Improving Distortion via Queries

Georgios Birmpas

Sapienza University of Rome

Part I

Improving **Distortion** via Queries

How we decide?

- Suppose that a set of people will vote for the winner in a hypothetical battle

How we decide?

- Suppose that a set of people will vote for the winner in a hypothetical battle
- The contenders are:

How we decide?

- Suppose that a set of people will vote for the winner in a hypothetical battle
- The contenders are:

How we decide?

- Suppose that a set of people will vote for the winner in a hypothetical battle
- The contenders are:

Intensity of preferences

- Member 1: Close but I was always afraid of bats, so I will go for Batman

Intensity of preferences

- Member 1: Close but I was always afraid of bats, so I will go for Batman
- Member 2: I have not read the comics but I saw BvS and Batman won the fight, although it was close. So, I suppose Batman wins

Intensity of preferences

- Member 1: Close but I was always afraid of bats, so I will go for Batman
- Member 2: I have not read the comics but I saw BvS and Batman won the fight, although it was close. So, I suppose Batman wins

...

- Member n/2+1: It is close if you consider that Superman is weak to kryptonite. I will vote for Batman

Intensity of preferences

- Member 1: Close but I was always afraid of bats, so I will go for Batman
- Member 2: I have not read the comics but I saw BvS and Batman won the fight, although it was close. So, I suppose Batman wins

...

- Member n/2+1: It is close if you consider that Superman is weak to kryptonite. I will vote for Batman
- Member n/2+2: Come on people are you serious? Superman wins!!

Intensity of preferences

- Member 1: Close but I was always afraid of bats, so I will go for Batman
- Member 2: I have not read the comics but I saw BvS and Batman won the fight, although it was close. So, I suppose Batman wins

…

- Member n/2+1: It is close if you consider that Superman is weak to kryptonite. I will vote for Batman
- Member n/2+2: Come on people are you serious? Superman wins!!

…

- Member n: This is not even a contest… Superman would destroy him

Intensity of preferences

- Batman is the winner according to the majority

Intensity of preferences

- Batman is the winner according to the majority
- However, the outcome may would have been different if we had information about the intensity of the preferences

The setting

- A set of n agents N and a set of m alternatives A
- Each agent $i \in N$ has a value v_{ix} for every alternative $x \in A$ (cardinal preferences)
  - Captures how intense a preference is

The setting

- The agents submit a preference ranking over the alternatives that is consistent to their values (ordinal preferences)
- An ordinal mechanism takes these rankings as an input
  - Outputs a single alternative as the winner

Utilitarian Social Choice

- Objective: Maximize the social welfare, i.e., select the alternative x that maximizes

$$\sum_{i \in N} v_{ix}$$

Utilitarian Social Choice

- Objective: Maximize the social welfare, i.e., select the alternative *x* that maximizes

$$\sum_{i \in N} v_{ix}$$

- Expresses how the society feels about the produced outcome

Utilitarian Social Choice

- Objective: Maximize the social welfare, i.e., select the alternative *x* that maximizes

$$\sum_{i \in N} v_{ix}$$

- This is easy to achieve when the cardinal preferences are known

Utilitarian Social Choice

- Objective: Maximize the social welfare, i.e., select the alternative x that maximizes

$$\sum_{i \in N} v_{ix}$$

- It may not be possible when only the ordinal preferences are known, due to the lack of information

Distortion

- The distortion of an *ordinal* mechanism M is the maximum ratio (over all possible inputs) of the maximum possible social welfare, over the social welfare achieved by the mechanism
  - Defined by Procaccia and Rosenschein [2006]

Distortion

- The distortion of an *ordinal* mechanism *M* is the maximum ratio (over all possible inputs) of the maximum possible social welfare, over the social welfare achieved by the mechanism
  - Defined by Procaccia and Rosenschein [2006]
- Expresses the guarantees of the mechanism in the worst-case scenario

Distortion

- Remark 1: A mechanism that has access to the cardinal information can obviously achieve a distortion of 1

Distortion

- Remark 1: A mechanism that has access to the cardinal information can obviously achieve a distortion of 1
- Remark 2: A mechanism that has access only to the ordinal information may elect an alternative that is different from the optimal
  - The distortion captures how good-bad is this alternative in comparison with the optimal one

Distortion

- Remark 1: A mechanism that has access to the cardinal information can obviously achieve a distortion of 1
- Remark 2: A mechanism that has access only to the ordinal information may elect an alternative that is different from the optimal
  - The distortion captures how good-bad is this alternative in comparison with the optimal one
- Remark 3: The distortion is usually expressed as a function of m (the number of alternatives)

What we know?

- Ordinal Deterministic Mechanisms
- Ordinal Randomized Mechanisms

What we know?

- Ordinal Deterministic Mechanisms
- Ordinal Randomized Mechanisms
  - There is randomness on how the mechanism elects the winner
  - The guarantees of the mechanism are in expectation

What we know?

- Ordinal Deterministic Mechanisms
- Ordinal Randomized Mechanisms
  - Unit-Sum Assumption: The values of an agent over the alternatives sum up to 1

What we know?

- Ordinal Deterministic Mechanisms
- Ordinal Randomized Mechanisms
  - Unit-Sum Assumption: The values of an agent over the alternatives sum up to 1
    - An agent assigns to each alternative a percentage that expresses how much he likes him
    - Without any normalization assumption the distortion can be arbitrarily bad

What we know?

