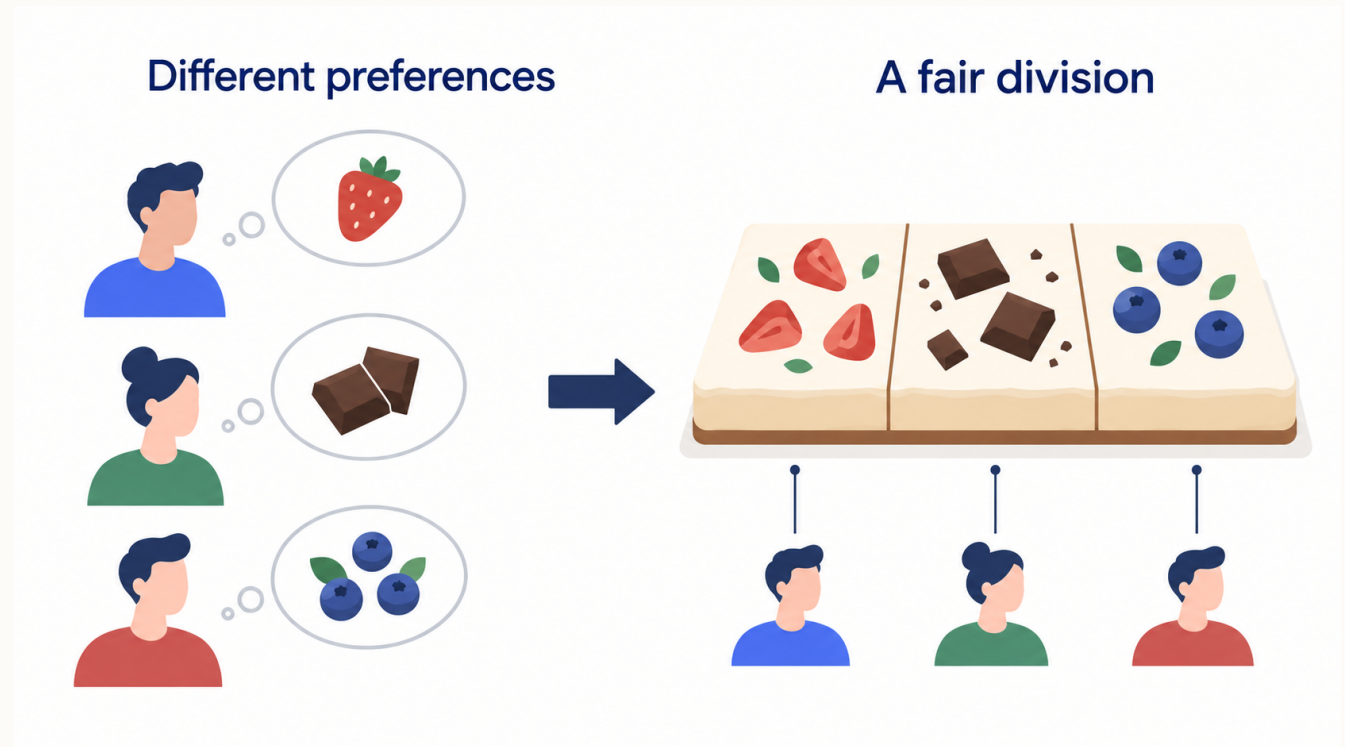


Fair Division

Lecture 01: Cake Cutting

Hannaneh Akrami



Course Logistics

Time & Location

Monday 10:15–11:45
Room 1.002 (Seminarraum)
Lennéstraße 2–4

Format

2-hour lecture per week
Oral exam

Credits

5 ECTS

Exercises

Not graded. I strongly encourage you to solve the exercises yourselves. If you want feedback on them, you can hand in your solution and I will give you feedback.

Contact

hakrami@uni-bonn.de



Course website

hanoon.github.io/Fair-Division-Course/

Outline of This Session

Introduction to **fair division**, **cake cutting**, **moving-knife procedures**, and the **Robertson–Webb model**.



Computational Social Choice Theory

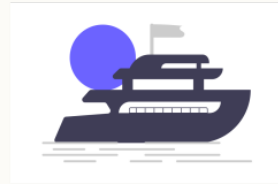
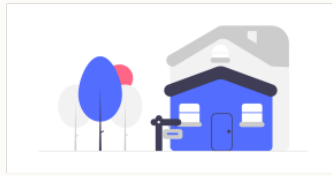
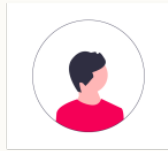
Social Choice Theory: Making a **collective** decision from **individual** preferences.

Computational Social Choice Theory

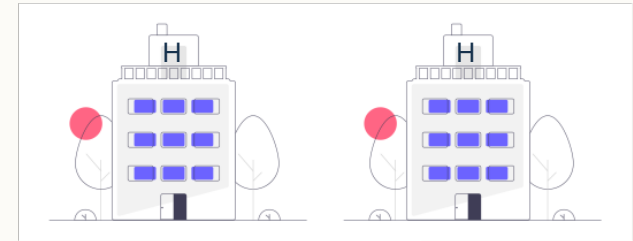
Social Choice Theory: Making a **collective** decision from **individual** preferences.



Voting



Resource Allocation



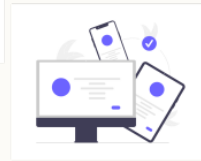
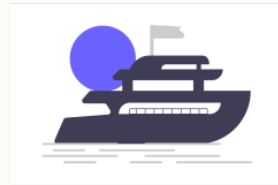
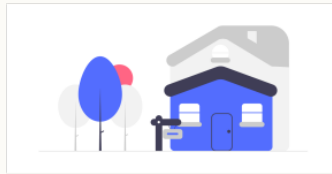
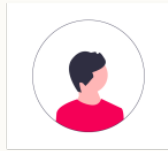
Stable Matchings

Computational Social Choice Theory

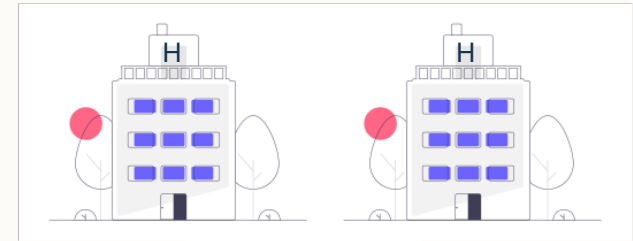
Social Choice Theory: Making a **collective** decision from **individual** preferences.



Voting



Resource Allocation



Stable Matchings

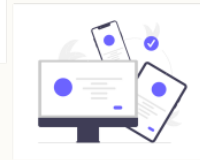
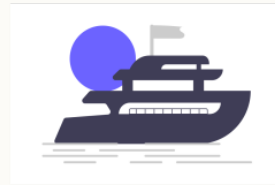
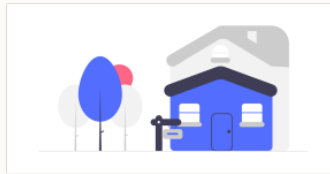
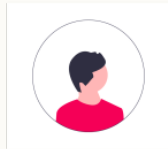
Economists and Politicians: Does there exist a **social choice** mechanism with the desired economic properties?

Computational Social Choice Theory

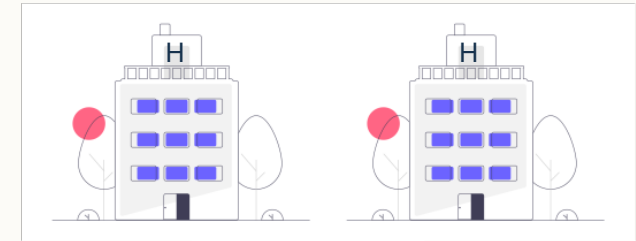
Social Choice Theory: Making a **collective** decision from **individual** preferences.



Voting



Resource Allocation



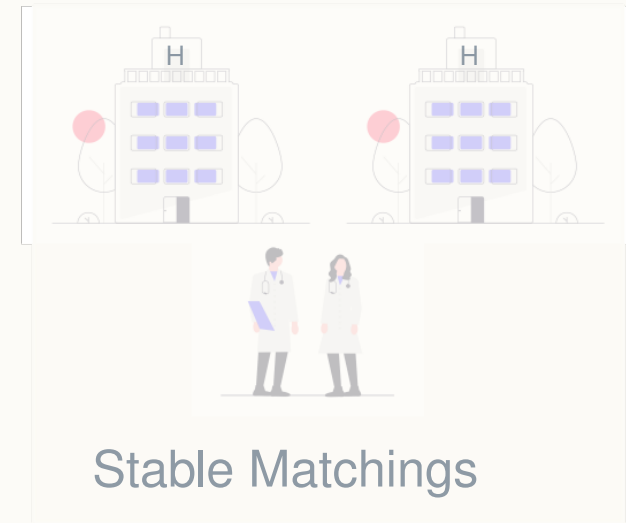
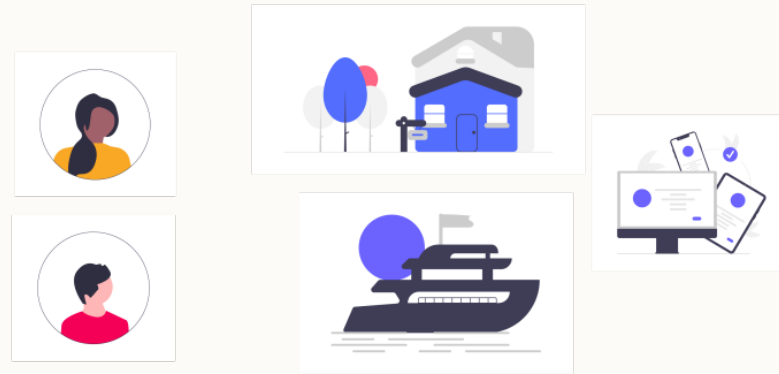
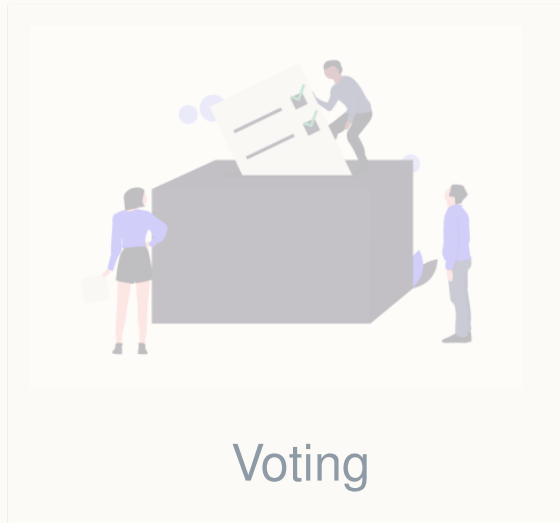
Stable Matchings

Economists and Politicians: Does there exist a **social choice** mechanism with the desired economic properties?

Mathematicians / Computer Scientists: How can we (efficiently) **compute** such a mechanism?

Computational Social Choice Theory

Social Choice Theory: Making a **collective** decision from **individual** preferences.



Economists and Politicians: Does there exist a **social choice** mechanism with the desired economic properties?

Mathematicians / Computer Scientists: How can we (efficiently) **compute** such a mechanism?

In this course, we focus on **resource allocation**.

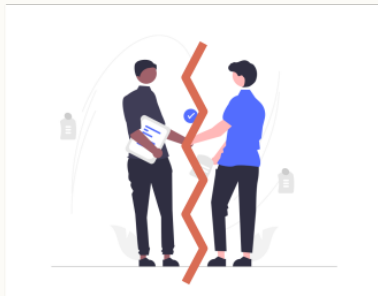
Fair Division

Fair division is the problem of dividing **items** among **agents** in a **fair** manner.

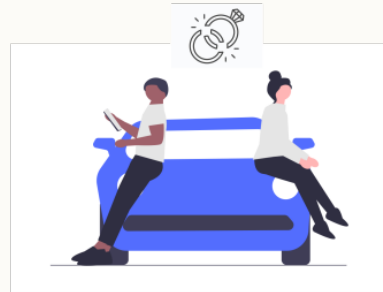
Fair Division

Fair division is the problem of dividing **items** among **agents** in a **fair** manner.

