Harvard CS 1360, Spring 2026
Instructor: Ariel Procaccia
Scribe: Sadie Zhao, Jakob de Raaij

Rent division

Lecture 11

In this lecture, we introduce the problem of rent division. A group of flatmates moving into a new apartment need to decide who gets which room. However, since the rooms aren't necessarily all the same, the rent for more sought-after rooms should be higher. The goal is to assign rooms and split the rent based on the reported valuations for the rooms, *fairly*.

We will consider two different approaches to this problem: First, with minimal assumptions on the utilities, and then, with more structured utilities. This illustrates a common tradeoff: A more general model will be closer to reality, while at the same time being harder to reason about. We compare the two models at the end of the lecture.

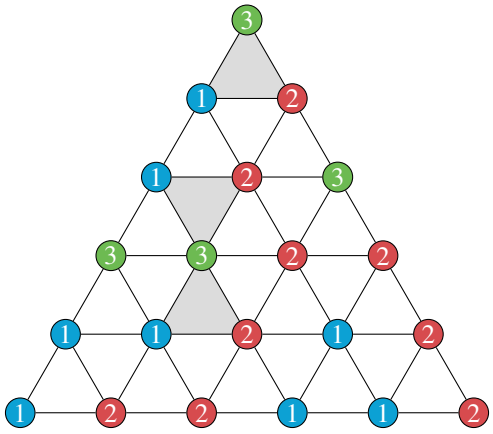
1 General Rent Division

Before we formally define the rent-division model, we introduce a combinatorial lemma that will be essential for the proofs.

Definition 1 (Sperner Labeling)**.** Given is a triangulation of a triangle, i.e., a partition of the big triangle into smaller triangles. A *Sperner labeling* assigns a number from $\{1, 2, 3\}$ to each of the vertices according to the following rules:

- The three vertices of the large triangle are labeled distinctly: one with 1, one with 2, and one with 3.
- Any vertex on an edge of the large triangle can only be labeled with one of the two labels of the endpoints of that edge.
- The interior vertices can be labeled arbitrarily with any of the three labels.

An example of a triangulation with a Sperner labeling is below.



Lemma 1 (Sperner's Lemma in two dimensions)**.** *In any Sperner labeling of a triangulation of a triangle, there exists at least one elementary (i.e., not subdivided) triangle whose vertices are labeled with three distinct numbers (one with 1, 2, and 3, each).*

For example, in the labeling above, the three elementary triangles shaded gray all have the property that their vertices have three distinct labels. We'll call those triangles *fully labeled*.

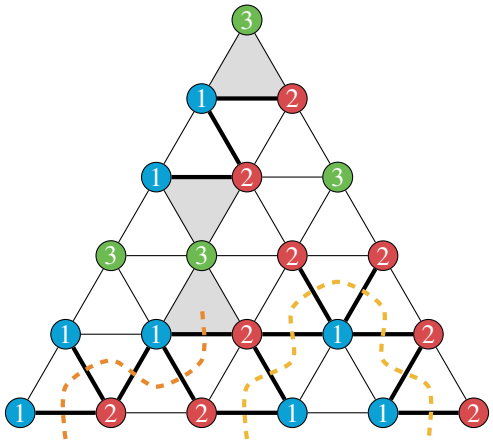
Proof. Let's now think of each elementary triangle as a *room* and of each edge between a vertex labeled 1 and a vertex labeled 2 as a *door*. This is illustrated in the triangulation below, where all doors are printed in bold.

There is an odd number of doors on the boundary of the triangle. On the sides of the big triangle, a door can only appear on the side between the corner labeled 1 and the corner labeled 2 (the bottom side in the figure below). If we move along this side of the triangle, we start at a vertex labeled with 1 and end at a vertex labeled with 2. Thus, the number of times the label switches from 1 to 2 or 2 to 1 — the number of doors that we encounter as we move across the side — needs to be odd.

Moreover, each room has at most two doors, and a room has 1 door if and only if the three vertices of the corresponding triangle have each of the labels 1, 2, and 3 once, i.e., if it is fully labeled. These are exactly the triangles for which we want to prove that at least one exists.

Let's consider going on a walk through the rooms. We start at any door on the boundary and walk into the adjacent room. This room may either have another door, in which case we proceed through it to a new room, or it has no second door, in which case we stop and know that we found a fully labeled triangle. We repeat this process to walk through the rooms until we either leave the triangle again (as is the case in the right, yellow walk shown below) or stop in a fully labeled room (as is the case in the left, orange walk shown below).

Importantly, note that we will never visit a room twice on the same walk, so every walk will come to an end, and note that two distinct walks will never meet (unless they were started at the 'entrance' and 'exit' door of the same walk). Each walk that ends with leaving the triangle connects two doors. However, since the total number of doors on the boundary of the triangle is odd, we know that at least one walk does not leave the triangle again, so it has to stop, which can only happen if it reaches a fully labeled triangle. We can conclude that, therefore, at least one fully labeled triangle has to exist.



