maximized lexicographically. This produces an allocation that remains envy-free for all valuation profiles that are within a maximally large ℓ_1-radius of the reported profile. In the example of Table 1, the lexislack rule assigns the rooms in the same way as does Spliddit, but charges the roommates \$200, \$450, and \$350. One can verify that with these prices, each agent prefers their allocation to any other agent's by at least 150. This means that even after Bob's adjustment to 600, he does not envy Alice. We show that the lexislack rule always selects an essentially unique outcome, which can be found in polynomial time by linear programming.

This notion of robustness may not always be appropriate. Consider two perturbations with equal ℓ_1-distance to the reported valuations: one changes agent i's valuations for all rooms by a small amount, the other changes i's valuation for one room by a large amount. Lexislack places equal importance on them. But the former perturbation seems more likely: even if a player is uncertain about the value of a room, that value is more likely to be close to their best estimate than further away. Thus, arguably, different valuation profiles should be weighted differently: we do not want to sacrifice envy-freeness for a likely perturbation in order to obtain it for an unlikely perturbation.

To capture this idea, we propose to add noise — such as Gaussian noise — around the reported valuations. This way, we impute a probability distribution $\mathcal{D}$ over valuations. In this setting, our interpretation of robustness is to look for allocations that are envy-free with *maximum probability*. However, it is not clear how one could efficiently find the most robust allocation given the noisy valuations. As part of our methodological contribution, we propose an approach based on synthetic sampling. Specifically, we sample a number of valuation profiles from $\mathcal{D}$, and then find an allocation that is optimal on this sample using integer linear programming (ILP). By calculating the VC dimension of the space of rent divisions, we give polynomial sample complexity bounds that show how many samples are sufficient so that this approach identifies an almost optimal allocation with high probability. Note that the samples are synthetic, but low sample complexity is crucial nevertheless: a small number of samples leads to a sufficiently small ILP that, as we show, can be optimally solved in practice (despite the NP-completeness of the problem, which we establish).

We also show that one can use the sampling approach to find an allocation that minimizes the expected amount by which one agent envies another. In contrast to maximizing the probability of envy-freeness, the minimum envy objective places more emphasis on avoiding bad violations of envy-freeness.

An advantage of our sampling-based approach is that it is very general and does not place any restrictions on the distribution $\mathcal{D}$. In particular, our algorithms could also be used for rent division problems with uncertainty, where agents might explicitly report distributions over their valuations. For example, a simple Spliddit-like user interface could allow agents to report their room valuations as a *range* rather than a number.

We end with some experiments on data taken from Spliddit. They suggest that our three new rules significantly outperform the Spliddit maximin rule on robustness metrics. Interestingly, the lexislack solution does comparably well to the rules based on sampling. Given its conceptual simplicity and easy computation, this suggests lexislack as a good rule when robustness is desired.

Related Work

The rent division model is well-studied in the economics literature [Svensson, 1983, Alkan et al., 1991, Aragones, 1995, Su, 1999, Velez, 2018], often without assuming quasilinear utilities. That literature includes results on the structure of the envy-free set and about strategic aspects. Computer scientists have studied the computation of allocation rules [Gal et al., 2017, Procaccia et al., 2018]. Bei et al. [2021] study a generalization of the rent division problem.

Robustness has been studied in several areas of computational social choice, such as in voting [Shiryaev et al., 2013], in committee elections [Bredereck et al., 2021, Gawron and Faliszewski, 2019, Misra and Sonar, 2019], and in stable matching [Chen et al., 2019, Mai and Vazirani, 2018]. We are not aware of such work for fair division, though Menon and Larson [2020] study a related problem of "stability" which requires that the allocation should not change much if valuations change slightly. For rent division, a blog post by Critch [2015] argues in favor of aiming for robustness in the rent division problem. Critch [2015] implemented an algorithm for robust rent division that appears in experiments to maximize the slack, but it differs from the lexislack rule, and no theoretical analysis of this algorithm is available.

Our sampling-based approach is conceptually related to work on data-driven algorithm design [Balcan, 2020], which typically seeks to optimize the hyperparameters of an algorithm with respect to an underlying distribution over instances, based on samples. One thing that distinguishes our distributional setting is that we are using the samples to optimize a single solution to our problem.

2 Preliminaries: Rent Division

Let $n \in \mathbb{N}$ and write $[n] = \{1, \ldots, n\}$. Let $N = [n]$ be a set of n agents, and let $R = [n]$ be a set of n rooms. Without loss of generality, we let the total rent be 1. A *(valuation) profile* $v = (v_{ir})_{i \in N, r \in R}$ is a collection of values $v_{ir} \in \mathbb{Q}_+$, one for each agent $i \in N$ and each room $r \in R$.