- Ordinal Deterministic Mechanisms
  - The distortion of *Plurality* for unit-sum valuations is $O(m^2)$ [Caragiannis and Procaccia 2011]
  - The distortion of *any* deterministic ordinal mechanism for unit-sum valuations is $\Omega(m^2)$ [Caragiannis et al. 2017]

What we know?

- Ordinal Randomized Mechanisms
  - There is an ordinal randomized mechanism with $O(\sqrt{\mathrm{m}} \cdot \log^* m)$ distortion for unit-sum valuations [Boutilier et al. 2015]
  - The distortion of *any* randomized ordinal mechanism for unit-sum valuations is $\Omega(\sqrt{m})$ [Boutilier et al. 2015]

What we know?

- Most of the work on distortion regards ordinal mechanisms

Ordinal Preferences

Cardinal Values

Deterministic: $O(m^2)$
Randomized: $O(\sqrt{\mathrm{m}} \cdot \log^* m)$

Distortion=1

Question

- How can we improve the distortion?

Ordinal Preferences

Cardinal Values

Deterministic: $O(m^2)$
Randomized: $O(\sqrt{\mathrm{m}} \cdot \log^* m)$

Distortion=1

Part II

Improving Distortion via Queries

An idea

- What if we could elicit some cardinal information via simple queries?

An idea

- What if we could elicit some cardinal information via simple queries?
  - What is your value for alternative x?

An idea

- What if we could elicit some cardinal information via simple queries?
  - What is your value for alternative x?
  - Do you prefer alternative x by at least twice as much as alternative y?

Queries

- Value Query: Present agent i with an alternative x, and ask the agent for his value v_{ix}

Queries

- Value Query: Present agent *i* with an alternative *x*, and ask the agent for his value v_{ix}
- Comparison Query: Present agent *i* with two alternatives *x* and *y*, and a number *d*, and ask the agent whether $v_{ix} \geq d \cdot v_{iy}$

Queries

- Value Query: Present agent *i* with an alternative *x*, and ask the agent for his value v_{ix}
- Comparison Query: Present agent *i* with two alternatives *x* and *y*, and a number *d*, and ask the agent whether $v_{ix} \geq d \cdot v_{iy}$
  - A weaker form of query
  - Easier for an agent to answer

Mechanisms

Mechanism $\boldsymbol{M} = (Q, R)$

- Algorithm Q

- Modified voting rule R

Mechanisms

Mechanism $\boldsymbol{M} = (Q, R)$

- Algorithm Q
  - Input: the ordinal profile $\succ$
  - Makes a set of (value or comparison) queries per agent
  - Output: the answers to the queries
- Modified voting rule R

Mechanisms

Mechanism $\boldsymbol{M} = (Q, R)$

- Algorithm Q
  - Input: the ordinal profile $\succ$
  - Makes a set of (value or comparison) queries per agent
  - Output: the answers to the queries
- Modified voting rule R
  - Input: the ordinal profile $\succ$, and the answers to the queries $Q(\succ)$
  - Output: a single alternative

Improving distortion via queries

Ordinal Preferences

What lies in between?

Cardinal Values

Deterministic: $O(m^2)$
Randomized: $O(\sqrt{\mathrm{m}} \cdot \log^* m)$

Distortion=1

Improving distortion via queries

Let's try this again

Ordinal Preferences

Number of queries per agent

Cardinal Values

Deterministic: $O(m^2)$
Randomized: $O(\sqrt{m} \cdot \log^* m)$

Distortion=1

Part III

Improving Distortion via Queries

Approach

Before we begin

- Every result holds without making any normalization assumption about the values of the agents
  - Unless stated otherwise

Before we begin

- Every result holds without making any normalization assumption about the values of the agents
  - Unless stated otherwise
- The focus will be on: Deterministic mechanisms

Before we begin

- Every result holds without making any normalization assumption about the values of the agents
  - Unless stated otherwise
- The focus will be on: Deterministic mechanisms
  - $O(\sqrt{m})$ distortion
  - $O(1)$ distortion

Before we begin

- Every result holds without making any normalization assumption about the values of the agents
  - Unless stated otherwise
- The focus will be on: Deterministic mechanisms
  - $O(\sqrt{m})$ distortion: Bound of the randomized ordinal mechanisms
  - $O(1)$ distortion: Provides a very good approximation of the optimal outcome

Before we begin

- Every result holds without making any normalization assumption about the values of the agents
  - Unless stated otherwise
- The focus will be on: Deterministic mechanisms
  - $O(\sqrt{m})$ distortion: Bound of the randomized ordinal mechanisms
  - $O(1)$ distortion: Provides a very good approximation of the optimal outcome
- Goal: Reach these bounds with as few queries (per agent) as possible

A Warm-Up

- If we have λ available queries per agent, what is the best way to spend them?

A Warm-Up

- If we have λ available queries per agent, what is the best way to spend them?
- A first idea: There is a lot of value hidden under the λ-best alternatives of each agent
  - Since we have the ordering, we know who they are
  - Maybe we should focus there

A Warm-Up

- Mechanism: λ-Prefix Range Voting (λ-PRV)

A Warm-Up

- λ-PRV
  - Ask every agent for the value that he has at the best λ positions

A Warm-Up

- λ-PRV
  - Ask every agent for the value that he has at the best λ positions
  - Set the rest of the values to 0

A Warm-Up

- λ-PRV
  - Ask every agent for the value that he has at the best λ positions
  - Set the rest of the values to 0
  - Choose the alternative that maximizes the social welfare, according to these values

Performance?