□

Let's now formalize the rent division problem.

Definition 2 (Rent Division)**.** An instance of the *rent division* problem consists of n players (here, we will consider $n = 3$ and denote the players as A, B, and C) and n rooms (which we'll denote as rooms 1, 2, and 3). We want to match one room to each player and set a price of p_i for each room. We assume that the total rent of the apartment is 1, so it has to hold that $p_1 + ... + p_n = 1$.

The goal is to assign the rooms and divide the rent in an envy-free way.

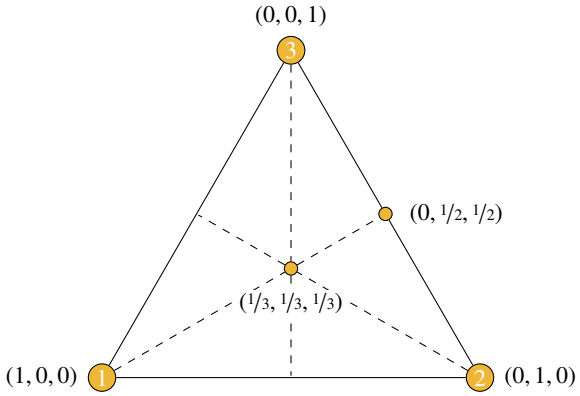
Definition 3. An allocation of rooms to players and set of prices $(p_1, ..., p_n)$ is *envy-free* if no player prefers another player's room over their assigned room, at the given prices.

We only make a single assumption on the player's preferences:

Definition 4 (Miserly Tenant Assumption)**.** Under the *miserly tenant assumption*, every player strictly prefers any room with price 0 over any room with a positive price.

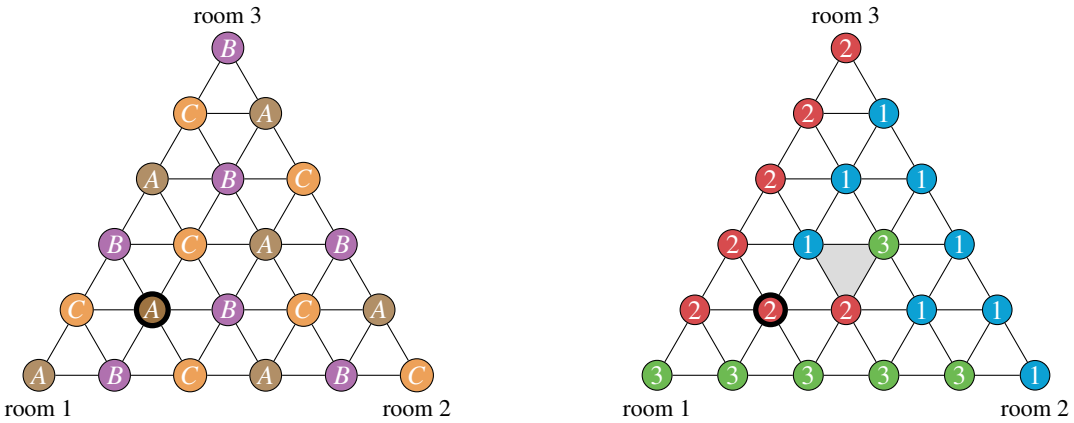
Theorem 1. *Under the miserly tenant assumption, an envy-free allocation and prices always exist.*

Proof. We represent the possible divisions of the rent for the three rooms as a ternary plot.



Each point in the triangle corresponds to some prices (p_1, p_2, p_3) with $p_1 + p_2 + p_3 = 1$, where p_i is larger the closer the point is to the vertex corresponding to room i.[1] The bottom left vertex represents $(p_1, p_2, p_3) = (1, 0, 0)$, the bottom right vertex represents $(p_1, p_2, p_3) = (0, 1, 0)$, and the top vertex represents $(p_1, p_2, p_3) = (0, 0, 1)$.

We can triangulate this large triangle. For each elementary triangle, we assign A, B, and C to a vertex so that every elementary triangle is a fully labeled triangle, giving the left labeling.



For every vertex with label A, we ask player A which room in $\{1, 2, 3\}$ they would prefer at the prices corresponding to that vertex. We do that same for vertices with label B and C. This will give us a new labeling by $\{1, 2, 3\}$ of the triangulation, shown on the right. For example, the marked A vertex in the labeling on the left corresponds to asking player A "Which room do you prefer at prices $(3/5, 1/5, 1/5)$?" Suppose they answer "Room 2", in which case we label this vertex with a 2 in the right labeling.