A *room assignment* is a bijection $\sigma : N \to R$, so that agent i is assigned room $\sigma(i)$. Given a valuation profile v, we say that σ is *optimal* if it maximizes utilitarian social welfare $\sum_{i \in N} v_{i\sigma(i)}$. An *allocation* (σ, p) is a room assignment σ together with a payment vector $p = (p_1, \ldots, p_n) \in \mathbb{R}^n$ with $\sum_{r \in R} p_r = 1$, where p_r is the rent of room r. (The value p_r is usually non-negative.)

We assume that agents have *quasilinear utilities*. This means that if v is a valuation profile and (σ, p) is an allocation, then agent i's utility in this allocation is $v_{i\sigma(i)} - p_{\sigma(i)}$, i.e., the valuation of i for her room $\sigma(i)$ minus the room's rent. An allocation (σ, p) is *envy-free* if $v_{i\sigma(i)} - p_{\sigma(i)} \geqslant v_{ir} - p_r$ for all $i \in N$ and $r \in R$, so that each agent i weakly prefers her allocation to receiving any other room.

A *solution* is a function that given a valuation profile, selects a set of allocations (usually a singleton, but ties may occur). A solution is *essentially single-valued* if when it selects more than one allocation, then all agents are indifferent between them: every agent gets the same utility from all tied allocations.

The following facts are well-known [see, e.g., Velez, 2018]. We include proofs in the appendix.

Theorem 2.1. *(a) For every optimal room assignment σ, there are prices p so that (σ, p) is envy-free.*
(b) If (σ, p) is envy-free then σ is optimal.
(c) Let σ_1, σ_2 be optimal room assignments, and let (σ_1, p) be an envy-free allocation. Then (σ_2, p) is also an envy-free allocation, with all agents indifferent between the two: $v_{i\sigma_1(i)} - p_{\sigma_1(i)} = v_{i\sigma_2(i)} - p_{\sigma_2(i)}$ for all $i \in N$.

Theorem 2.1(a) implies that an envy-free allocation exists for all valuation profiles. We can compute one in polynomial time: find an optimal room assignment σ using bipartite matching, then use linear programming to find prices p that make the allocation envy-free [Gal et al., 2017]. Theorem 2.1(c) implies that when selecting among envy-free allocations, we can restrict attention to any fixed σ and only consider allocations of the form (σ, p). By Theorem 2.1(c), all utility vectors achievable in an envy-free allocation are achieved by allocations of this form.

3 The Lexislack Solution

We start by considering a common form of robustness: we look for allocations that remain fair for all valuations that are within some radius of input valuations, for as large a radius as possible. Thus, unlike in later sections, we do not assume that valuations come from a probability distribution.

Let v be a valuation profile, fixed throughout. Let (σ, p) be an allocation. For $i \in N$ and $r \in R$, let

$$\Delta_{ir}(\sigma, p) = (v_{i\sigma(i)} - p_{\sigma(i)}) - (v_{ir} - p_r).$$

Then define the *slack* of this allocation as

$$\mathsf{slack}(\sigma, p) = \min_{i \in N} \min_{r \neq \sigma(i)} \Delta_{ir}.$$

Thus, an allocation has positive slack if every agent strictly prefers their allocation to all other agents' allocations. An allocation (σ, p) is envy-free if and only if $\mathsf{slack}(\sigma, p) \geqslant 0$.

Slack is a measure of how robustly fair an allocation is, which we formalize in the following result.

Proposition 3.1. *Let (σ, p) be an envy-free allocation with $\mathsf{slack}(\sigma, p) = s \geqslant 0$. If v' is a valuation profile that is s-close to v in the sense that*

$$\|v_i - v'_i\|_1 = \textstyle\sum_{r \in R} |v_{ir} - v'_{ir}| \leqslant s$$

for all $i \in N$, then (σ, p) is also envy-free under v'.

Proof. Let $i, j \in N$. Then $\sum_{r \in R} |v_{ir} - v'_{ir}| \leqslant s$ implies

$$(v_{i\sigma(i)} - v'_{i\sigma(i)}) + (v_{i\sigma(j)} - v'_{i\sigma(j)}) \leqslant s \tag{1}$$

Adding $p_{\sigma(j)} - p_{\sigma(i)}$ to both sides and rearranging, we get

$$(v'_{i\sigma(i)} - p_{\sigma(i)}) - (v'_{i\sigma(j)} - p_{\sigma(j)}) \geqslant (v_{i\sigma(i)} - p_{\sigma(i)}) - (v_{i\sigma(j)} - p_{\sigma(j)}) - s \geqslant 0$$

by definition of slack. Thus, i does not envy j under v'. □

One can also prove variants of Proposition 3.1. For example, $\|v_i - v'_i\|_\infty \leqslant s/2$ also implies (1).