- λ-PRV
  - By asking λ queries per agent achieves an m/λ distortion

Performance?

- λ-PRV
  - By asking λ queries per agent achieves an m/λ distortion

- Achieves distortion $O(\sqrt{m})$ using $\Theta(\sqrt{m})$ queries per agent
- Achieves distortion $O(1)$ using $\Theta(m)$ queries per agent

Can we do better?

- Is it possible to achieve these distortion bounds by asking each agent fewer queries?

Can we do better?

- Is it possible to achieve these distortion bounds by asking each agent fewer queries?
  - Yes!

Can we do better?

- Is it possible to achieve these distortion bounds by asking each agent fewer queries?
  - Yes!
- We will try to use the fact that the ordinal preferences are known in a more clever way

Can we do better?

- Is it possible to achieve these distortion bounds by asking each agent fewer queries?
  - Yes!
- We will try to use the fact that the ordinal preferences are known in a more clever way
  - What about Binary Search?

Binary Search

- Consider a set of m items the value of which is hidden

Binary Search

- Consider a set of m items the value of which is hidden
- Suppose however that the items are sorted in an increasing manner, and their ordering is given

Binary Search

- Consider a set of m items the value of which is hidden
- Suppose however that the items are sorted in an increasing manner, and their ordering is given
- Input: A number and the ordering of the items

Binary Search

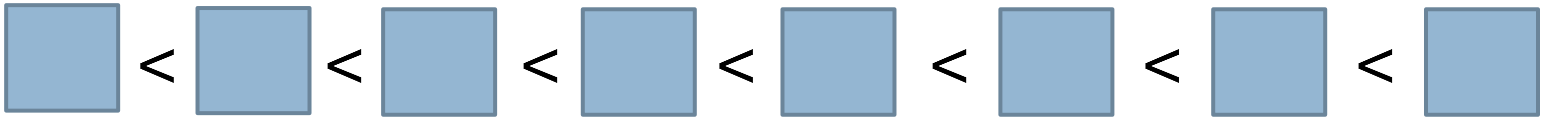
- Consider a set of m items the value of which is hidden
- Suppose however that the items are sorted in an increasing manner, and their ordering is given
- Input: A number and the ordering of the items
- Output: The item with the closest value to the given number

Binary Search

- Consider a set of m items the value of which is hidden
- Suppose however that the items are sorted in an increasing manner, and their ordering is given
- Input: A number and the ordering of the items
- Output: The item with the closest value to the given number
- Allowed actions: Ask what is the hidden value of an item

The Naive Way

- Number: 41



The Naive Way

- Number: 41

$1 < \square < \square < \square < \square < \square < \square < \square$

The Naive Way

- Number: 41

$$1 < 8 < \square < \square < \square < \square < \square < \square$$

The Naive Way

- Number: 41

1 < 8 < 19 < □ < □ < □ < □ < □

The Naive Way

- Number: 41

$1 < 8 < 19 < 37 < \square < \square < \square < \square$

The Naive Way

- Number: 41

$1 < 8 < 19 < 37 < 43 < \square < \square < \square$

The Naive Way

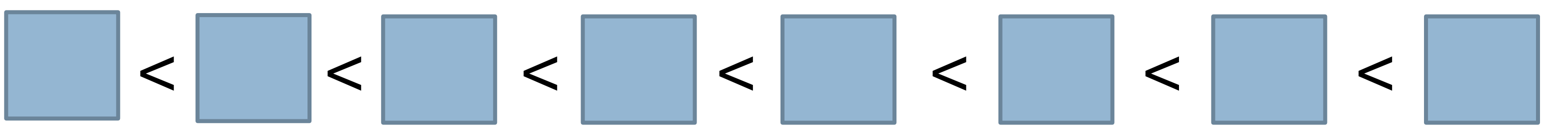
- Number: 41

1 < 8 < 19 < 37 < 43 < □ < □ < □

- We found the desired item (no need to check the rest)
  - However, in the worst-case scenario we will make *m* queries

Binary Search

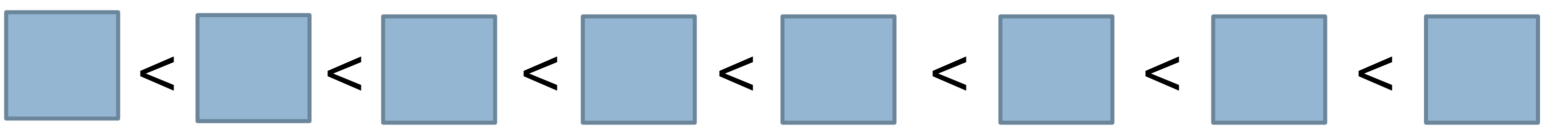
- Number: 41



- Can we solve the problem with fewer queries?