Applications:



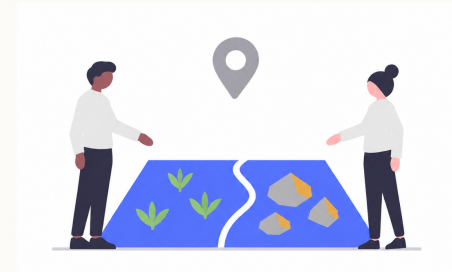
Partnership
dissolution



Divorce
settlements



Budget
allocation



Land
division

Types of Items

```
graph TD; A[Types of Items] --> B[Desirable]; A --> C[Undesirable];
```

Desirable

Undesirable

Types of Items

Desirable



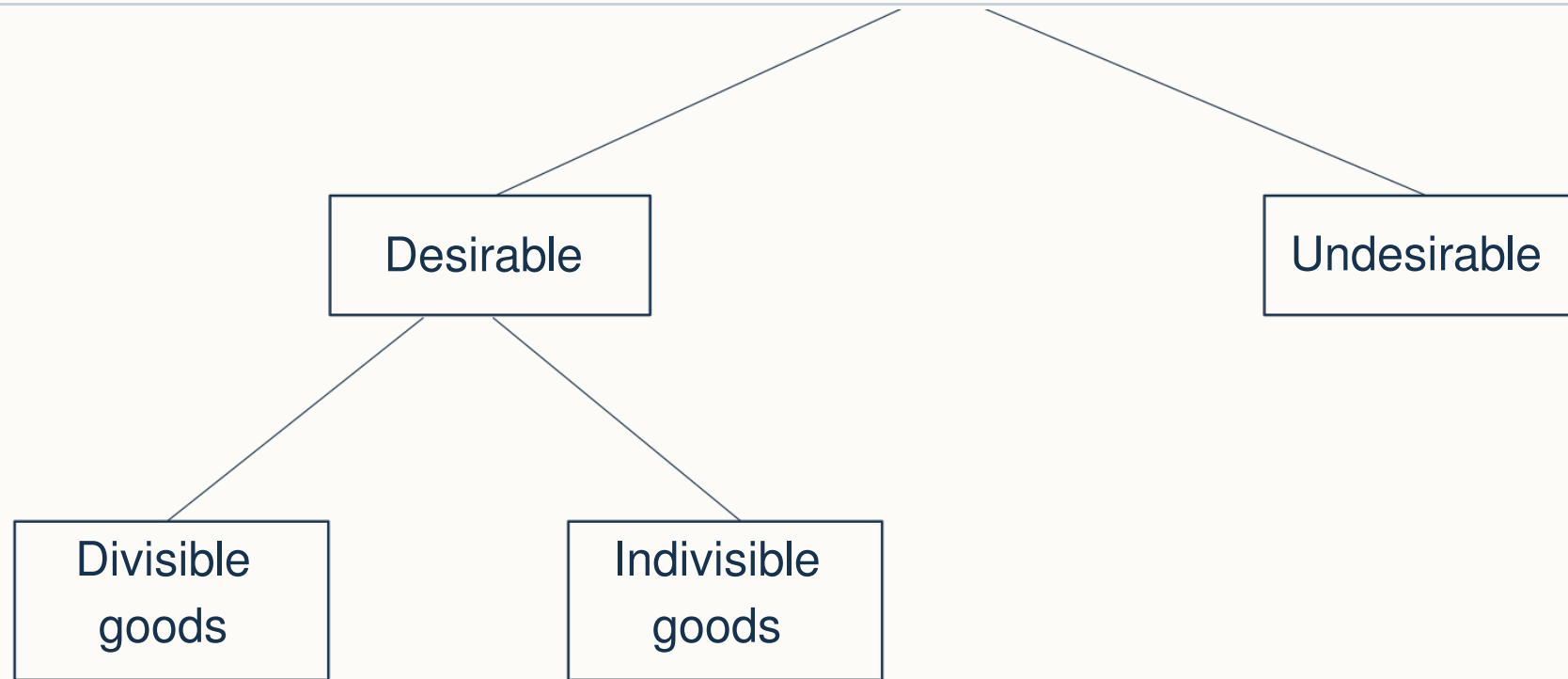
Divorce
settlements

Undesirable

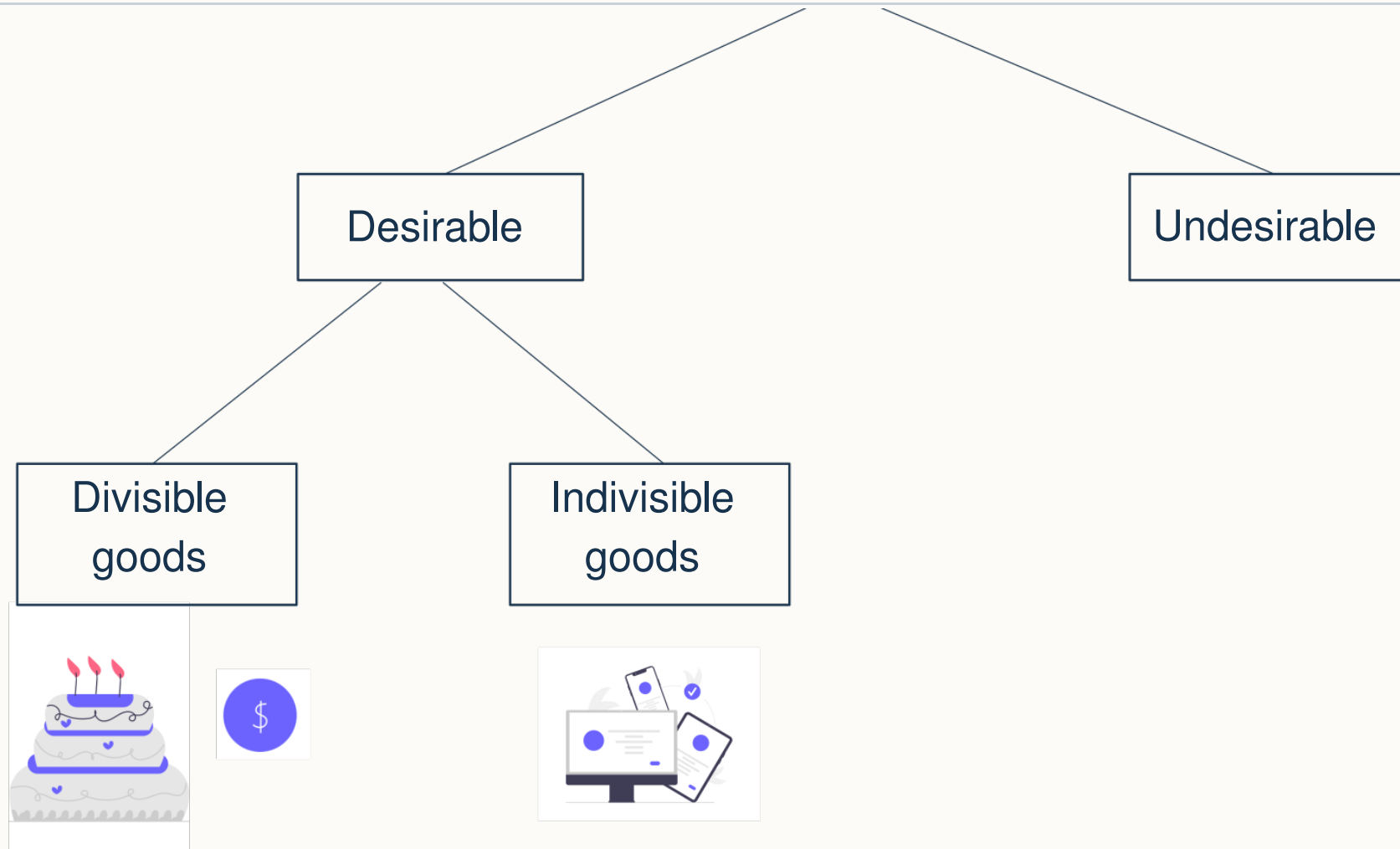


Household
chores

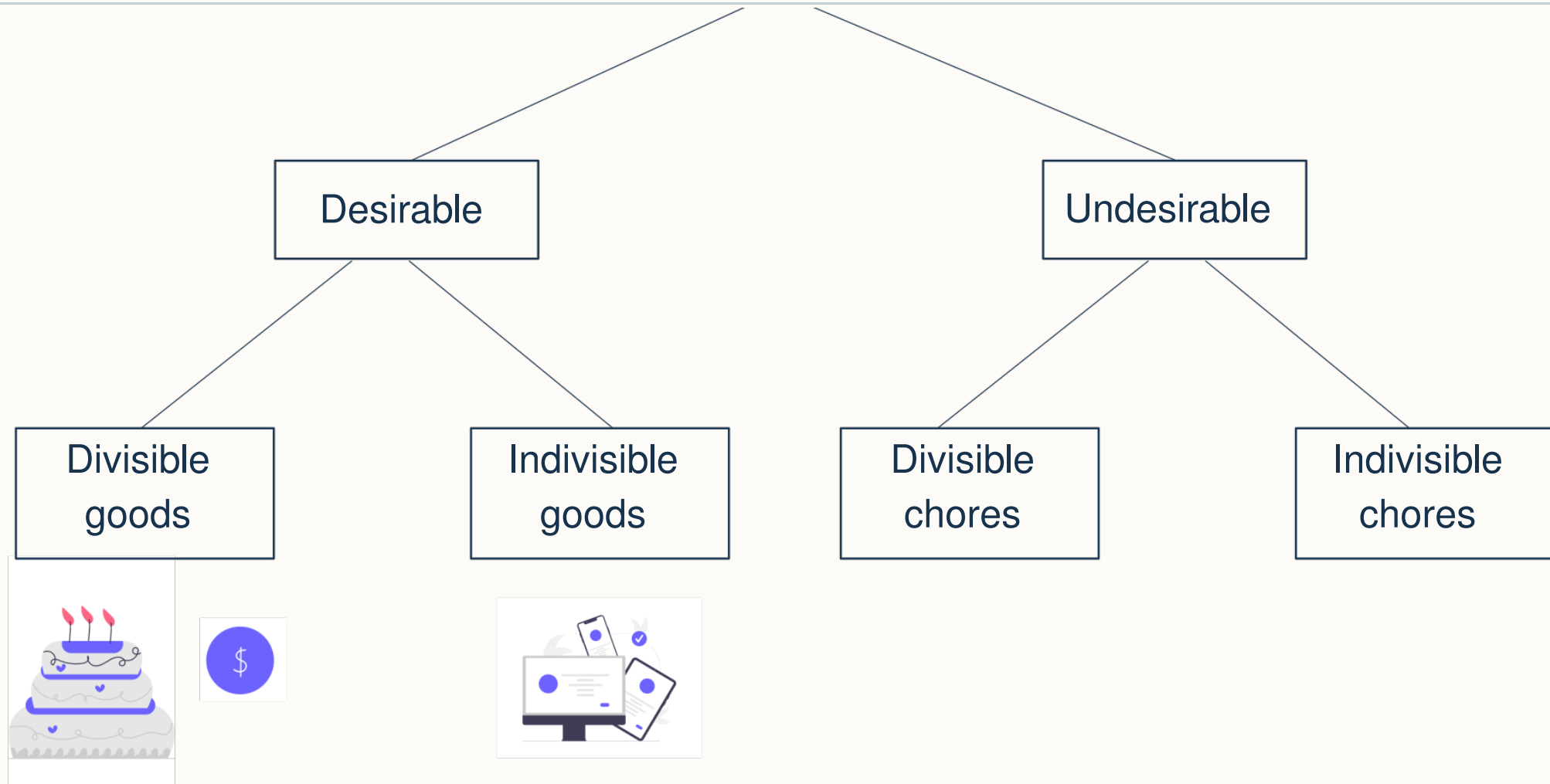
Types of Items



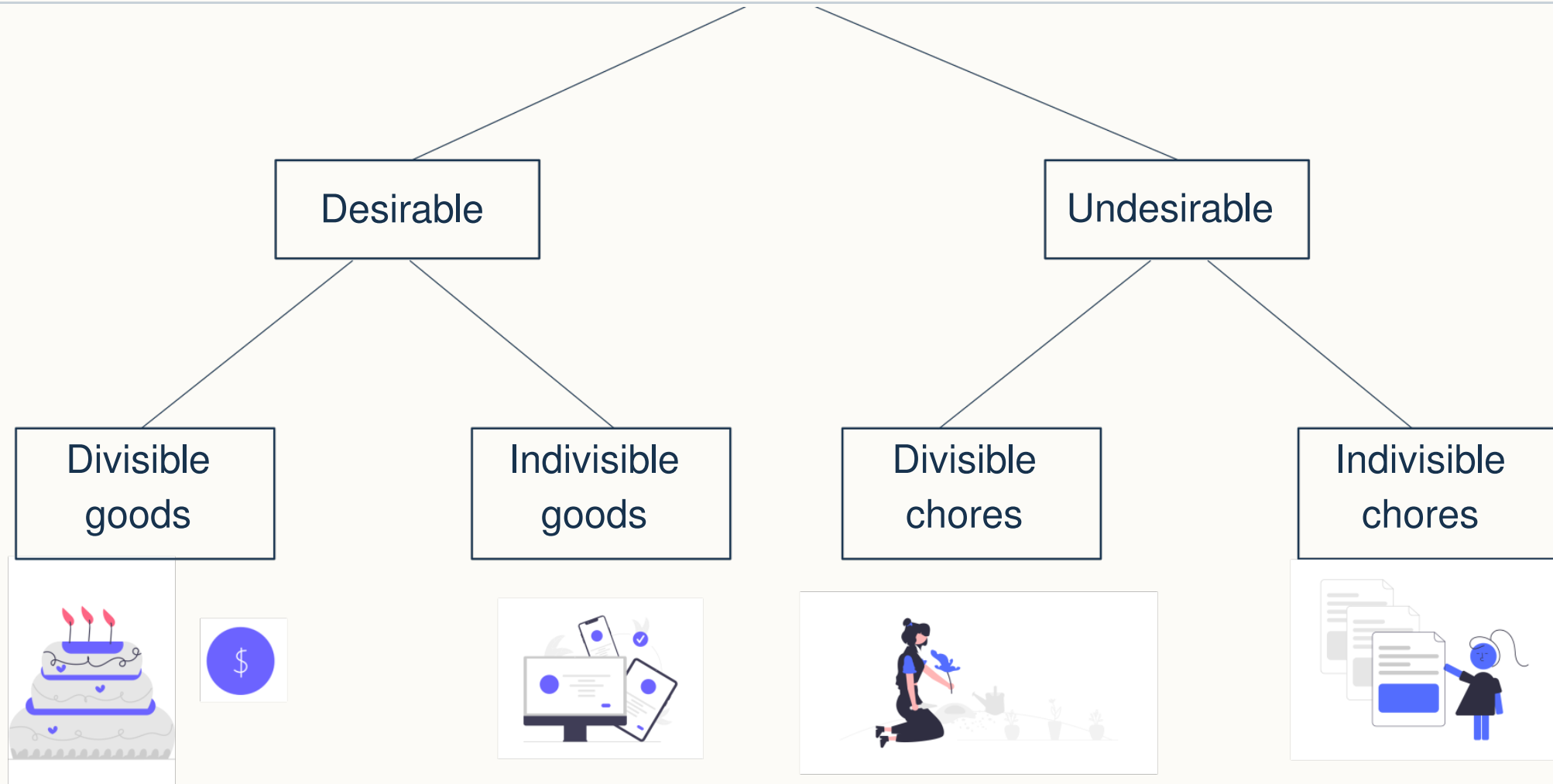
Types of Items



Types of Items



Types of Items



Two Central Questions

What is **fairness as a concept**?

How do we **compute a fair allocation**?

Goal: to **divide** the resource among the agents in a **fair** manner.

Two Central Questions

What is **fairness as a concept**?

How do we **compute a fair allocation**?

- **Mathematical study** of fairly allocating resources among agents with distinct preferences, but equal entitlements.

Goal: to **divide** the resource among the agents in a **fair** manner.

Two Central Questions

What is **fairness as a concept**?

How do we **compute a fair allocation**?

- **Mathematical study** of fairly allocating resources among agents with distinct preferences, but equal entitlements.
- Focus on **provable guarantees**.

Goal: to **divide** the resource among the agents in a **fair** manner.

Two Central Questions

What is **fairness as a concept**?

How do we **compute a fair allocation**?

- **Mathematical study** of fairly allocating resources among agents with distinct preferences, but equal entitlements.
- Focus on **provable guarantees**.
- **Computational Perspective**: work towards algorithms & hardness results and approximation algorithms

Goal: to **divide** the resource among the agents in a **fair** manner.

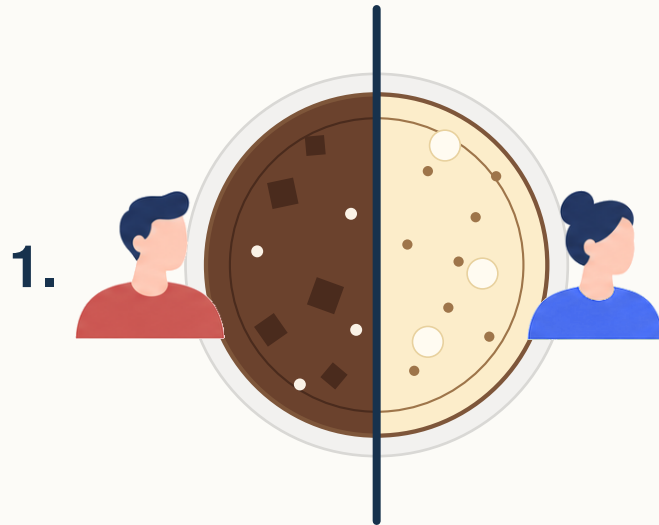
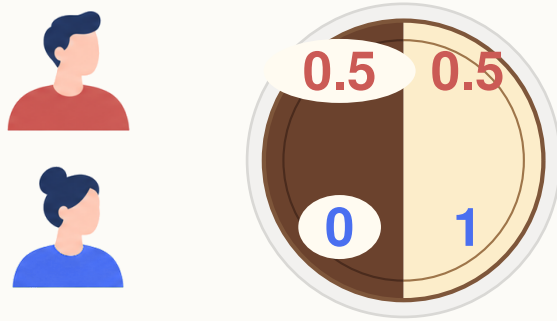
Cake Cutting

Fair division of **divisible** goods is also known as **cake cutting**.



How to **fairly** divide a cake among agents with **differing preferences**?

What is fair?



Cake Cutting: Formal Definition

In the cake cutting problem, we have one cake (one resource) that must be divided among a number of individuals.

- **Cake** C : we model the cake as the interval between zero and one, $C = [0, 1]$.



Cake Cutting: Formal Definition

In the cake cutting problem, we have one cake (one resource) that must be divided among a number of individuals.

- **Cake C** : we model the cake as the interval between zero and one, $C = [0, 1]$.
- **Set of agents**: $N = [n] = \{1, 2, \dots, n\}$, the agents among whom the cake must be divided.



Cake Cutting: Formal Definition

In the cake cutting problem, we have one cake (one resource) that must be divided among a number of individuals.