All vertices on the side opposite of room i correspond to the price for room i being 0. Thus, by the miserly tenant assumption, we know that all these vertices are labeled with i since any player will always prefer a free room over a room with a positive price. The only exception are the corners, where two rooms are free, that are labeled with either of the two possible room numbers.

These boundary conditions don't perfectly match the criteria of a Sperner labeling. However, we can apply the exact same proof as for Sperner's Lemma to see that a fully labeled triangle has to always exist: There will exist exactly one 'door' on the boundary of the big triangle, so the 'walk' from this door has to end in some 'room,' which corresponds to a fully labeled triangle.

If we assign the rooms to the players according to their preferences at the three corners of the fully labeled triangle, and set the prices $\boldsymbol{p}$ to be a point in the interior of the triangle (e.g., the centroid), this outcome will be *almost* envy-free. In particular, we know that each player does indeed prefer their assigned room at prices very similar to $\boldsymbol{p}$, in particular at the vertex corresponding to their choice of favorite room in the triangle. As we refine the triangulation by increasing the number of elementary triangles, this approximation to envy-freeness becomes increasingly precise.

[1] In particular, imagine drawing a line from the vertex labeled i through the point (p_1, p_2, p_3) up to the opposite side of the triangle. Then the fraction of the length of this line that is between the point and the opposite side of the triangle is p_i.

In the limit, assuming the preference space is compact[2], we can extend this argument to establish the existence of a (perfectly, not just approximately) envy-free rent division. □

The same proof technique extends to settings with more than three players almost analogously by applying the higher-dimensional versions of Sperner's Lemma (which can be proved very similarly to the proof we saw above). Furthermore, a similar argument also using Sperner's Lemma, establishes the existence of an envy-free cake cutting allocation (as seen last lecture) where each player gets a connected piece. In this case, each vertex in the triangulation corresponds to the size of the cake piece, with piece i being $[\sum_{j=1}^{i-1} p_j, \sum_{j=1}^{i} p_j]$ of size p_i (and an 'expensive' piece of cake being desirable).

The proof above also naturally leads to an algorithm for computing an approximately envy-free rent division by querying players' preferences at increasingly small elementary triangles to find a fully labeled elementary triangle (the smaller that triangle, the better the approximation). You can see this algorithm implemented here. This algorithm is very general, since it requires no assumptions about players' valuations beyond the miserly tenant assumption (and preferences being compact). However, a downside of this approach is that we do not know which envy-free allocation we end up in if there exists more than one. In particular, we cannot optimize over the set of all envy-free allocations for the one maximizing some second-order objective, such as the prices being as equal as possible.

2 Quasi-Linear Rent Division

Definition 5 (Rent Division with Quasi-Linear Utilities)**.** An instance of the *rent division* problem with *quasi-linear utilities* consists of

- a set N of n players,
- a set M of n *rooms*,
- a *rent* R, where we will assume $R = 1$, and
- for each player $i \in N$ and room $r \in M$ the valuation v_{ir} of player i for room r. For each player, their valuations sum up to the total rent, i.e., $\sum_{r \in M} v_{ir} = R$.

A solution $(\pi, \boldsymbol{p})$ to a rent-division problem consists of an assignment $\pi : N \to M$ matching players to rooms and a price vector $\boldsymbol{p} = p_1, ..., p_n$, where p_r is the price of room r and $\sum_{r \in M} p_r = R$. The utility of player i for this solution is their valuation for their allocated room $r = \pi(i)$ minus its price p_r, i.e., $v_{ir} - p_r$.

Let's restate the definition of envy-freeness, applied to this setting.

Definition 6. A solution $(\pi, \boldsymbol{p})$ to a rent division problem is *envy-free* if for all players $i, j \in N$, $v_{i\pi(i)} - p_{\pi(i)} \geq v_{i\pi(j)} - p_{\pi(j)}$

Note that the miserly tenant assumption is not consistent with this model. If a player i values a room r higher than a room r', at any price $p_r \in (0, v_{ir} - v_{ir'})$, this player i will prefer room r at price p_r over room r' for free. Nonetheless, it holds that:

Theorem 2. *In rent division with quasi-linear utilities, an envy-free solution always exists.*

We'll next examine two useful properties of envy-free solutions in this setting, relying on the following definition:

Definition 7. An assignment π is *welfare-maximizing* if it maximizes the valuations of the players for their rooms. That is,

$$\pi \in \arg\max_{\sigma} \sum_{i \in N} v_{i\sigma(i)}.$$

Lemma 2. *If $(\pi, \boldsymbol{p})$ is an envy-free solution, then π is a welfare-maximizing assignment.*

[2] A space X is compact if every infinite sequence in X has a convergent subsequence whose limit lies in X.