If we wish to ensure robustness in a sense like Proposition 3.1, this suggests the following rule:

$$\mathsf{maxislack}(v) = \mathrm{argmax}_{(\sigma,p)}\, \mathsf{slack}(\sigma, p).$$

This rule always selects an envy-free allocation: since envy-free allocations exist for every v, there exists an allocation with non-negative slack, and hence the maxislack solution also has non-negative slack. A maxislack solution can be found in polynomial time by computing an optimal assignment σ and then solving the following LP:

$$\max L \text{ subject to } (v_{i\sigma(i)} - p_{\sigma(i)}) - (v_{i\sigma(j)} - p_{\sigma(j)}) \geqslant L \ \forall i \neq j \text{ and } \textstyle\sum_{r \in R} p_r = 1.$$

However, there are a few drawbacks to the maxislack rule. First, the rule is not essentially single-valued: there may be several maxislack allocations which induce different utilities. This is unlike Spliddit's maximin rule which is essentially single-valued [Alkan et al., 1991]. Second, there may be maxislack allocations that do not maximize robustness for *all* agents. To see this, suppose that two agents i_1 and i_2 agree on the valuation of every room. Then in any envy-free allocation, the utility they assign to the two bundles allocated to them is equal. Hence the maximum slack attainable is 0, and so *every* envy-free allocation is maxislack. However, there may be allocations for which the slack between other pairs of agents is larger than 0, and such allocations are more robustly fair.

In this spirit, to obtain robustness for a larger collection of agents (or of agent pairs), we can refine the maxislack solution using a leximin strategy. We call the resulting solution the *lexislack rule*. Write $\Delta(\sigma, p) = (\Delta_{ir}(\sigma, p))_{i \in N, r \in R}$ for the vector of slacks. The lexislack rule selects an allocation (σ, p) that maximizes the smallest of the n^2 values in $\Delta(\sigma, p)$, and subject to that maximizes the second-smallest of these values, and so on.

In contrast to maxislack, the lexislack rule is essentially single-valued. The proof is in the appendix.

Theorem 3.2. *The lexislack rule is essentially single-valued.*

In addition, this rule remains efficiently computable, as we show in the appendix.

Theorem 3.3. *A lexislack allocation can be found in polynomial time by solving $O(n^4)$ linear programs.*

4 Maximizing Probability of Envy-Freeness

In the previous section, we defined robustness using a measure of closeness based on the ℓ_1-distance. We now look at a more flexible model where true valuations are assumed to be noisy perturbations of the reported ones. A distribution $\mathcal{D}$ over valuations v, therefore, is obtained by asking agents for valuations, and then adding noise (e.g., Gaussian or uniform) around those valuations. Our goal will be to find an allocation (σ, p) that maximizes the probability of being envy-free with respect to $\mathcal{D}$, i.e., one that maximizes

$$\mathsf{EFrate}_{\mathcal{D}}(\sigma, p) = \Pr_{v \sim \mathcal{D}}[(\sigma, p) \text{ is envy-free under } v].$$

Our algorithmic approach for finding an allocation with high probability of envy-freeness is to obtain a sample S of m valuation profiles sampled from $\mathcal{D}$, and to compute an allocation that is envy-free on the most profiles in S, i.e., one that maximizes

$$\mathsf{EFrate}_{S}(\sigma, p) = \tfrac{1}{m} \cdot |\{v \in S : (\sigma, p) \text{ is envy-free under } v\}|.$$

If the number m of samples is sufficiently high, we may hope that the best allocation on the sample S is also approximately the best on the distribution $\mathcal{D}$. In this section, we will give a bound for the sample size m to be sufficient to ensure this property, and then we will discuss the computational problem of finding the best allocation for a given sample.

4.1 Sample Complexity

In this section, we will prove the following result.

Theorem 4.1. *Let $\varepsilon, \delta > 0$. There is a value $m \in \mathbb{N}$ with $m = O\left(\frac{n^2 \log n + \log(1/\delta)}{\varepsilon^2}\right)$ such that for every distribution $\mathcal{D}$ over valuation profiles, if S is a collection of at least m samples drawn i.i.d. from $\mathcal{D}$, and (σ^*, p^*) is the allocation that maximizes EFrate_S, then with probability at least $1 - \delta$,*

$$\mathsf{EFrate}_{\mathcal{D}}(\sigma^*, p^*) \geqslant \max_{(\sigma,p)} \mathsf{EFrate}_{\mathcal{D}}(\sigma, p) - \varepsilon.$$

We prove this theorem by adapting standard tools from learning theory. Let X be any set, with an unknown ground truth labeling $\tau : X \to \{0, 1\}$. A *hypothesis* is a function $h : X \to \{0, 1\}$. Given a *sample* $S = (x_1, \ldots, x_m)$ of m elements of X (not necessarily distinct), write $\mathsf{err}_S(h) = \frac{1}{m}|\{x_i : h(x_i) \neq \tau(x_i)\}|$ for the fraction of samples that h labeled incorrectly. For a probability distribution $\mathcal{D}$ over X, write $\mathsf{err}_{\mathcal{D}}(h) = \Pr_{x \sim D}[h(x) \neq \tau(x)]$ for the probability that h incorrectly labels a point sampled from $\mathcal{D}$.