- **Cake** C : we model the cake as the interval between zero and one, $C = [0, 1]$.
- **Set of agents**: $N = [n] = \{1, 2, \dots, n\}$, the agents among whom the cake must be divided.
- **Valuation function** of player i : $v_i : 2^{[0,1]} \rightarrow \mathbb{R}_{\geq 0}$. For every subinterval I of $[0, 1]$, $v_i(I)$ specifies the value of that subinterval.



Assumptions on Valuations

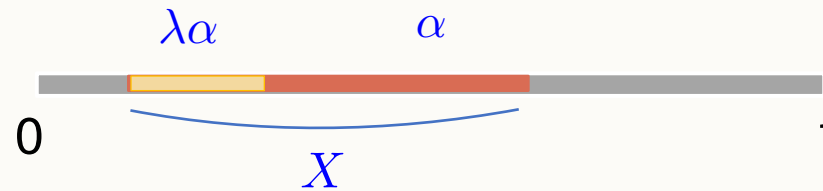
We assume that the valuation functions have the following properties:

1. **Normalization**: the valuation functions are normalized so that $\forall i : v_i(C) = 1$.

Assumptions on Valuations

We assume that the valuation functions have the following properties:

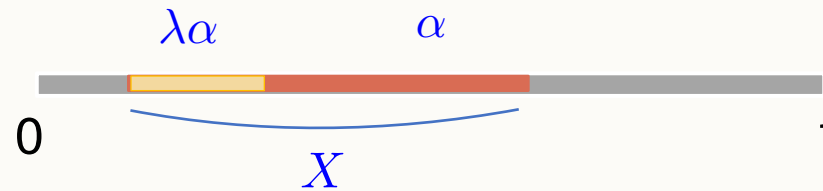
1. **Normalization**: the valuation functions are normalized so that $\forall i : v_i(C) = 1$.
2. **Divisibility**: for every interval $[x, y]$ and every $0 \leq \lambda \leq 1$, there exists a point $z \in [x, y]$ such that $v_i([x, z]) = \lambda v_i([x, y])$.



Assumptions on Valuations

We assume that the valuation functions have the following properties:

1. **Normalization**: the valuation functions are normalized so that $\forall i : v_i(C) = 1$.
2. **Divisibility**: for every interval $[x, y]$ and every $0 \leq \lambda \leq 1$, there exists a point $z \in [x, y]$ such that $v_i([x, z]) = \lambda v_i([x, y])$.

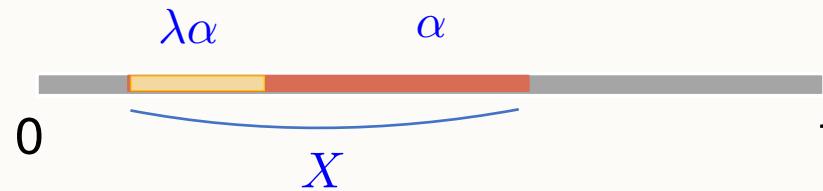


This implies that the valuation functions are *non-atomic* and that intervals of length zero have value zero. Hence, in our discussions, there is no difference between “]” and “)”

Assumptions on Valuations

We assume that the valuation functions have the following properties:

1. **Normalization**: the valuation functions are normalized so that $\forall i : v_i(C) = 1$.
2. **Divisibility**: for every interval $[x, y]$ and every $0 \leq \lambda \leq 1$, there exists a point $z \in [x, y]$ such that $v_i([x, z]) = \lambda v_i([x, y])$.



This implies that the valuation functions are *non-atomic* and that intervals of length zero have value zero. Hence, in our discussions, there is no difference between “]” and “)”.

3. **Non-negativity**: $\forall i, \forall I : v_i(I) \geq 0$.

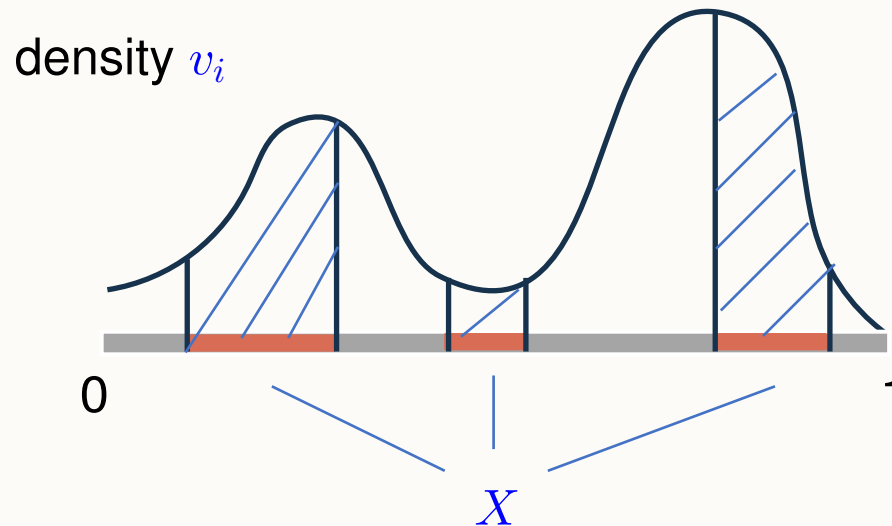
Pieces and Value Densities

Definition. A **piece** of the cake is a set of finitely many disjoint intervals of the cake. For a piece A , $|A|$ is the number of intervals in A . We also define $v_i(A) = \sum_{I \in A} v_i(I)$.

Pieces and Value Densities

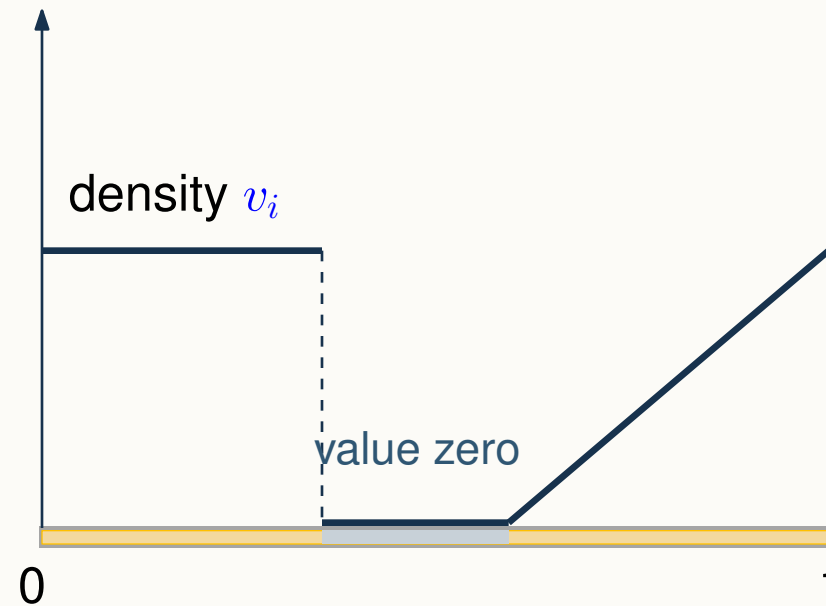
Definition. A **piece** of the cake is a set of finitely many disjoint intervals of the cake. For a piece A , $|A|$ is the number of intervals in A . We also define $v_i(A) = \sum_{I \in A} v_i(I)$.