Binary Search

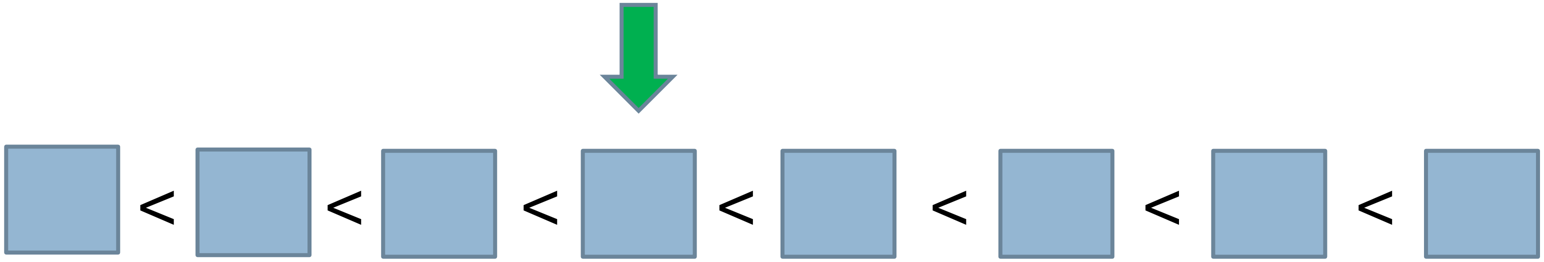
- Number: 41



- Yes! Use the ordering in a more clever way!