Proof. Let $(\pi, \boldsymbol{p})$ be an envy-free solution and let σ be another assignment. Envy-freeness tells us that for all $i \in N$, $v_{i\pi(i)} - p_{\pi(i)} \geq v_{i\sigma(i)} - p_{\sigma(i)}$. Summing this over all $i \in N$, we obtain that

$$\sum_{i \in N} v_{i\pi(i)} - \sum_{i \in N} p_{\pi(i)} \geq \sum_{i \in N} v_{i\sigma(i)} - \sum_{i \in N} p_{\sigma(i)}$$

Since $\sum_{i \in N} p_{\pi(i)} = \sum_{i \in N} p_{\sigma(i)} = R$, i.e., the total prices always equal the rent R, it holds that

$$\sum_{i \in N} v_{i\pi(i)} \geq \sum_{i \in N} v_{i\sigma(i)},$$

so π is welfare-maximizing. □

Lemma 3. *If $(\pi, \mathbf{p})$ is an envy-free solution and σ is a welfare-maximizing assignment, then $(\sigma, \mathbf{p})$ is an envy-free solution.*

Based on the insights from Theorem 2 and Lemmas 2 and 3, we can define a polynomial-time algorithm for finding envy-free assignments.

Algorithm 1 Envy-Free Solution to Rent Division with Quasi-Linear Utilities

1: Find a welfare-maximizing assignment π.
2: Find prices $\boldsymbol{p}$ that satisfy the envy-freeness constraints for allocation π.
3: Return $(\pi, \boldsymbol{p})$.

Theorem 3. *This algorithm always returns an envy-free solution and can be implemented in polynomial time.*

Proof. First, we need to confirm that for any welfare-maximizing assignment π, there exists some $\boldsymbol{p}$ so that $(\pi, \boldsymbol{p})$ is envy-free. By Theorem 2, we know that some envy-free solution $(\sigma, \mathbf{p})$ exists, so by Lemma 3, we know that $(\pi, \mathbf{p})$ is also envy-free.

Let's now consider how to implement this algorithm in polynomial time: The first part — finding a welfare maximizing assignment — is equivalent to finding a maximum weight perfect matching in a bipartite graph. In particular, the bipartite graph has players on one side and the rooms on the other side with edges between any player and room pair. The weight of the edge (i, r) for player $i \in N$ and room r is v_{ir}. Any maximum weight perfect matching of this graph corresponds to a welfare maximizing assignment; such a maximum weight perfect matching can be found in polynomial time, for example with maximum-flow algorithms.

The second part — finding the $\boldsymbol{p}$ so that $(\pi, \boldsymbol{p})$ is envy-free — is equivalent to finding prices $p_1, ..., p_n$ that satisfy the $\binom{n}{2}$ inequalities from the envy-freeness definition simultaneously. This can be solved in polynomial time with linear programming. □

It may be the case that more than one envy-free solution exists.

Example 1. Consider the following utilities for the three rooms with rent $R = 3$.

	Player A	Player B	Player C
Room 1	3	0	0
Room 2	0	3	0
Room 3	0	0	3

The only welfare-maximizing solution is to assign room 1 to player A, room 2 to player B, and room 3 to player C. Any price vector will lead to an envy-free solution. In particular, even if player A pays all the rent for room 1, $p_1 = 3$, they are still going to be envy-free since they get a utility of 0, equivalent to their utility of 0 for getting room 2 or 3 and paying nothing.

This example shows that envy-freeness is not enough to argue that a solution is fair. Thus, it makes sense to consider additional criteria for which envy-free solution to pick:

- A *straw man solution* is to maximize the sum of utilities subject to envy-freeness. However, as we have seen in Lemmas 2 and 3, any envy-free solution maximizes the sum of utilities, so this objective does not narrow the set of solutions down any further.
- The *maximin solution* is to choose the envy-free solution that maximizes the minimum utility of any player.
- The *equitable solution* is to choose the envy-free solution minimizing the maximum difference between the utilities of any two players.

Theorem 4. *The maximin and equitable solutions can be computed in polynomial time.*

Theorem 5. *For any player i, the utility of this player i is the same in all maximin solutions.*

Therefore, at least in our model, it cannot happen that a player prefers one maximin solution over another (as was the case for envy-free solutions in Example 1).

Theorem 6. *The maximin solution is equitable, but not vice versa.*

3 Comparison Between Two Models

The following table summarizes the tradeoffs between the more general approach, leading to a generally applicable existence result conditioned on one assumption, to the quasi-linear utility approach, that makes stronger assumptions but also allows for better algorithms.

	General Rent Division	**Quasi-Linear Utilities**
Expressiveness	More expressive and general.	Less expressive (e.g., fails to capture budget constraints or non-linear utilities).
Computation	No computational guarantees.	Polynomial-time algorithm for computing envy-free solutions.
Querying	Preference elicitation requires many questions to the players	Each player only needs to reveal a single number, their valuation.
Extensions	There is no obvious way to choose between envy-free solutions.	We can choose better envy-free solutions efficiently.