For convenience, we may assume that each individual has a **density function** v_i defined on $[0, 1]$, so that $v_i([x, y]) = \int_x^y v_i(t) dt$. The function differs between individuals.



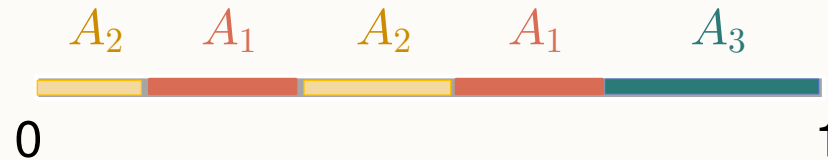
Pieces and Value Densities

However, v_i need not be continuous; some parts of the cake may have value zero, for example (piecewise integral):



Allocations

Definition. We denote an **allocation** of the cake to the individuals by $A = (A_1, A_2, \dots, A_n)$, where A_i is the piece (possibly several intervals) allocated to individual i .



Here, agent 1 and agent 2 each receive a piece that consists of two intervals.

Fairness Criteria

The goal of the problem is the fair division (allocation) of the cake among the individuals. But first, we must specify what *fair* means. Many fairness criteria have been introduced; we present three of them here:

1. **Proportional**: an allocation $A = (A_1, \dots, A_n)$ is proportional if every individual feels that they have received at least $\frac{1}{n}$ of the cake:
$$\forall i : v_i(A_i) \geq \frac{1}{n}$$

Fairness Criteria

The goal of the problem is the fair division (allocation) of the cake among the individuals. But first, we must specify what *fair* means. Many fairness criteria have been introduced; we present three of them here:

1. **Proportional**: an allocation $A = (A_1, \dots, A_n)$ is proportional if every individual feels that they have received at least $\frac{1}{n}$ of the cake:

$$\forall i : v_i(A_i) \geq \frac{1}{n}$$

2. **Envy-free**: everyone prefers their own share to the shares of others:

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

Fairness Criteria

The goal of the problem is the fair division (allocation) of the cake among the individuals. But first, we must specify what *fair* means. Many fairness criteria have been introduced; we present three of them here:

1. **Proportional**: an allocation $A = (A_1, \dots, A_n)$ is proportional if every individual feels that they have received at least $\frac{1}{n}$ of the cake:

$$\forall i : v_i(A_i) \geq \frac{1}{n}$$

2. **Envy-free**: everyone prefers their own share to the shares of others:

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

3. **Equitable**: everyone's level of satisfaction with their own share is the same:

$$\forall i, j : v_i(A_i) = v_j(A_j)$$

Additional Properties

Besides fairness, we may want the division to have other properties as well:

1. **Contiguous**: each individual receives exactly one interval: $\forall i : |A_i| = 1$.

Additional Properties

Besides fairness, we may want the division to have other properties as well:

1. **Contiguous**: each individual receives exactly one interval: $\forall i : |A_i| = 1$.
2. **Pareto optimal**: there is no allocation A' in which nobody is worse off and at least one individual is strictly better off:

$$\nexists A' : \quad \forall i : v_i(A'_i) \geq v_i(A_i) \quad \text{and} \quad \exists j : v_j(A'_j) > v_j(A_j)$$

Additional Properties

Besides fairness, we may want the division to have other properties as well:

1. **Contiguous**: each individual receives exactly one interval: $\forall i : |A_i| = 1$.
2. **Pareto optimal**: there is no allocation A' in which nobody is worse off and at least one individual is strictly better off:

$$\nexists A' : \quad \forall i : v_i(A'_i) \geq v_i(A_i) \quad \text{and} \quad \exists j : v_j(A'_j) > v_j(A_j)$$

3. **Non-wasteful**: the whole cake is allocated.

Additional Properties

Besides fairness, we may want the division to have other properties as well:

1. **Contiguous**: each individual receives exactly one interval: $\forall i : |A_i| = 1$.
2. **Pareto optimal**: there is no allocation A' in which nobody is worse off and at least one individual is strictly better off:
$$\nexists A' : \quad \forall i : v_i(A'_i) \geq v_i(A_i) \quad \text{and} \quad \exists j : v_j(A'_j) > v_j(A_j)$$
3. **Non-wasteful**: the whole cake is allocated.
4. **Truthful**: if agents were asked to report their valuations, they would have no incentive to lie.

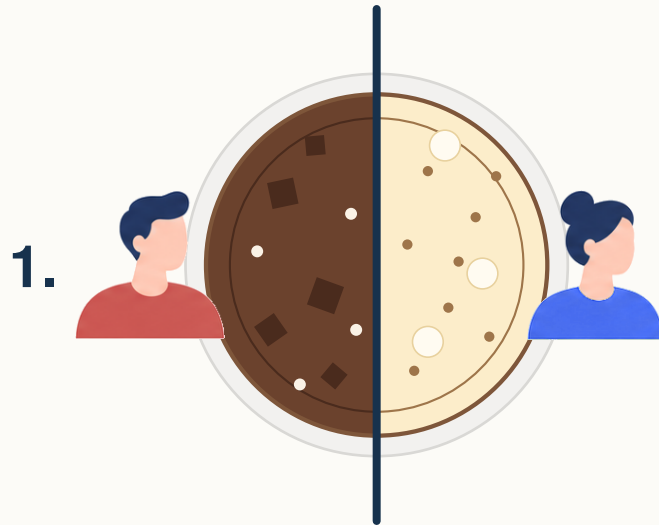
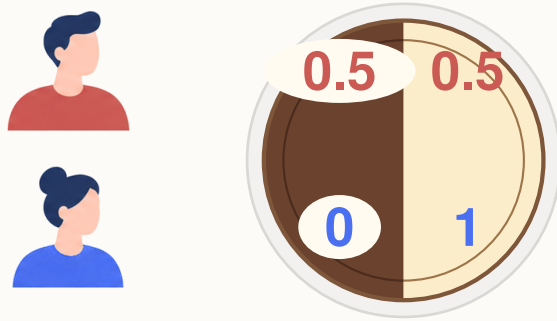
Additional Properties

Besides fairness, we may want the division to have other properties as well:

1. **Contiguous**: each individual receives exactly one interval: $\forall i : |A_i| = 1$.
2. **Pareto optimal**: there is no allocation A' in which nobody is worse off and at least one individual is strictly better off:
$$\nexists A' : \forall i : v_i(A'_i) \geq v_i(A_i) \quad \text{and} \quad \exists j : v_j(A'_j) > v_j(A_j)$$
3. **Non-wasteful**: the whole cake is allocated.
4. **Truthful**: if agents were asked to report their valuations, they would have no incentive to lie.

This is the overall definition of the fair allocation problem and of our goals in it.

What is fair?



Envy-Freeness and Proportionality

Theorem. In non-wasteful allocations, we have:

- $\text{EF} \Rightarrow \text{proportionality}$.
- For 2 agents: $\text{EF} \iff \text{proportionality}$.
- For $n > 2$ agents: $\text{proportionality} \nRightarrow \text{EF}$.

Envy-Freeness and Proportionality

Theorem. In non-wasteful allocations, we have:

- $EF \Rightarrow \text{proportionality}$.
- For 2 agents: $EF \iff \text{proportionality}$.
- For $n > 2$ agents: $\text{proportionality} \nRightarrow EF$.

EF



Prop

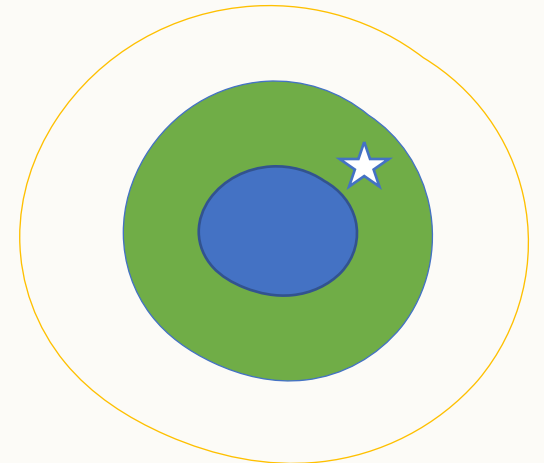
for *any* number of agents

Prop



EF

for *two* agents



All allocations

Proof: $EF \Rightarrow$ Proportionality

Proof: EF $\Rightarrow$ Proportionality

By additivity and normalization, for every agent i ,

$$\sum_{j=1}^n v_i(A_j) = v_i(C) = 1. \tag{1}$$

Proof: EF $\Rightarrow$ Proportionality

By additivity and normalization, for every agent i ,

$$\sum_{j=1}^n v_i(A_j) = v_i(C) = 1. \tag{1}$$

Let A be envy-free and fix an agent i .

By envy-freeness, $v_i(A_i) \geq v_i(A_j)$ for all j .

Summing over all $j \in N$ and using (1), we get

$$n v_i(A_i) \geq \sum_{j=1}^n v_i(A_j) = 1,$$

hence $v_i(A_i) \geq \frac{1}{n}$.

Proof: EF $\Rightarrow$ Proportionality

By additivity and normalization, for every agent i ,

$$\sum_{j=1}^n v_i(A_j) = v_i(C) = 1. \tag{1}$$

Let A be envy-free and fix an agent i .

By envy-freeness, $v_i(A_i) \geq v_i(A_j)$ for all j .

Summing over all $j \in N$ and using (1), we get

$$n v_i(A_i) \geq \sum_{j=1}^n v_i(A_j) = 1,$$

hence $v_i(A_i) \geq \frac{1}{n}$.

Since i was arbitrary, A is **proportional**.

This also proves the “ $\Rightarrow$ ” direction for 2 agents.

Proof: Proportionality $\Rightarrow$ EF for $n = 2$

Proof: Proportionality $\Rightarrow$ EF for $n = 2$

Let $A = (A_1, A_2)$ be proportional, and let $i \in \{1, 2\}$ and j be the other agent.

By (1) and proportionality,

$$v_i(A_j) = 1 - v_i(A_i) \leq 1 - \frac{1}{2} = \frac{1}{2} \leq v_i(A_i).$$

Proof: Proportionality $\Rightarrow$ EF for $n = 2$

Let $A = (A_1, A_2)$ be proportional, and let $i \in \{1, 2\}$ and j be the other agent.

By (1) and proportionality,

$$v_i(A_j) = 1 - v_i(A_i) \leq 1 - \frac{1}{2} = \frac{1}{2} \leq v_i(A_i).$$

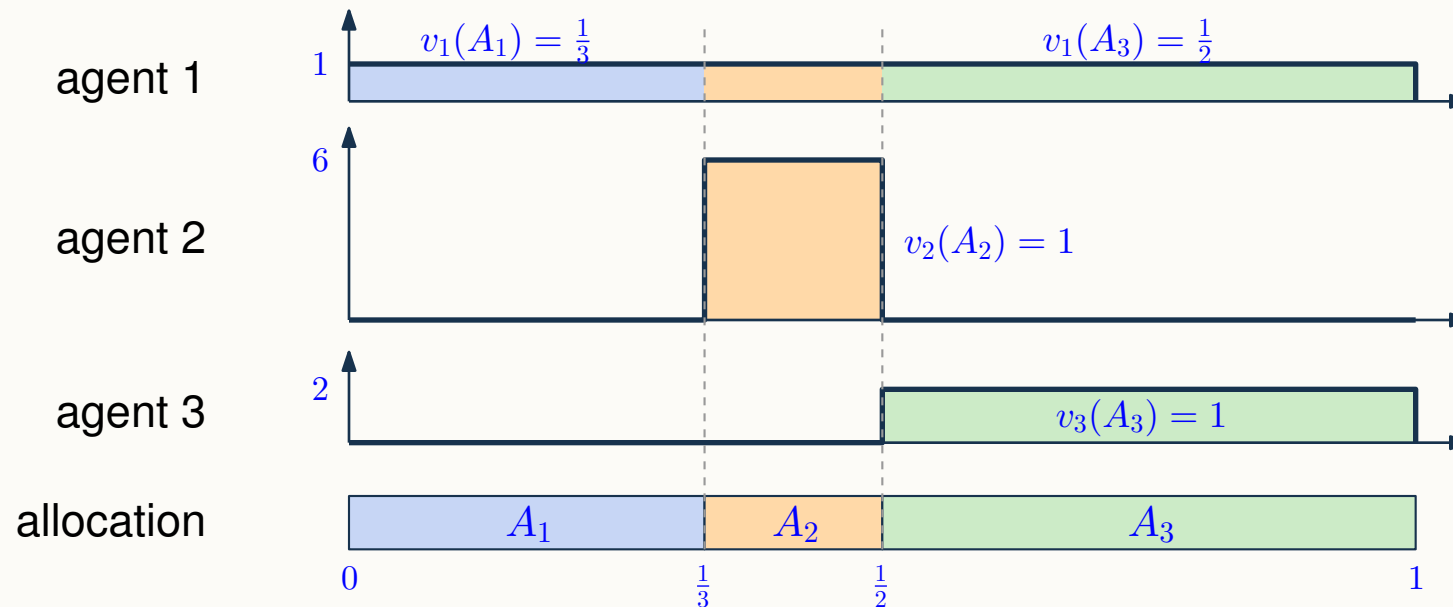
Hence no agent envies the other, so A is **envy-free**.