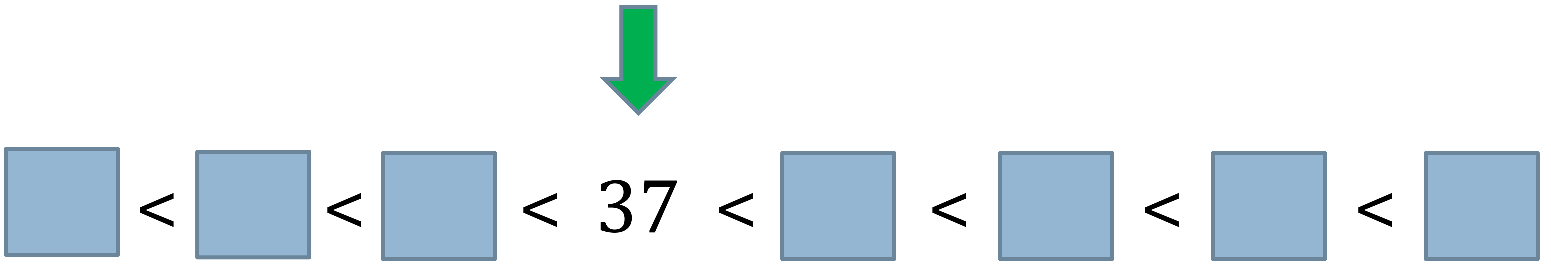
Binary Search

- Number: 41



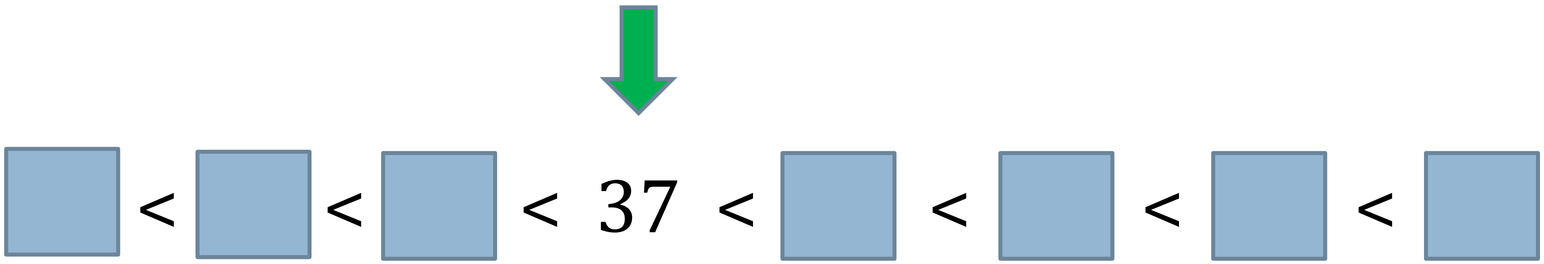
Binary Search

- Number: 41



Binary Search

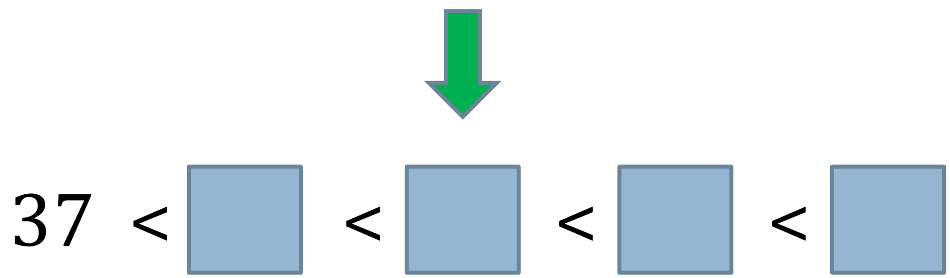
- Number: 41



- The numbers on the left are smaller than 37, so there is no need to check them

Binary Search

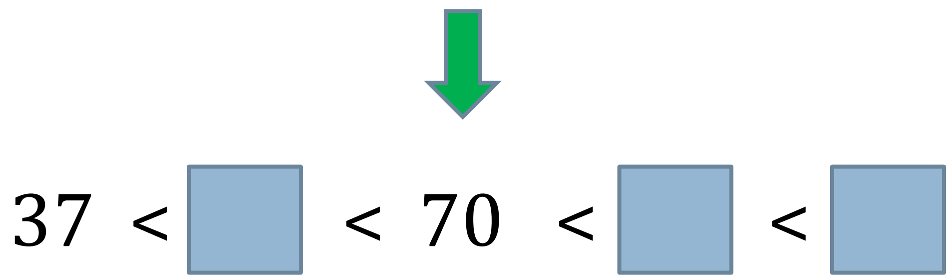
- Number: 41



- Do the same recursively

Binary Search

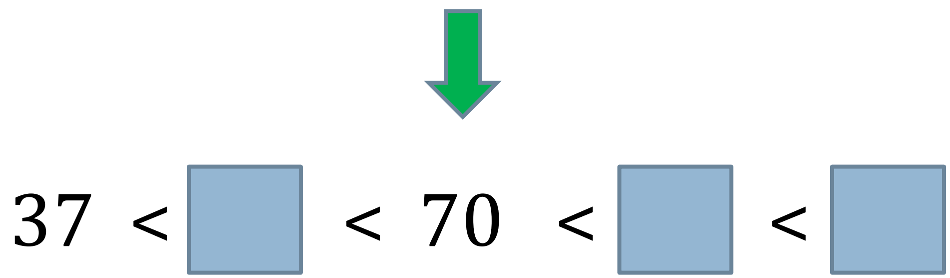
- Number: 41



- Do the same recursively

Binary Search

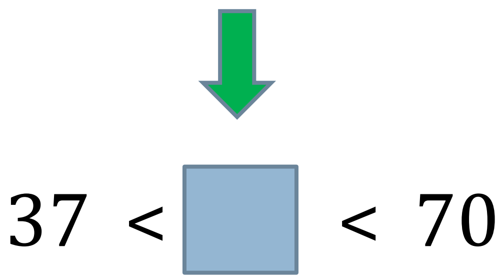
- Number: 41



- The numbers on the right are larger than 70, so there is no need to check them

Binary Search

- Number: 41



- Do the same recursively

Binary Search

- Number: 41

$$37 < 43 < 70$$

- This procedure makes at most $log m$ queries!

Can we do better?

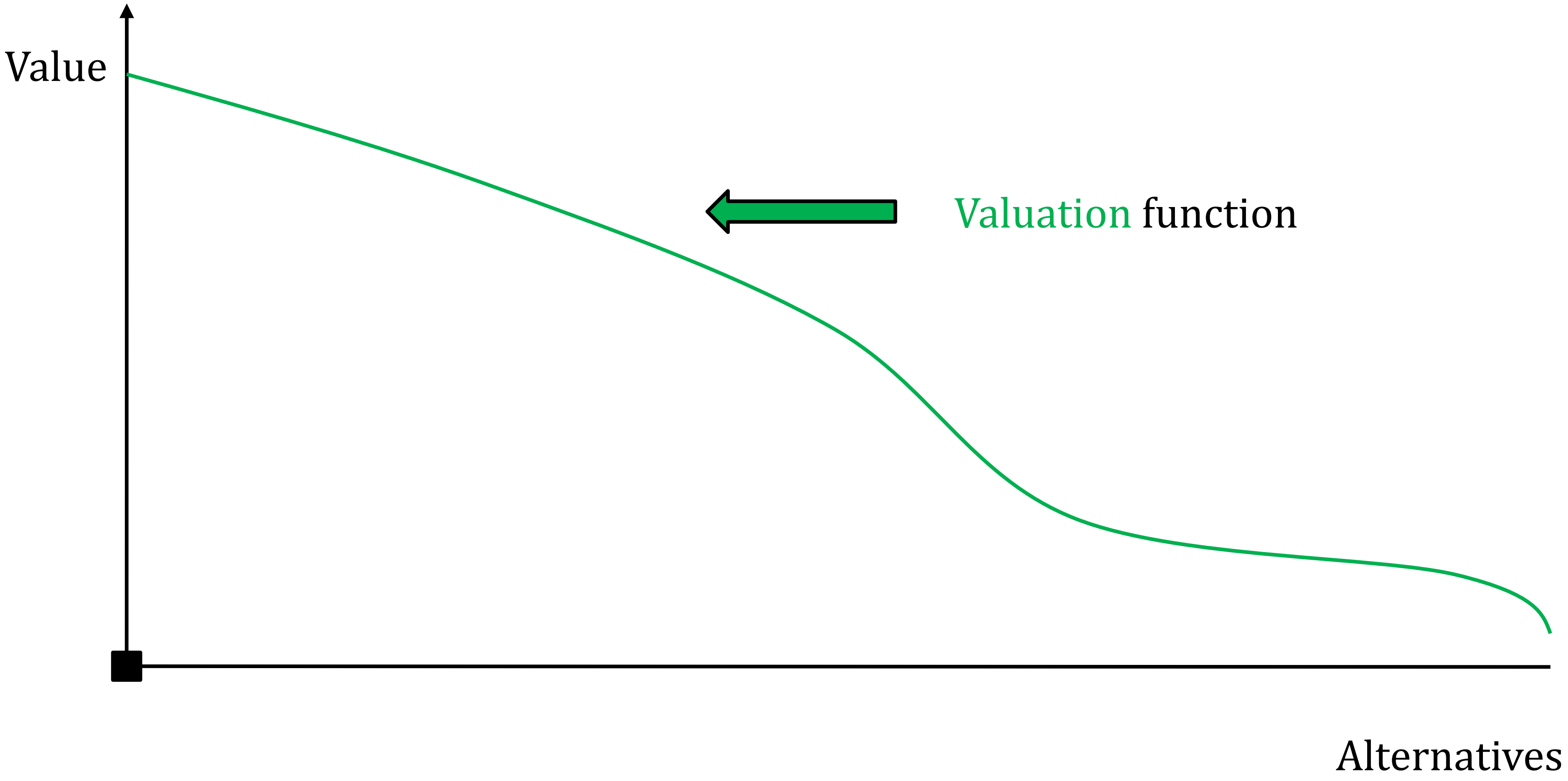
- Is it possible to achieve these distortion bounds by asking each agent fewer queries?
  - Yes!
  - k-Acceptance Range Voting (k-ARV): A mechanism that runs the Binary Search as a sub-routine