A *hypothesis class* $\mathcal{H}$ is a set of hypotheses. Given a random sample S drawn i.i.d. from $\mathcal{D}$, and knowledge of the true labeling τ of those samples, our goal is to find a hypothesis $h \in H$ that approximately minimizes $\mathsf{err}_{\mathcal{D}}(h)$, with high probability. Note that the ground truth τ, interpreted as a hypothesis, need not be a member of $\mathcal{H}$. In learning theory, this setup corresponds to "agnostic PAC learning", where the realizability assumption need not hold.

We say that a set $C \subseteq X$ is *shattered* by $\mathcal{H}$ if for all $S \subseteq C$, there exists $h \in \mathcal{H}$ with $h(x) = 1$ if $x \in S$ and $h(x) = 0$ if $x \in C \setminus S$. In other words, if we restrict the hypotheses in $\mathcal{H}$ to the set C, then all possible labelings of C are part of $\mathcal{H}$. The *VC dimension* $\mathsf{VCdim}(\mathcal{H})$ of $\mathcal{H}$ is the cardinality of the largest subset of X that is shattered by $\mathcal{H}$. We are interested in VC dimension due to the following standard result, adapted from Shalev-Shwartz and Ben-David [2014, Theorem 6.8], which says that PAC learning is possible on hypothesis classes of finite VC dimension.

Theorem 4.2. *Let $\varepsilon, \delta > 0$. Let $\mathcal{H}$ be a hypothesis class with $\mathsf{VCdim}(\mathcal{H}) = d$. Then there exists a value $m \in \mathbb{N}$ with $m = O\left(\frac{d + \log(1/\delta)}{\varepsilon^2}\right)$ such that for every probability distribution $\mathcal{D}$ over X, if S is a collection of at least m samples drawn i.i.d. from $\mathcal{D}$, and $h^* \in H$ is the hypothesis that minimizes err_S, then with probability at least $1 - \delta$,*

$$\mathsf{err}_{\mathcal{D}}(h^*) \leqslant \min_{h \in \mathcal{H}} \mathsf{err}_{\mathcal{D}}(h) + \varepsilon.$$

For our application, we let X be the set of all valuation profiles v. The "correct" labeling is $\tau(v) = 1$ for all v. We identify allocations with hypotheses: For an allocation (σ, p), we define the hypothesis $h_{(\sigma,p)}$ so that for each v,

$$h_{(\sigma,p)}(v) = \begin{cases} 1 & \text{if } (\sigma, p) \text{ is envy-free under } v, \\ 0 & \text{otherwise.} \end{cases}$$

By these definitions, we have that for all S and $\mathcal{D}$,

$$\mathsf{EFrate}_S(\sigma, p) = 1 - \mathsf{err}_S(h_{(\sigma,p)}) \quad \text{and} \quad \mathsf{EFrate}_{\mathcal{D}}(\sigma, p) = 1 - \mathsf{err}_{\mathcal{D}}(h_{(\sigma,p)}).$$

We study the hypothesis class $\mathcal{H}$ of all such hypotheses:

$$\mathcal{H} = \{h_{(\sigma,p)} : \text{allocations } (\sigma, p)\}.$$

To bound its VC dimension, the following result is useful:

Lemma 4.3 (Shalev-Shwartz and Ben-David, 2014, Exercise 6.11)**.** *Let $\mathcal{H}_1, \ldots, \mathcal{H}_t$ be hypothesis classes over X, with $\mathsf{VCdim}(\mathcal{H}_i) \leqslant d$ for each $i = 1, \ldots, t$. Then*

$$\mathsf{VCdim}(\mathcal{H}_1 \cup \cdots \cup \mathcal{H}_t) \leqslant 4d \log(2d) + 2\log(t).$$

We can now bound the VC dimension of $\mathcal{H}$.

Lemma 4.4. $\mathsf{VCdim}(\mathcal{H}) = O(n^2 \log n)$.

Proof. For each room assignment σ, define the hypothesis class $\mathcal{H}_\sigma = \{h_{(\sigma,p)} : p \in \mathbb{R}^n\}$ corresponding to allocations whose room assignment is σ. Then $\mathcal{H} = \bigcup_\sigma \mathcal{H}_\sigma$ where the union ranges over all room assignments. We will show that $\mathsf{VCdim}(\mathcal{H}_\sigma) \leqslant n^2$ for each σ. Since there are $n!$ different room assignments and $\log n! = O(n \log n)$, it follows from Lemma 4.3 that $\mathsf{VCdim}(\mathcal{H}) = O(n^2 \log n)$, as required.