Proof: Proportionality $\nRightarrow$ EF for $n = 3$

Proof: Proportionality $\nRightarrow$ EF for $n = 3$

Let agent 1 have the uniform density on $[0, 1]$, agent 2 the uniform density on $[\frac{1}{3}, \frac{1}{2}]$, and agent 3 the uniform density on $[\frac{1}{2}, 1]$.

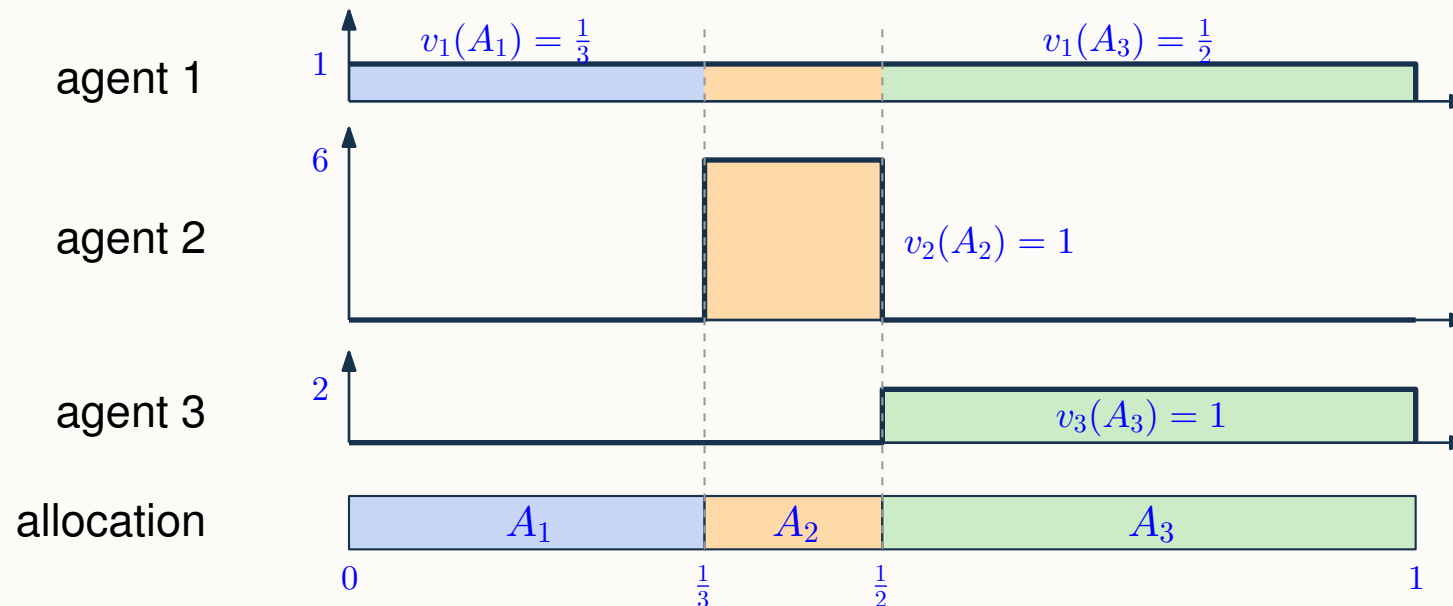
Consider $A_1 = [0, \frac{1}{3}]$, $A_2 = [\frac{1}{3}, \frac{1}{2}]$, $A_3 = [\frac{1}{2}, 1]$.



Proof: Proportionality $\nRightarrow$ EF for $n = 3$

Let agent 1 have the uniform density on $[0, 1]$, agent 2 the uniform density on $[\frac{1}{3}, \frac{1}{2}]$, and agent 3 the uniform density on $[\frac{1}{2}, 1]$.

Consider $A_1 = [0, \frac{1}{3}]$, $A_2 = [\frac{1}{3}, \frac{1}{2}]$, $A_3 = [\frac{1}{2}, 1]$.

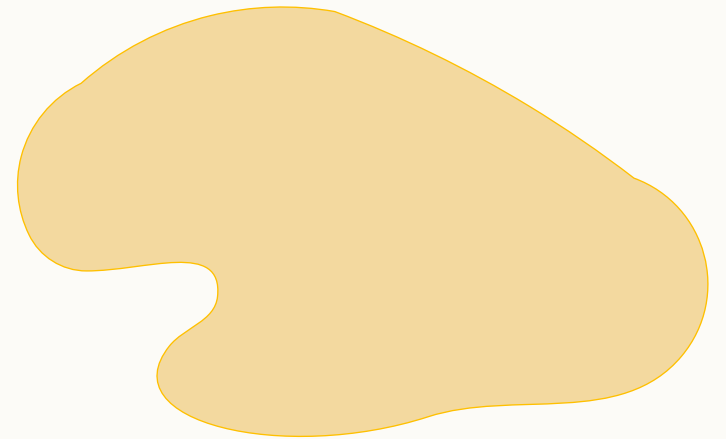


Then $v_1(A_1) = \frac{1}{3}$ and $v_2(A_2) = v_3(A_3) = 1$, so A is **proportional**.

However, $v_1(A_3) = \frac{1}{2} > \frac{1}{3} = v_1(A_1)$, so agent 1 envies agent 3 and A is **not envy-free**. $\square$

Cut & Choose: Dividing a Cake between 2 Agents

This method is very old (Genesis 13).

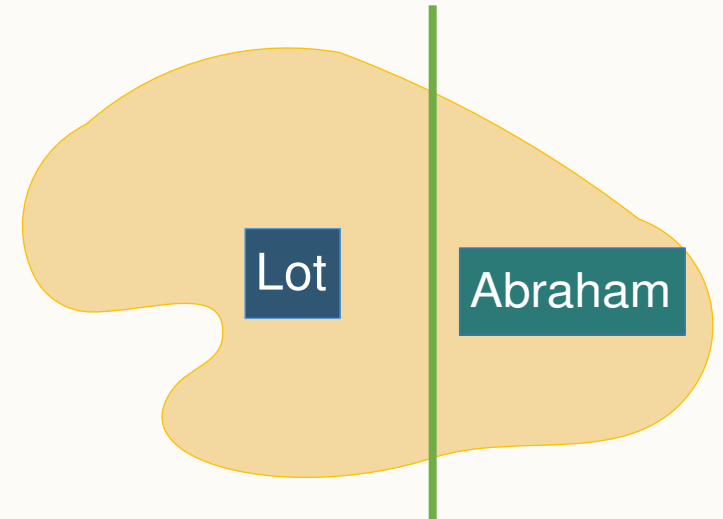


Cut & Choose: Dividing a Cake between 2 Agents

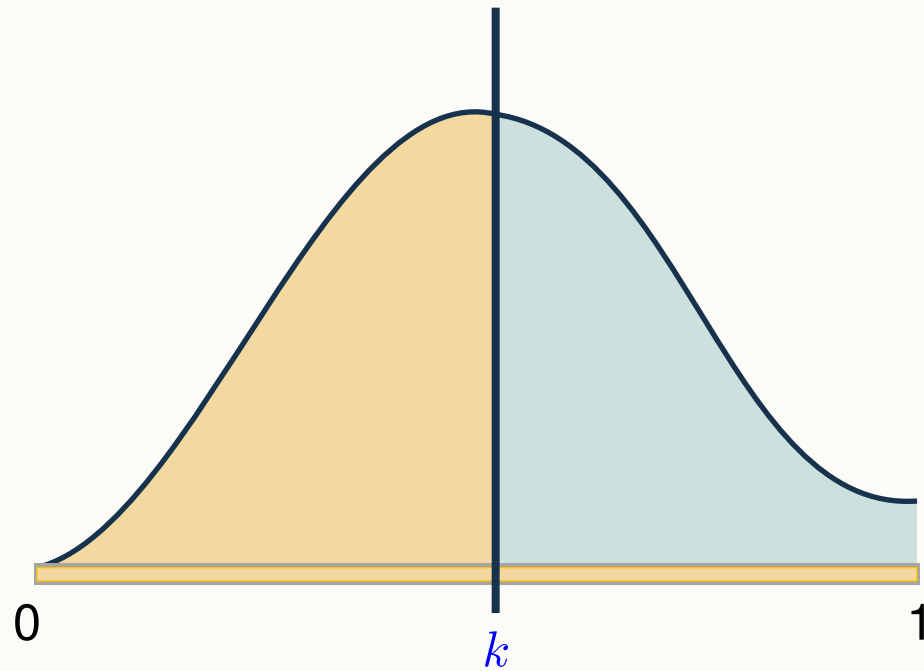
This method is very old (Genesis 13).

First, we ask **agent 1** to **divide** the cake into two equal parts (according to their own valuation); then **agent 2 chooses** the piece that is more valuable to them, and the remaining piece goes to agent 1.

In the picture: Abraham (agent 1) cuts, Lot (agent 2) chooses.



Cut & Choose: Properties



$$v_1([0, k]) = \frac{1}{2} = v_1([k, 1])$$

Properties:

Proportional

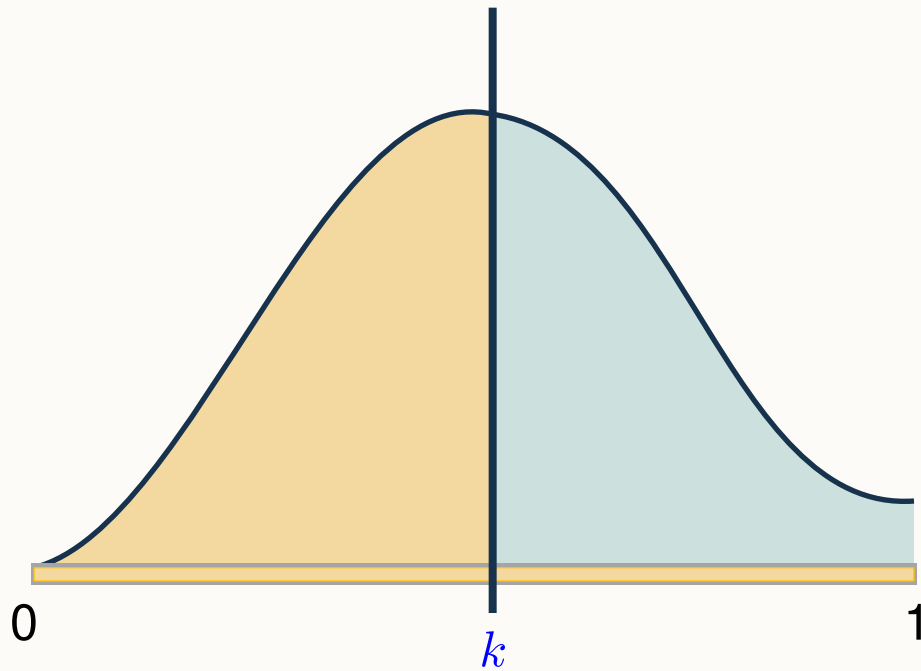
Envy-free

Equitable

Contiguous

Pareto optimal

Cut & Choose: Properties



$$v_1([0, k]) = \frac{1}{2} = v_1([k, 1])$$

Properties:

Proportional



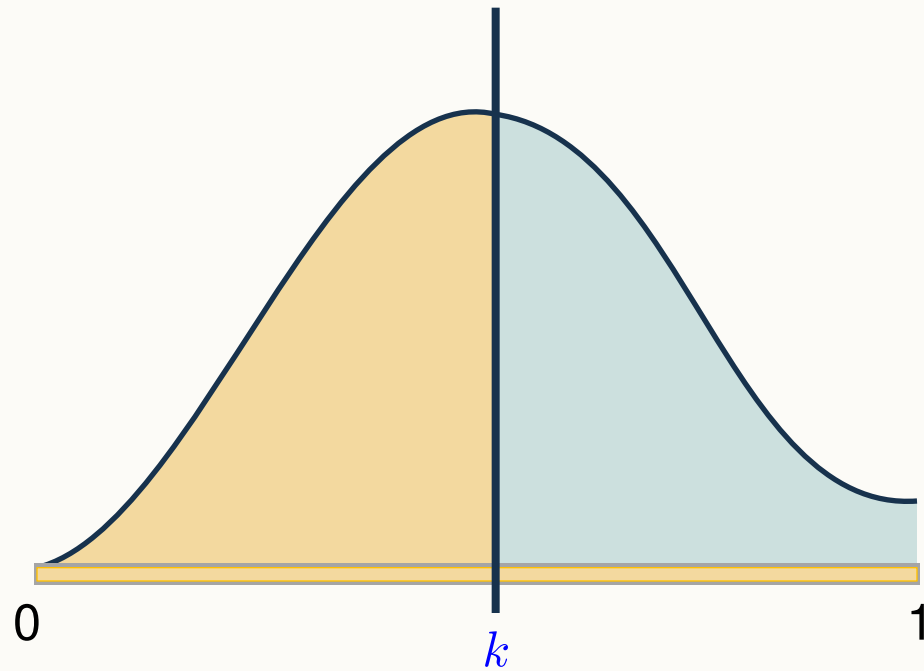
Envy-free

Equitable

Contiguous

Pareto optimal

Cut & Choose: Properties



$$v_1([0, k]) = \frac{1}{2} = v_1([k, 1])$$

Properties:

Proportional ✓

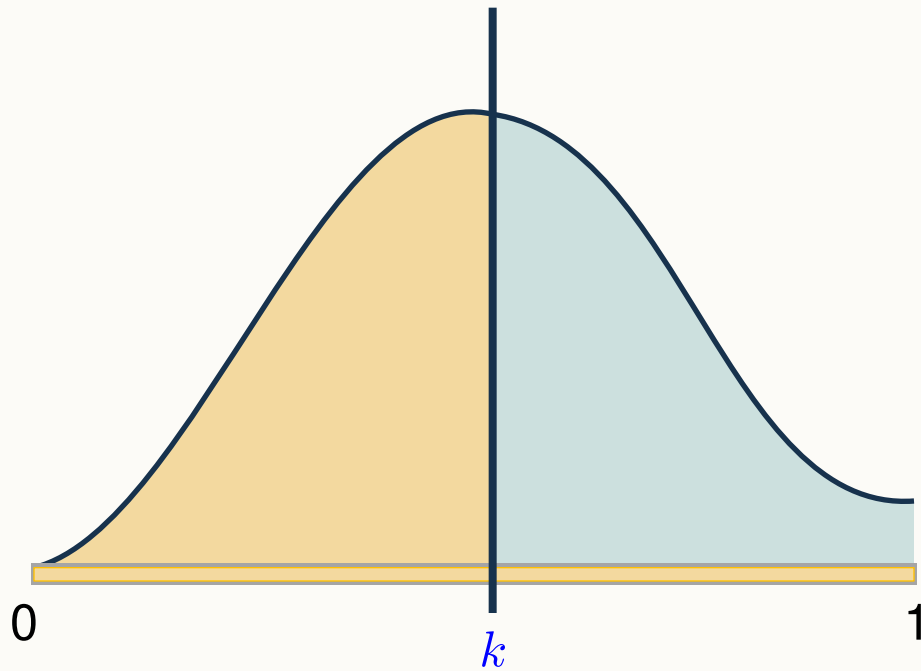
Envy-free ✓

Equitable

Contiguous

Pareto optimal

Cut & Choose: Properties

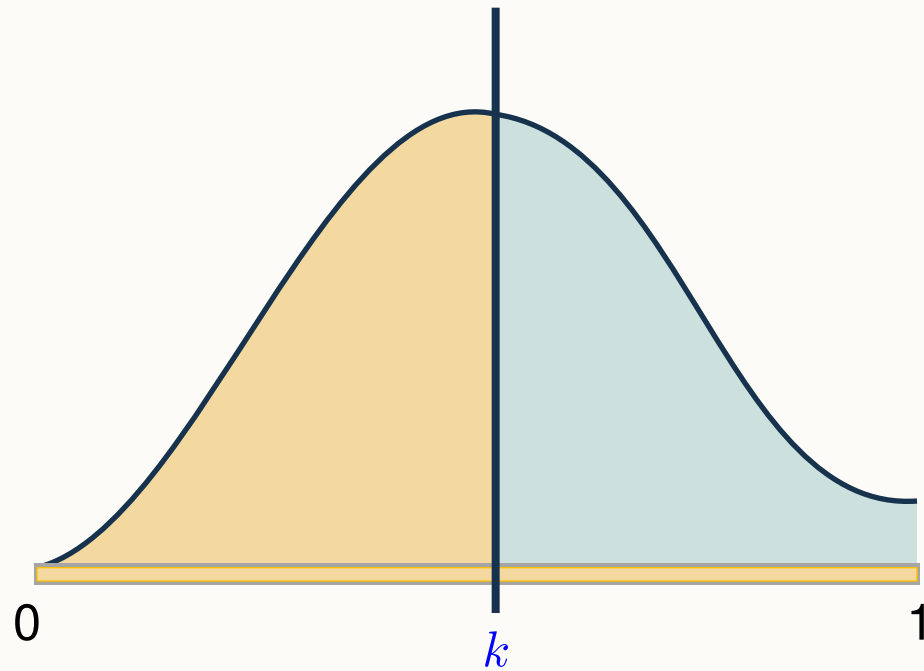


$$v_1([0, k]) = \frac{1}{2} = v_1([k, 1])$$

Properties:

Proportional	✓
Envy-free	✓
Equitable	✗
Contiguous	
Pareto optimal	

Cut & Choose: Properties

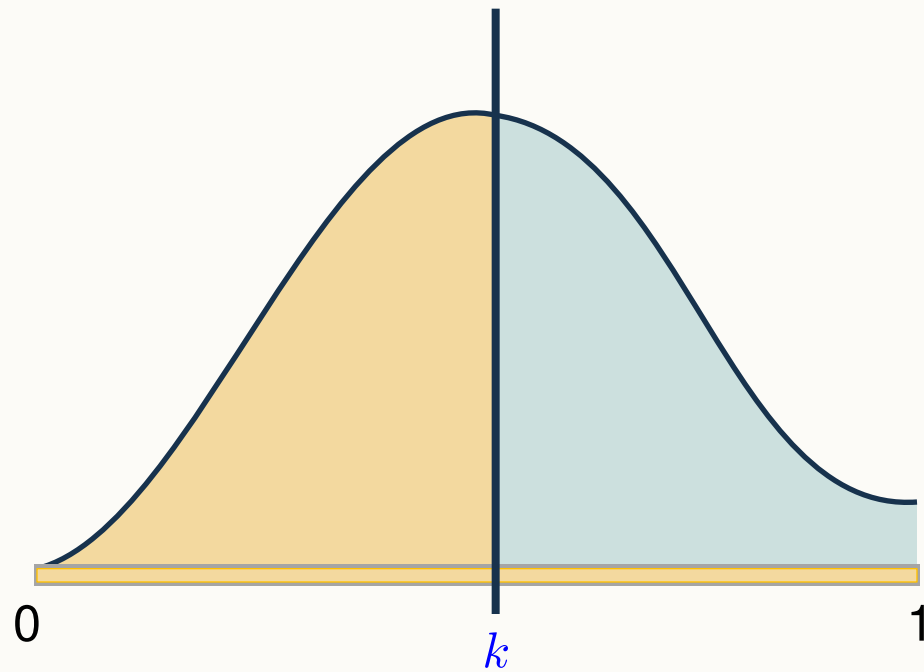


$$v_1([0, k]) = \frac{1}{2} = v_1([k, 1])$$

Properties:

Proportional	✓
Envy-free	✓
Equitable	✗
Contiguous	✓
Pareto optimal	

Cut & Choose: Properties



$$v_1([0, k]) = \frac{1}{2} = v_1([k, 1])$$

Properties:

Proportional	✓
Envy-free	✓
Equitable	✗
Contiguous	✓
Pareto optimal	✗

Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

A referee moves a knife from the beginning of the cake (point 0) to the right.

Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

A referee moves a knife from the beginning of the cake (point 0) to the right.

This continues until one agent says “stop”, meaning that the part to the left of the knife is worth $\frac{1}{n}$ to them.

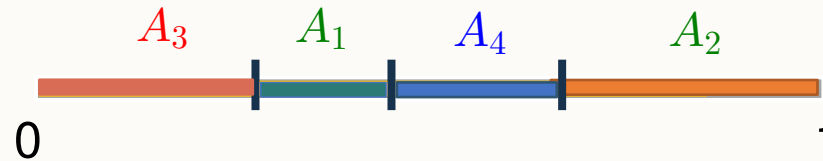
Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

A referee moves a knife from the beginning of the cake (point 0) to the right.

This continues until one agent says “stop”, meaning that the part to the left of the knife is worth $\frac{1}{n}$ to them.

We cut that piece, give it to that agent, and continue the process with the remaining players.



Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

A referee moves a knife from the beginning of the cake (point 0) to the right.

This continues until one agent says “stop”, meaning that the part to the left of the knife is worth $\frac{1}{n}$ to them.

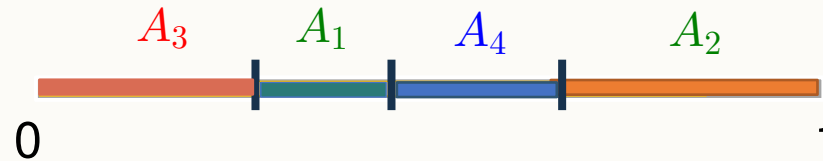
We cut that piece, give it to that agent, and continue the process with the remaining players.

Properties:

Proportional
Contiguous

Envy-free
Pareto optimal

Equitable



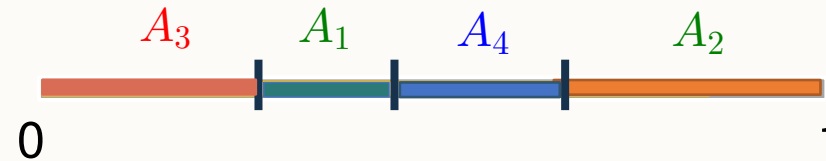
Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

A referee moves a knife from the beginning of the cake (point 0) to the right.

This continues until one agent says “stop”, meaning that the part to the left of the knife is worth $\frac{1}{n}$ to them.

We cut that piece, give it to that agent, and continue the process with the remaining players.



Properties:	Proportional	✓	Envy-free	Equitable
	Contiguous		Pareto optimal	

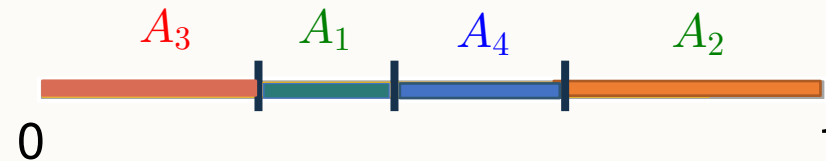
Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

A referee moves a knife from the beginning of the cake (point 0) to the right.

This continues until one agent says “stop”, meaning that the part to the left of the knife is worth $\frac{1}{n}$ to them.

We cut that piece, give it to that agent, and continue the process with the remaining players.



Properties:	Proportional	✓	Envy-free	✗	Equitable
	Contiguous		Pareto optimal		

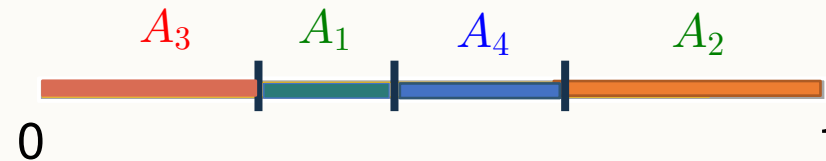
Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

A referee moves a knife from the beginning of the cake (point 0) to the right.

This continues until one agent says “stop”, meaning that the part to the left of the knife is worth $\frac{1}{n}$ to them.

We cut that piece, give it to that agent, and continue the process with the remaining players.



Properties:	Proportional	✓	Envy-free	✗	Equitable	✗
	Contiguous		Pareto optimal			

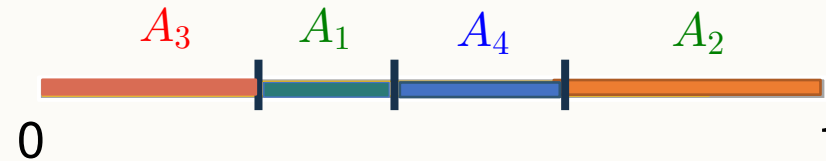
Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

A referee moves a knife from the beginning of the cake (point 0) to the right.

This continues until one agent says “stop”, meaning that the part to the left of the knife is worth $\frac{1}{n}$ to them.

We cut that piece, give it to that agent, and continue the process with the remaining players.



Properties:	Proportional	✓	Envy-free	✗	Equitable	✗
	Contiguous	✓	Pareto optimal			

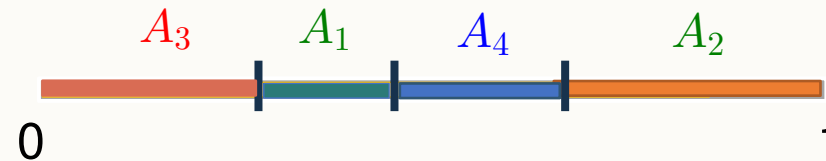
Moving-Knife Procedure (Dubins & Spanier, 1961)

A procedure for **proportional** division among n agents.