k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \ldots, \lambda_k$

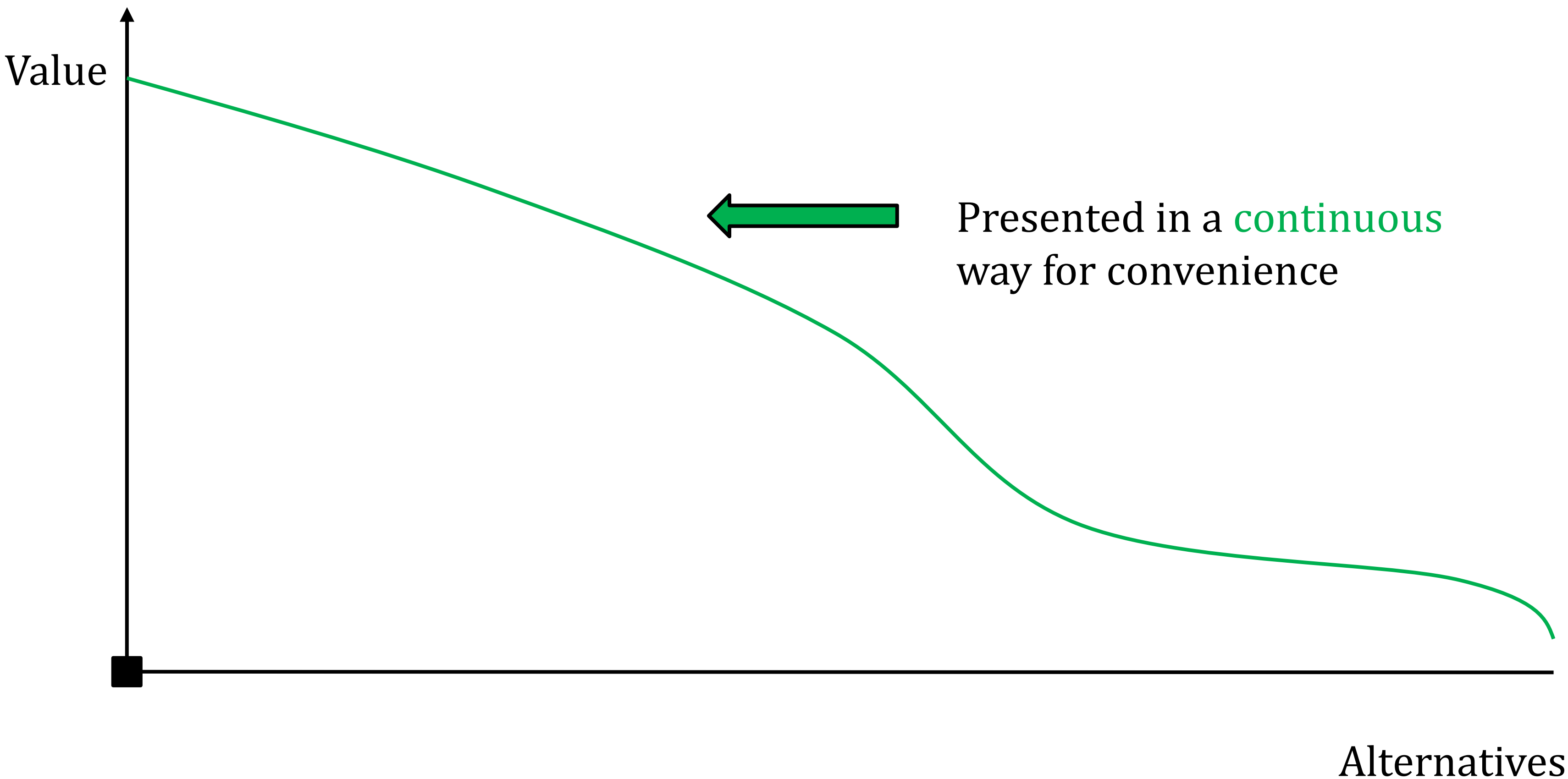
k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \ldots, \lambda_k$