Let σ be a room assignment. Without loss of generality assume that $\sigma(i) = i$. Let $d \geqslant n^2 + 1$. Consider a collection of d distinct valuation profiles $v^{(1)}, \ldots, v^{(d)}$. We show that this collection cannot be shattered by $\mathcal{H}_\sigma$.

For $i, j \in N$, say $v^{(k)}$ is *uniquely restricting for* (i, j) if

$$v_{ij}^{(k)} - v_{ii}^{(k)} > v_{ij}^{(\ell)} - v_{ii}^{(\ell)} \quad \text{for all } \ell \neq k.$$

Thus, such a profile uniquely maximizes the amount by which agent i prefers j's room to her own room, ignoring prices. Clearly, for any pair $i, j \in N$, at most one profile can be uniquely restricting for it. Since there are n^2 many pairs (i, j) and $d > n^2$, there is at least one profile which is not uniquely restricting for any pair, say $v^{(1)}$.

We now ask if there is an allocation (σ, p) that is envy-free under $v^{(2)}, \ldots, v^{(d)}$, but not envy-free under $v^{(1)}$. We show that the answer is no, so $\mathcal{H}_\sigma$ fails to shatter this collection.

Assume for a contradiction that (σ, p) is such an allocation. Since it is not envy-free under $v^{(1)}$, there is a pair $i, j \in N$ with $v_{ij}^{(1)} - p_j > v_{ii}^{(1)} - p_i$ or equivalently

$$v_{ij}^{(1)} - v_{ii}^{(1)} > p_j - p_i. \tag{2}$$

As $v^{(1)}$ is not uniquely restricting for (i, j), for some $\ell \neq 1$,

$$v_{ij}^{(\ell)} - v_{ii}^{(\ell)} \geqslant v_{ij}^{(1)} - v_{ii}^{(1)}. \tag{3}$$

Combining (2) and (3), it follows that $v_{ij}^{(\ell)} - v_{ii}^{(\ell)} > p_j - p_i$. Thus, (σ, p) is not envy-free under $v^{(\ell)}$, a contradiction. □

Our main result in this section, Theorem 4.1, now follows immediately from Theorem 4.2.

4.2 Computational Complexity

To make use of Theorem 4.1, we need an algorithm that, given a collection $S = (v^{(1)}, \ldots, v^{(m)})$ of valuation profiles sampled from $\mathcal{D}$, finds an allocation that maximizes $\mathsf{EFrate}_S(\sigma, p)$. This problem can be encoded as an integer linear program via standard encoding techniques, using binary variables x_{ir} encoding that agent i receives room r, continuous variables p_r encoding the prices, and a binary variable y_ℓ for each sample $\ell \in [m]$, indicating whether the produced allocation will be envy-free under $v^{(\ell)}$. The full encoding appears in the appendix.

Instead of an ILP approach, can we hope for a polynomial time algorithm finding the best allocation? Let us formulate our optimization problem as a decision problem as follows.

EF-RATE MAXIMIZATION
Input: Set N of agents, set R of rooms, a list of m valuation profiles $v^{(1)}, \ldots, v^{(m)}$, number B.
Question: Does there exist an allocation that is envy-free for at least B of the m valuation profiles?

Unfortunately, this problem is computationally hard.

Theorem 4.5. EF-RATE MAXIMIZATION *is NP-complete.*

We prove this by a reduction from CLIQUE.

There are two sources of computational difficulty for solving EF-RATE MAXIMIZATION: we have to decide on one of the $n!$ possible room assignments, and we have to decide on which subset of valuation profiles we are aiming to be envy-free on. But in practice, there is a way to avoid the first source of hardness. Suppose the m valuation profiles are sampled from a *continuous* distribution $\mathcal{D}$. Then with probability 1, for each sampled profile $v^{(\ell)}$ there is a *unique* optimal room assignment $\sigma^{(\ell)}$. Any solution to the EF-rate maximization problem must use a room assignment that is optimal for at least one of the given valuation profiles. Thus, at most m different room assignments are candidates, and we can find an optimal solution using m calls to the following problem:

EF-RATE MAXIMIZATION WITH FIXED ASSIGNMENT
Input: A list of m valuation profiles $v^{(1)}, \ldots, v^{(m)}$, number B, room assignment σ.
Question: Is there a price vector p such that (σ, p) is envy-free for at least B of the m valuation profiles?

Unfortunately, this version of the problem is also hard, and so this trick for continuous distributions does not help. We prove this by reduction from the feedback arc set problem. The proof is in the appendix.

Theorem 4.6. EF-RATE MAXIMIZATION WITH FIXED ASSIGNMENT *is NP-complete.*

Nevertheless, as we show in Section 6, we can solve this problem in practice using integer linear programming (ILP). The reason this is possible is that the sample complexity is relatively low, leading to an ILP of practical size.