A referee moves a knife from the beginning of the cake (point 0) to the right.

This continues until one agent says “stop”, meaning that the part to the left of the knife is worth $\frac{1}{n}$ to them.

We cut that piece, give it to that agent, and continue the process with the remaining players.

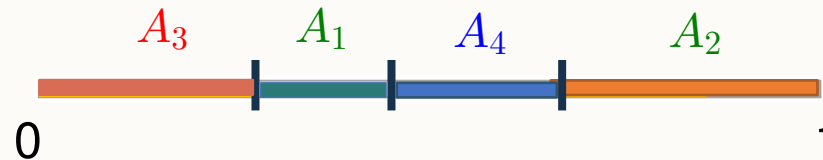


Properties:	Proportional	✓	Envy-free	✗	Equitable	✗
	Contiguous	✓	Pareto optimal	✗		

Moving Knives and Computers

A fundamental problem with the previous procedures is their **continuous** nature.

Basically, in computer science continuous movements cannot be simulated without a loss of precision.



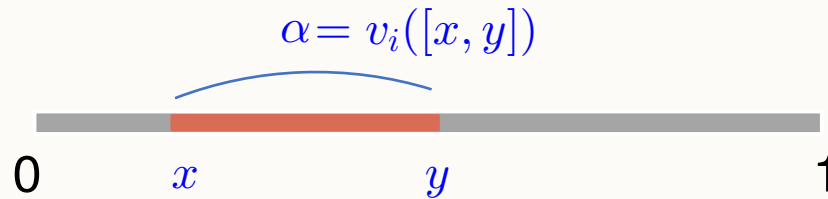
The Robertson–Webb Model (1998)

In this model we may use only **two types of query**.

The Robertson–Webb Model (1998)

In this model we may use only **two types of query**.

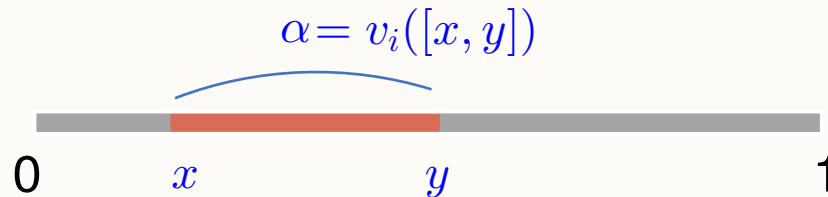
- $\text{Cut}_i(x, \alpha)$: ask person i to specify the point y of the cake such that $v_i([x, y]) = \alpha$, where y lies to the right of x . (Note: this query does *not* actually cut the cake!)



The Robertson–Webb Model (1998)

In this model we may use only **two types of query**.

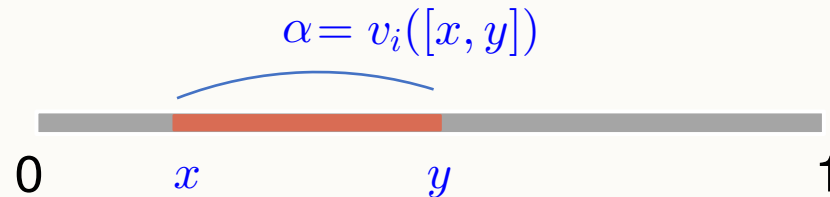
- $\text{Cut}_i(x, \alpha)$: ask person i to specify the point y of the cake such that $v_i([x, y]) = \alpha$, where y lies to the right of x . (Note: this query does *not* actually cut the cake!)
- $\text{Eval}_i(x, y)$: ask person i to specify the value of the interval $[x, y]$, i.e. what $v_i([x, y])$ is equal to.



The Robertson–Webb Model (1998)

In this model we may use only **two types of query**.

- $\text{Cut}_i(x, \alpha)$: ask person i to specify the point y of the cake such that $v_i([x, y]) = \alpha$, where y lies to the right of x . (Note: this query does *not* actually cut the cake!)
- $\text{Eval}_i(x, y)$: ask person i to specify the value of the interval $[x, y]$, i.e. what $v_i([x, y])$ is equal to.

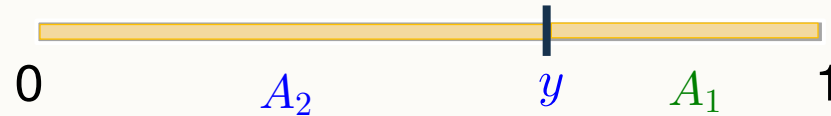


A cake-cutting algorithm that uses only these two types of query is called a **protocol**.

Implementing Cut & Choose in the Robertson–Webb Model

Implementing Cut & Choose in the Robertson–Webb Model

1. $\text{Cut}_1(0, \frac{1}{2}) = y$
2. $\text{Eval}_2(0, y)$

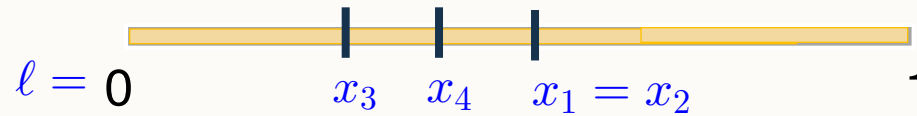


With these two queries, cut & choose can be simulated completely.

Implementing Dubins–Spanier in the Robertson–Webb Model

Implementing Dubins–Spanier in the Robertson–Webb Model

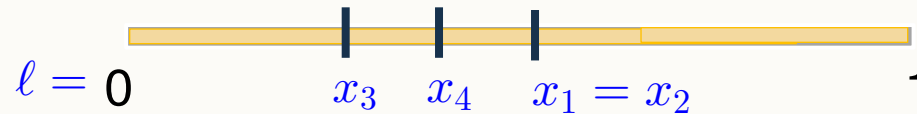
1. First, we make n queries of the form $\text{Cut}_i(0, \frac{1}{n})$. Let y be the smallest value among the answers.



$$W = \{1, 2, 3, 4\}$$

Implementing Dubins–Spanier in the Robertson–Webb Model

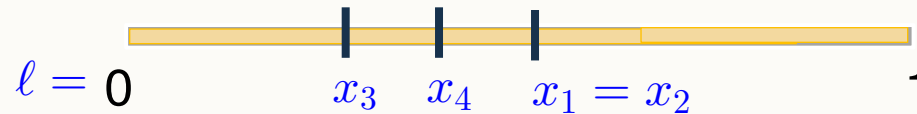
1. First, we make n queries of the form $\text{Cut}_i(0, \frac{1}{n})$. Let y be the smallest value among the answers.
2. We give the interval $[0, y]$ to the person whose answer was y . In the next step we make $n - 1$ queries of the form $\text{Cut}_i(y, \frac{1}{n})$, $\dots$. We continue in this way until everyone has received a share.



$$W = \{1, 2, 3, 4\}$$

Implementing Dubins–Spanier in the Robertson–Webb Model

1. First, we make n queries of the form $\text{Cut}_i(0, \frac{1}{n})$. Let y be the smallest value among the answers.
2. We give the interval $[0, y]$ to the person whose answer was y . In the next step we make $n - 1$ queries of the form $\text{Cut}_i(y, \frac{1}{n})$, $\dots$. We continue in this way until everyone has received a share.

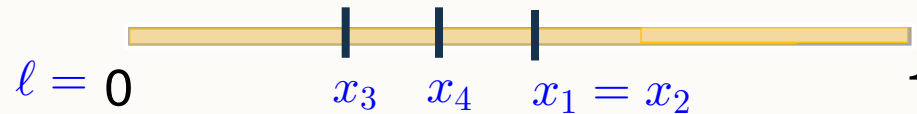


$$W = \{1, 2, 3, 4\}$$

Number of queries: $n + (n - 1) + (n - 2) + \dots + 2 = \mathcal{O}(n^2)$.

Implementing Dubins–Spanier in the Robertson–Webb Model

1. First, we make n queries of the form $\text{Cut}_i(0, \frac{1}{n})$. Let y be the smallest value among the answers.
2. We give the interval $[0, y]$ to the person whose answer was y . In the next step we make $n - 1$ queries of the form $\text{Cut}_i(y, \frac{1}{n})$, ... We continue in this way until everyone has received a share.



$$W = \{1, 2, 3, 4\}$$

Number of queries: $n + (n - 1) + (n - 2) + \cdots + 2 = \mathcal{O}(n^2)$.

Can we obtain a proportional allocation with **fewer queries**?

Even–Paz Algorithm (1984)

Assume n is a power of 2. A referee moves a knife from the beginning of the cake to the right.

Even–Paz Algorithm (1984)

Assume n is a power of 2. A referee moves a knife from the beginning of the cake to the right.

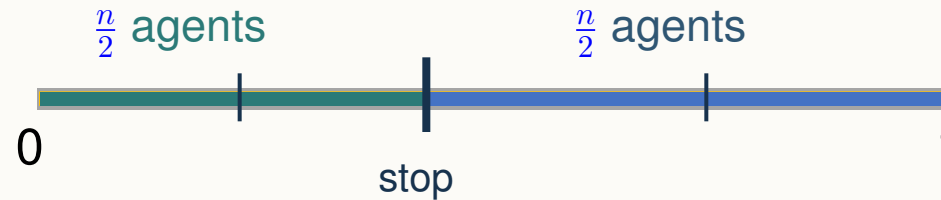
This continues until $\frac{n}{2}$ of the individuals shout “stop”, meaning that the part to the left of the knife is worth $\frac{1}{2}$ of the cake to them.

Even–Paz Algorithm (1984)

Assume n is a power of 2. A referee moves a knife from the beginning of the cake to the right.

This continues until $\frac{n}{2}$ of the individuals shout “stop”, meaning that the part to the left of the knife is worth $\frac{1}{2}$ of the cake to them.

We halve the cake at this point and solve the problem recursively for the two sides.

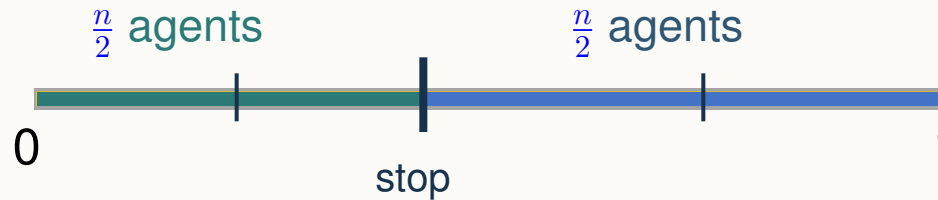


Even–Paz Algorithm (1984)

Assume n is a power of 2. A referee moves a knife from the beginning of the cake to the right.

This continues until $\frac{n}{2}$ of the individuals shout “stop”, meaning that the part to the left of the knife is worth $\frac{1}{2}$ of the cake to them.

We halve the cake at this point and solve the problem recursively for the two sides.



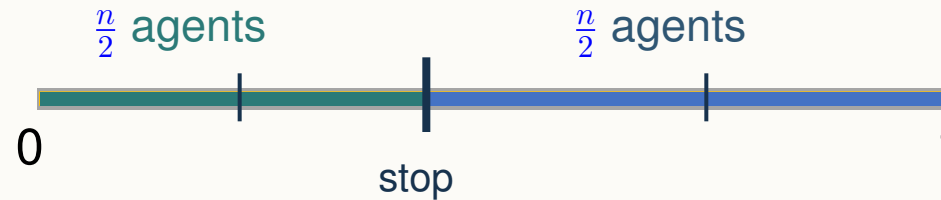
Properties:	Proportional	Envy-free	Equitable
	Contiguous	Pareto optimal	

Even–Paz Algorithm (1984)

Assume n is a power of 2. A referee moves a knife from the beginning of the cake to the right.

This continues until $\frac{n}{2}$ of the individuals shout “stop”, meaning that the part to the left of the knife is worth $\frac{1}{2}$ of the cake to them.

We halve the cake at this point and solve the problem recursively for the two sides.



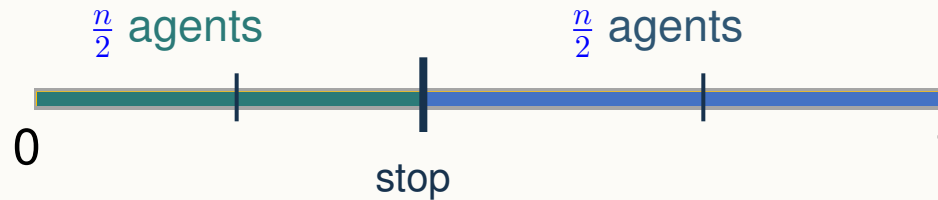
Properties:	Proportional	✓	Envy-free	Equitable
	Contiguous		Pareto optimal	

Even–Paz Algorithm (1984)

Assume n is a power of 2. A referee moves a knife from the beginning of the cake to the right.