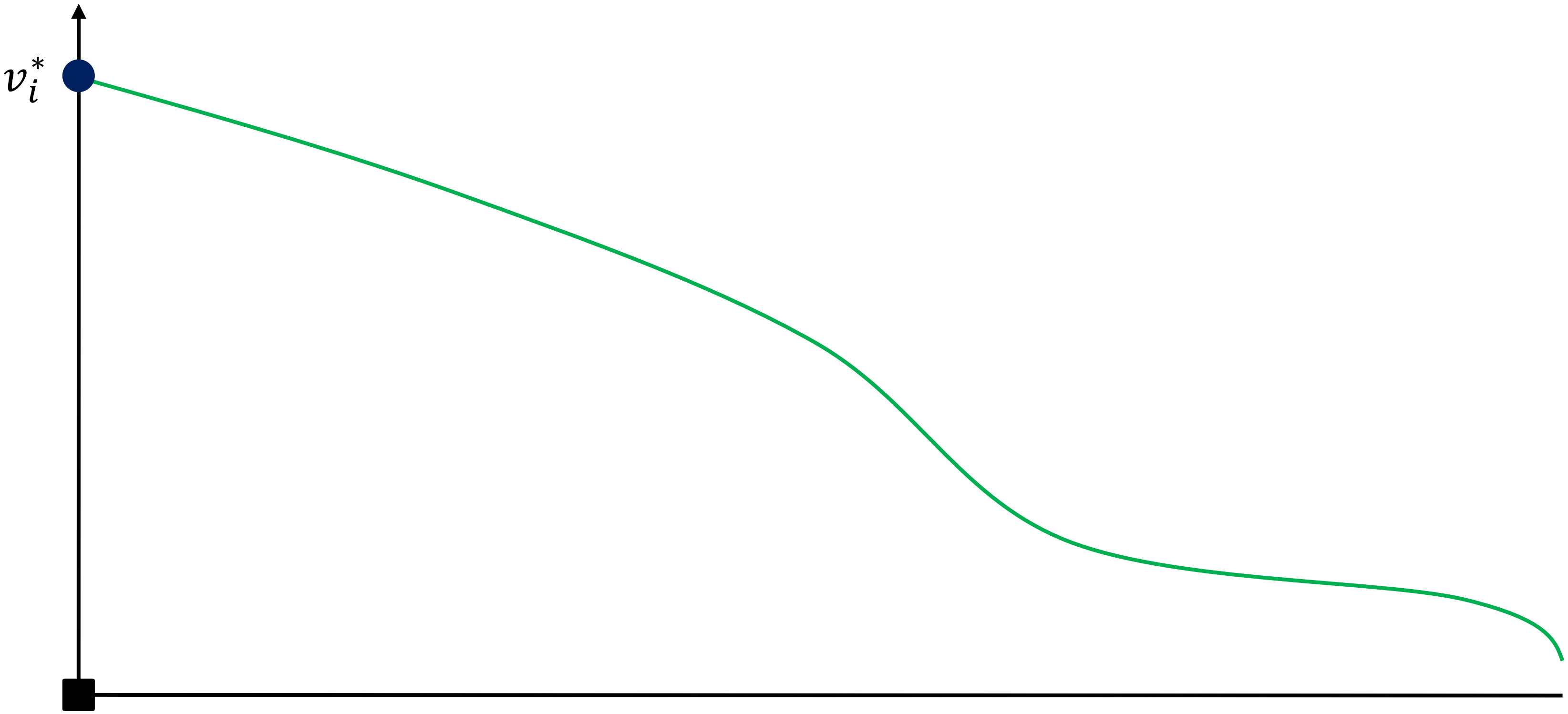
k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \dots, \lambda_k$



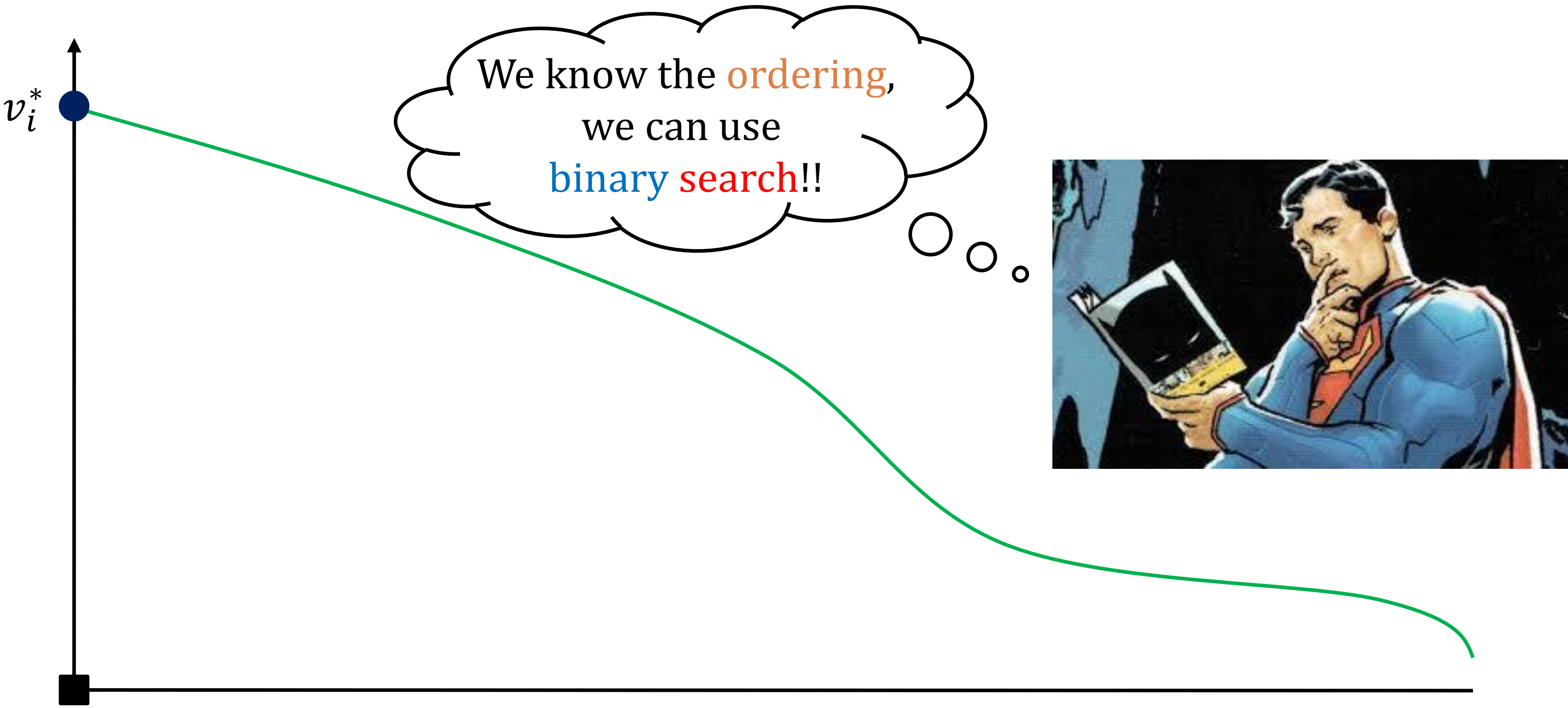
k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \ldots, \lambda_k$