5 Minimizing Expected Envy

In Section 4, we defined robust envy-freeness as allocations that have a high probability of being envy-free when valuations come from a given distribution $\mathcal{D}$. In this section, we consider a different objective function that is more fine-grained. In measuring the probability of envy-freeness, we implicitly treat all failures of envy-freeness equally. We will now minimize *expected envy*, which treats cases where one agent envies another by a lot as more severe.

Given a valuation profile v and an allocation (σ, p), we define the allocation's *(maximum) envy*, $\mathsf{envy}_v(\sigma, p)$, to be

$$\max\left\{0, \max_{i,j \in N} \left[(v_{i\sigma(j)} - p_{\sigma(j)}) - (v_{i\sigma(i)} - p_{\sigma(i)})\right]\right\}.$$

This quantity, which is related to slack as considered in Section 3, measures the biggest amount by which one agent prefers another's bundle. In principle one could allow negative values of $\mathsf{envy}_v(\sigma, p)$ for allocations that have positive slack, but we chose to force these values to be non-negative, since our focus is on avoiding envy. Note that an allocation is envy-free if and only if $\mathsf{envy}_v(\sigma, p) = 0$.

Our goal is to find an allocation minimizing the expected envy with respect to $\mathcal{D}$, defined as

$$\mathsf{envy}_{\mathcal{D}}(\sigma, p) = \mathbb{E}_{v \sim \mathcal{D}}[\mathsf{envy}_v(\sigma, p)].$$

Our approach will be similar to before: we obtain a sufficiently large sample S of m profiles from $\mathcal{D}$ and select the allocation that does best on the sample, i.e. it minimizes

$$\mathsf{envy}_S(\sigma, p) = \tfrac{1}{m} \textstyle\sum_{v \in S} \mathsf{envy}_v(\sigma, p).$$

5.1 Sample Complexity

For stating our sample complexity bound, we assume that valuations v are normalized: let $v_{ir} \geqslant 0$ for all $i \in N$ and $r \in R$, and $\sum_{r \in R} v_{ir} = 1$ for all $i \in N$.

Theorem 5.1. *Let $\varepsilon, \delta > 0$, and let $\mathcal{D}$ be a distribution. If we draw $m = O(\frac{n}{\varepsilon^2} \log \frac{n}{\varepsilon\delta})$ samples i.i.d. from $\mathcal{D}$ and if (σ^*, p^*) minimizes envy_S , then with probability at least $1 - \delta$, we have*

$$\mathsf{envy}_{\mathcal{D}}(\sigma^*, p^*) < \min_{(\sigma,p)} \mathsf{envy}_{\mathcal{D}}(\sigma, p) + \varepsilon.$$

Thus, if we draw sufficiently many samples, then with high probability the allocation minimizing expected envy on the sample will, up to ε, be minimizing with respect to $\mathcal{D}$.

We prove this result by discretizing the space of allocations. We then use a concentration inequality to show that w.h.p. the expected envy with respect to $\mathcal{D}$ is close to the expected envy with respect to the sample S. Due to space constraints, the proof of Theorem 5.1 appears in the appendix.

Note that for this result we employed a direct approach. This technique and its discretization step would not have worked for the envy-free rate, because two very close rent divisions can in principle have very different EF rates. On the other hand, we expect that similar bounds for the minimum envy objective could be obtained by using extensions of VC dimension to real-valued functions (e.g., pseudo-dimension).

5.2 Computational Complexity

Again, our sample complexity result needs an algorithm that finds the best allocation for a given sample S. Like for EFrate, we can solve this problem using integer linear programming (see the appendix). For the formal complexity analysis, consider the following decision problem:

> EXPECTED ENVY MINIMIZATION
> **Input:** List $S = (v^{(1)}, \ldots, v^{(m)})$, number B.
> **Question**: Is there (σ, p) with $\mathsf{envy}_S(\sigma, p) \leqslant B$?

This problem is again NP-complete. The proof is in the appendix, using a similar reduction from CLIQUE as before.

Theorem 5.2. EXPECTED ENVY MINIMIZATION *is NP-complete, even for binary valuation profiles.*

Interestingly, this problem becomes easy once we fix a room assignment σ, because the best price vector can then be computed by linear programming. In particular, this means that the problem can be solved in time $n! \cdot \text{poly}(n, m)$, and thus is fixed-parameter tractable with respect to the number of agents n. This is good news: instances will often have a small number of agents, but we will want to consider as large a sample as feasible to ensure low maximum envy. In fact, as we will see momentarily, the NP-completeness of the problem is not an obstacle in real-world instances.