This continues until $\frac{n}{2}$ of the individuals shout “stop”, meaning that the part to the left of the knife is worth $\frac{1}{2}$ of the cake to them.

We halve the cake at this point and solve the problem recursively for the two sides.



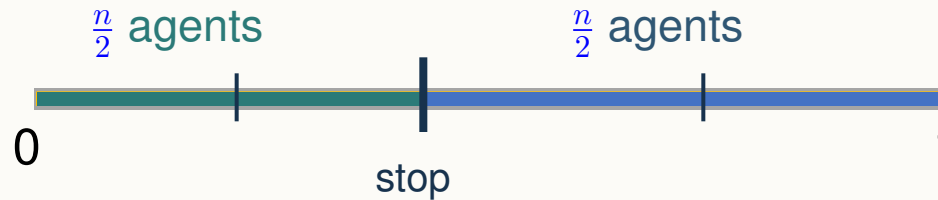
Properties:	Proportional	✓	Envy-free	✗	Equitable
	Contiguous		Pareto optimal		

Even–Paz Algorithm (1984)

Assume n is a power of 2. A referee moves a knife from the beginning of the cake to the right.

This continues until $\frac{n}{2}$ of the individuals shout “stop”, meaning that the part to the left of the knife is worth $\frac{1}{2}$ of the cake to them.

We halve the cake at this point and solve the problem recursively for the two sides.



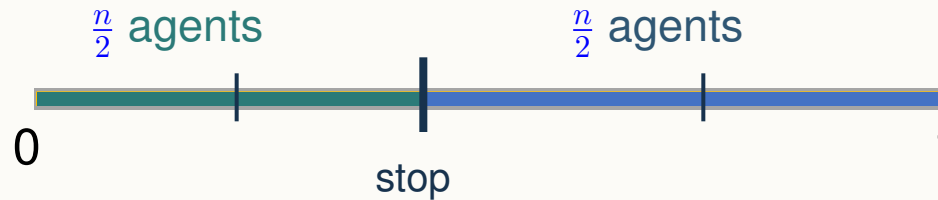
Properties:	Proportional	✓	Envy-free	✗	Equitable	✗
	Contiguous		Pareto optimal			

Even–Paz Algorithm (1984)

Assume n is a power of 2. A referee moves a knife from the beginning of the cake to the right.

This continues until $\frac{n}{2}$ of the individuals shout “stop”, meaning that the part to the left of the knife is worth $\frac{1}{2}$ of the cake to them.

We halve the cake at this point and solve the problem recursively for the two sides.



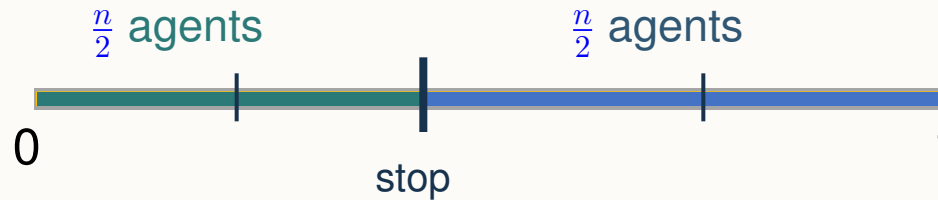
Properties:	Proportional	✓	Envy-free	✗	Equitable	✗
	Contiguous	✓	Pareto optimal			

Even–Paz Algorithm (1984)

Assume n is a power of 2. A referee moves a knife from the beginning of the cake to the right.

This continues until $\frac{n}{2}$ of the individuals shout “stop”, meaning that the part to the left of the knife is worth $\frac{1}{2}$ of the cake to them.

We halve the cake at this point and solve the problem recursively for the two sides.

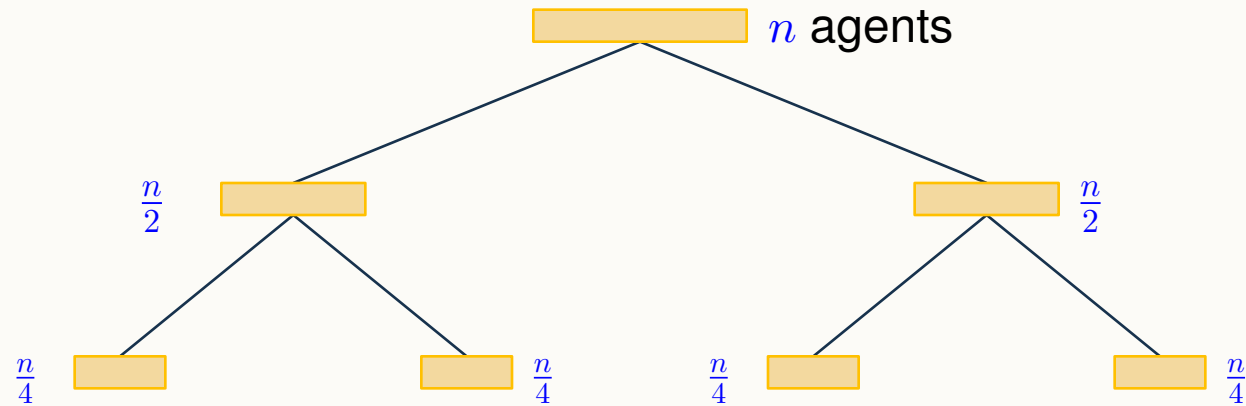


Properties:	Proportional	✓	Envy-free	✗	Equitable	✗
	Contiguous	✓	Pareto optimal	✗		

Implementing Even–Paz in the Robertson–Webb Model

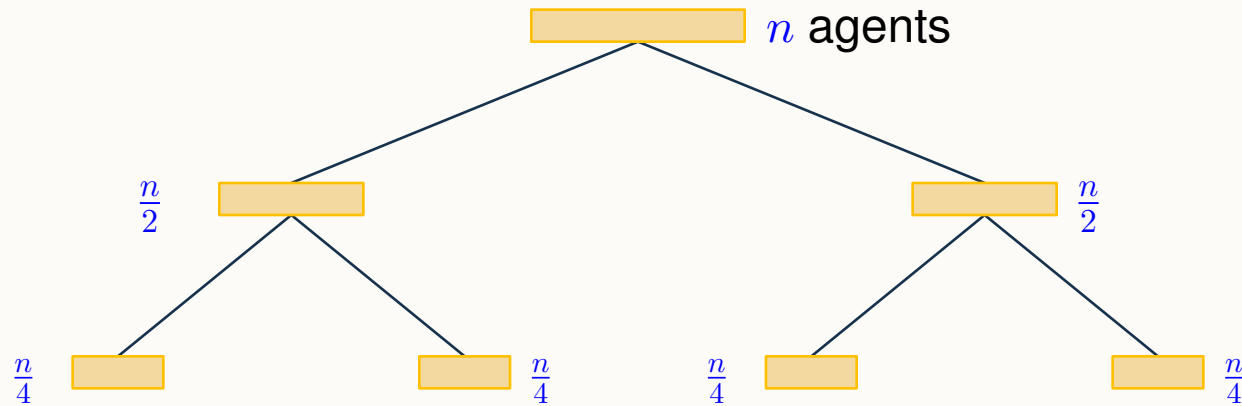
Implementing Even–Paz in the Robertson–Webb Model

1. $n \times \text{Cut}_i(0, \frac{1}{2}) \rightsquigarrow y$
2. $\frac{n}{2} \times \text{Cut}_i(0, \frac{1}{4}), \quad \frac{n}{2} \times \text{Cut}_i(y, \frac{1}{4})$
- ⋮



Implementing Even–Paz in the Robertson–Webb Model

1. $n \times \text{Cut}_i(0, \frac{1}{2}) \rightsquigarrow y$
2. $\frac{n}{2} \times \text{Cut}_i(0, \frac{1}{4}), \quad \frac{n}{2} \times \text{Cut}_i(y, \frac{1}{4})$
- ⋮



Number of queries: $T(n) = 2T(n/2) + n \longrightarrow T(n) = \mathcal{O}(n \log n).$

Query Complexity of Proportionality

So the Even–Paz procedure is better than the Dubins–Spanier procedure in terms of the number of queries.

Query Complexity of Proportionality

So the Even–Paz procedure is better than the Dubins–Spanier procedure in terms of the number of queries.

The question that now arises is: can a proportional division be achieved with **even fewer queries**?

Query Complexity of Proportionality

So the Even–Paz procedure is better than the Dubins–Spanier procedure in terms of the number of queries.

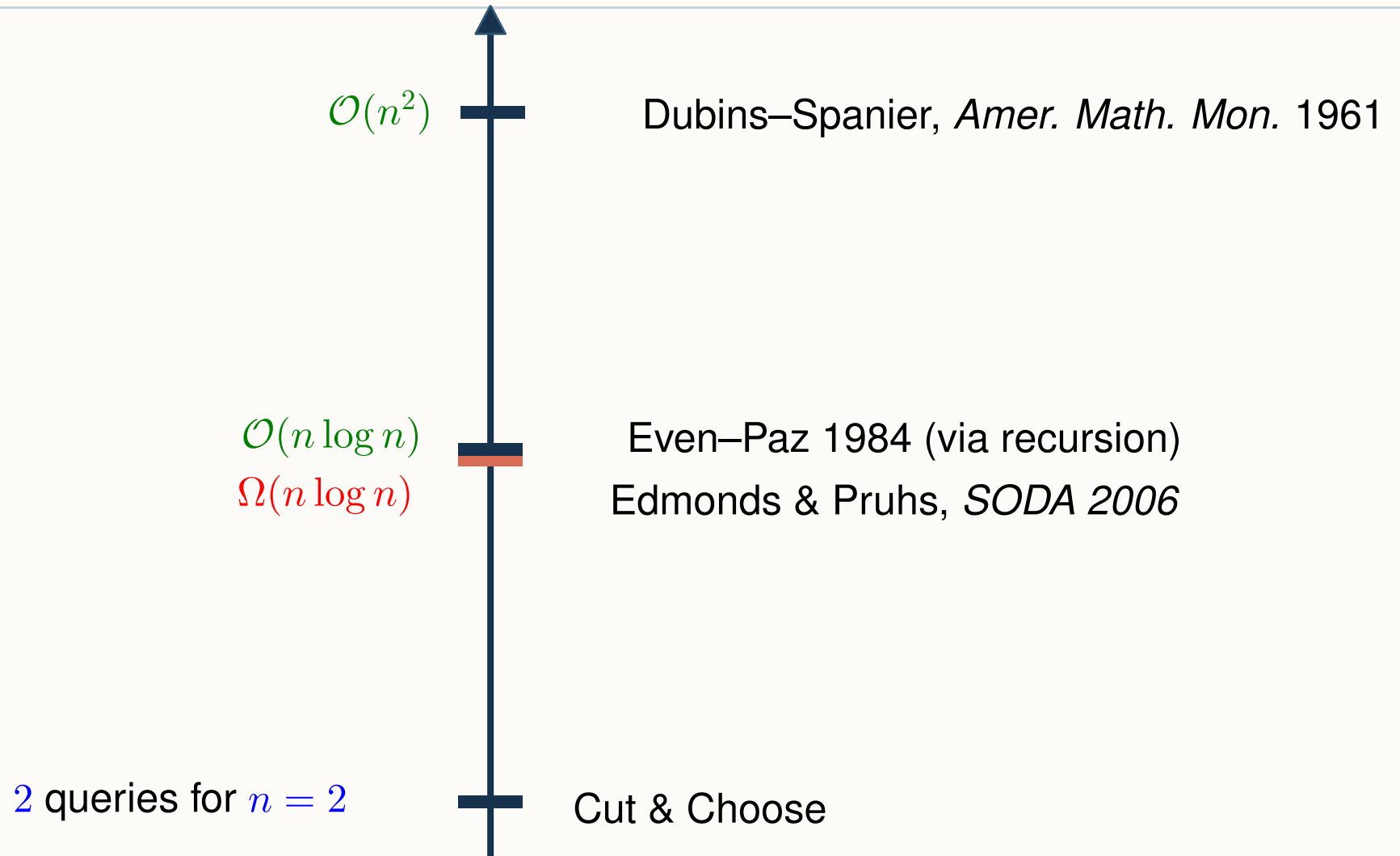
The question that now arises is: can a proportional division be achieved with **even fewer queries**?

Theorem (Edmonds & Pruhs 2006).

Every protocol for proportional cake division requires $\Omega(n \log n)$ queries.

(SODA '06: *Cake cutting really is not a piece of cake.*)

Query Complexity of Proportionality



Exercises

- Q 1.** Give a moving-knife procedure that outputs a contiguous proportional equitable allocation for two agents.
- Q 2.** Give a Robertson–Webb protocol that outputs a proportional allocation using $O(n \log n)$ queries for arbitrary n (not necessarily a power of 2).
- Q 3.** For any arbitrary $n > 3$, prove that proportionality does not imply envy-freeness.