6 Experiments

We evaluated our rules on user data taken from Spliddit. We studied distributions obtained by adding noise to valuations. We started by selecting 1,000 instances v at random, to speed up computations. The same selection is used for each experiment. For each instance, we normalize the rent to 1, and normalize valuations to sum to 1. We considered three noise models, each parameterized by a choice of noise level $\varepsilon \in \{0, 0.01, \ldots, 0.09\}$.

$$v_{ir}^{(\ell)} \sim v_{ir} \cdot (1 + \text{Uniform}[-\varepsilon, +\varepsilon]) \qquad \text{(Uniform)}$$

$$v_{ir}^{(\ell)} \sim v_{ir} \cdot (1 + \text{N}[0, \varepsilon]) \qquad \text{(Normal)}$$

$$v_{ir}^{(\ell)} \sim v_{ir} \cdot (1 + r \cdot \text{N}[0, \varepsilon]) \qquad \text{(Biased Normal)}$$

In each of these noise models, valuations are increased or decreased by a random fraction. Here, $N[\mu, \sigma]$ is a normal distribution with mean μ and standard deviation σ. For the biased normal noise model, we put rooms in an arbitrary fixed order and label them with integers $0, 1, \ldots, n-1$. Rooms with a higher index have more noise.

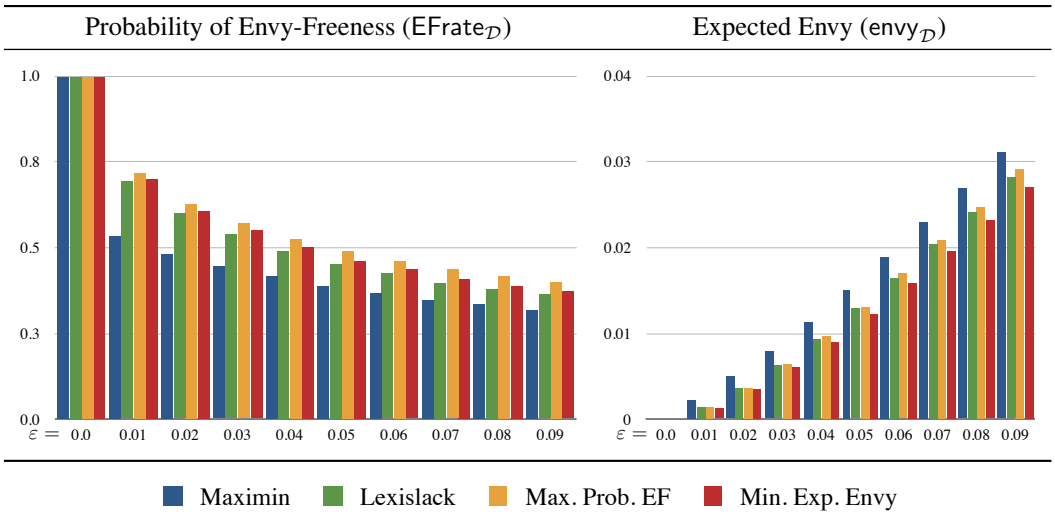

Figure 1: Results of experiments for the Normal noise model.

For each noise model and choice of ε, we produced a sample S of size $m = 100$. We then computed allocations maximizing EFrate_S and minimizing envy_S. We also computed the maximin and lexislack rules based on the input profile v. For each of the four allocations, we calculated their value of $\mathsf{EFrate}_{\mathcal{D}}$, and of $\mathsf{envy}_{\mathcal{D}}$ (by drawing a fresh sample of size 1,000). We then average over all 1,000 instances. The results are shown in Figure 1 for the Normal noise model. Results for the other noise models and more details are given in the appendix. As expected, on each of the two metrics, the rule optimizing it does best, but all three rules aiming for robustness do similarly well. Spliddit's maximin rule does significantly worse on our metrics. Before the experiments, we expected that lexislack would do worse for the biased noise model, but this does not appear to be the case. In the appendix, we also evaluate the sampling-based rules on a different probability distribution $\mathcal{D}$, to evaluate how sensitive these methods are to knowing the 'right' noise distribution.

Figure 2 shows average computation time to compute allocations optimizing EFrate_S and envy_S, using Gurobi 9.1.2 on four threads of an AMD Ryzen 2990WX (128 GB RAM) with the ILP formulations from the appendix. The results were obtained for a random selection of 300 Spliddit instances with $n = 4$, with the Uniform noise model for $\varepsilon = 0.05$, and sample sizes m varying from 1 to 125. Minimizing envy is much faster due to fewer integral variables. In the appendix, we report some additional computation times, for example as a function of the noise parameter ε.

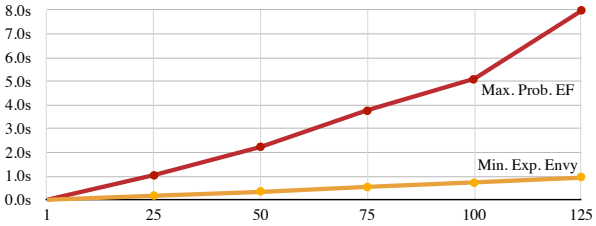

Figure 2: Computation time depending on sample size

7 Future Directions

Our approach should be applicable to many settings beyond rent division, such as homogeneous divisible goods, cake cutting, or even indivisible goods. For example, the lexislack rule can be adapted to these settings, and similar results as in our distribution-based approach might be achievable.

We have shown that the lexislack rule shares some key properties with the maximin rule, such as essential single-valuedness and polynomial-time computability. It would be interesting to axiomatically contrast the two solutions, to see if one might be better behaved than the other.

For our distribution-based approach, we assumed that we have access to $\mathcal{D}$ only via sampling. Often we may know $\mathcal{D}$ more explicitly, for example if we are just adding noise to reported valuations. For such well-behaved $\mathcal{D}$, can we design direct algorithms for finding optimal allocations with respect to our two objectives, without needing samples?

References

A. Alkan, G. Demange, and D. Gale. Fair allocation of indivisible goods and criteria of justice. *Econometrica*, 59(4):1023–1039, 1991.

E. Aragones. A derivation of the money Rawlsian solution. *Social Choice and Welfare*, 12:267–276, 1995.

M.-F. Balcan. Data-driven algorithm design. In T. Roughgarden, editor, *Beyond the Worst-Case Analysis of Algorithms*, chapter 29. Cambridge University Press, 2020.

X. Bei, Z. Li, J. Liu, S. Liu, and X. Lu. Fair division of mixed divisible and indivisible goods. *Artificial Intelligence*, 293:Article 103436, 2021.

R. Bredereck, P. Faliszewski, A. Kaczmarczyk, R. Niedermeier, P. Skowron, and N. Talmon. Robustness among multiwinner voting rules. *Artificial Intelligence*, 290:Article 103403, 2021.

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

I. Caragiannis, D. Kurokawa, H. Moulin, A. D. Procaccia, N. Shah, and J. Wang. The unreasonable fairness of maximum Nash welfare. *ACM Transactions on Economics and Computation (TEAC)*, 7 (3):1–32, 2019.

J. Chen, P. Skowron, and M. Sorge. Matchings under preferences: Strength of stability and trade-offs. In *Proceedings of the 2019 ACM Conference on Economics and Computation (EC)*, pages 41–59, 2019.

A. Critch. Robust rental harmony. *https://acritch.com/rent/*, 2015. Archived at https://perma.cc/Q9MD-HMAY.

B. Flanigan, P. Gölz, A. Gupta, B. Hennig, and A. D. Procaccia. Fair algorithms for selecting citizens' assemblies. *Nature*, 596:548–552, 2021.

Y. Gal, M. Mash, A. D. Procaccia, and Y. Zick. Which is the fairest (rent division) of them all? *Journal of the ACM*, 64(6):Article 39, 2017.

G. Gawron and P. Faliszewski. Robustness of approval-based multiwinner voting rules. In *Proceedings of the 6th International Conference on Algorithmic Decision Theory (ADT)*, pages 17–31, 2019.

J. Goldman and A. D. Procaccia. Spliddit: Unleashing fair division algorithms. *SIGecom Exchanges*, 13(2):41–46, 2014.

T. Mai and V. V. Vazirani. Finding stable matchings that are robust to errors in the input. In *Proceedings of the 26th Annual European Symposium on Algorithms (ESA)*, page Article No. 60, 2018.

V. Menon and K. Larson. Algorithmic stability in fair allocation of indivisible goods among two agents. *arXiv:2007.15203*, 2020.

N. Misra and C. Sonar. Robustness radius for Chamberlin–Courant on restricted domains. In *Proceedings of the 45th International Conference on Current Trends in Theory and Practice of Informatics (SOFSEM)*, pages 341–353, 2019.

H. Moulin. Fair division in the internet age. *Annual Review of Economics*, 11:407–441, 2019.

A. D. Procaccia, R. A. Velez, and D. Yu. Fair rent division on a budget. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 1177–1184, 2018.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

D. Shiryaev, L. Yu, and E. Elkind. On elections with robust winners. In *Proceedings of the 12th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 415–422, 2013.

F. E. Su. Rental harmony: Sperner's lemma in fair division. *American Mathematical Monthly*, 106 (10):930–942, 1999.

L.-G. Svensson. Large indivisibles: An analysis with respect to price equilibrium and fairness. *Econometrica*, 51(4):939–954, 1983.

R. A. Velez. Equitable rent division. *ACM Transactions on Economics and Computation (TEAC)*, 6 (2):Article 9, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We mention open problem in the section on future directions, as well as throughout the paper.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] Due to space constraints, many proofs appear in the appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] In the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We report standard deviations, with respect to the 500 instances.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Figure 2 and the hardware details at the end of Section 6.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We use a proprietary dataset from Spliddit.org, whose creators we cite [Goldman and Procaccia, 2014].
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] The data we received was completely anonymous.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]