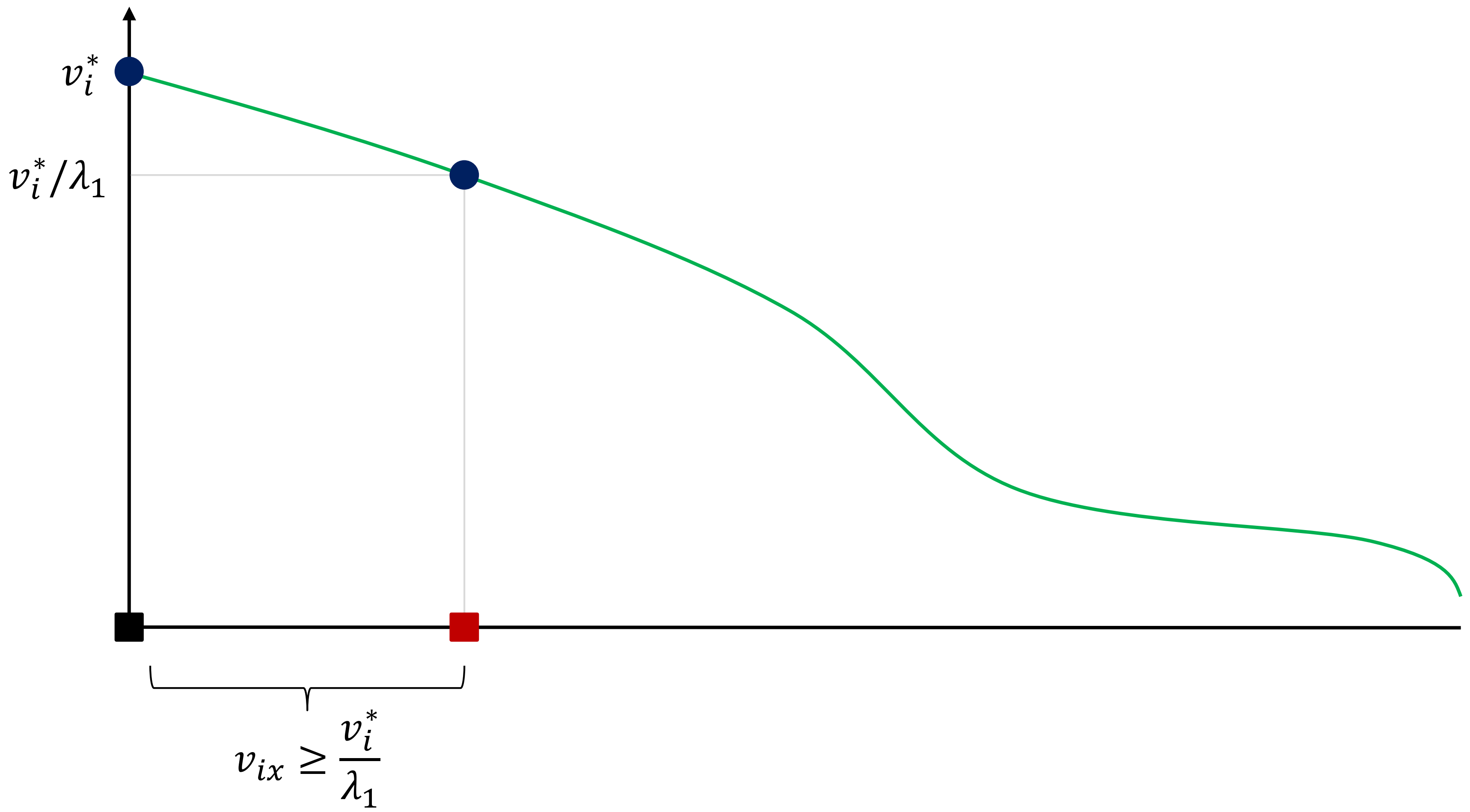
k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \ldots, \lambda_k$



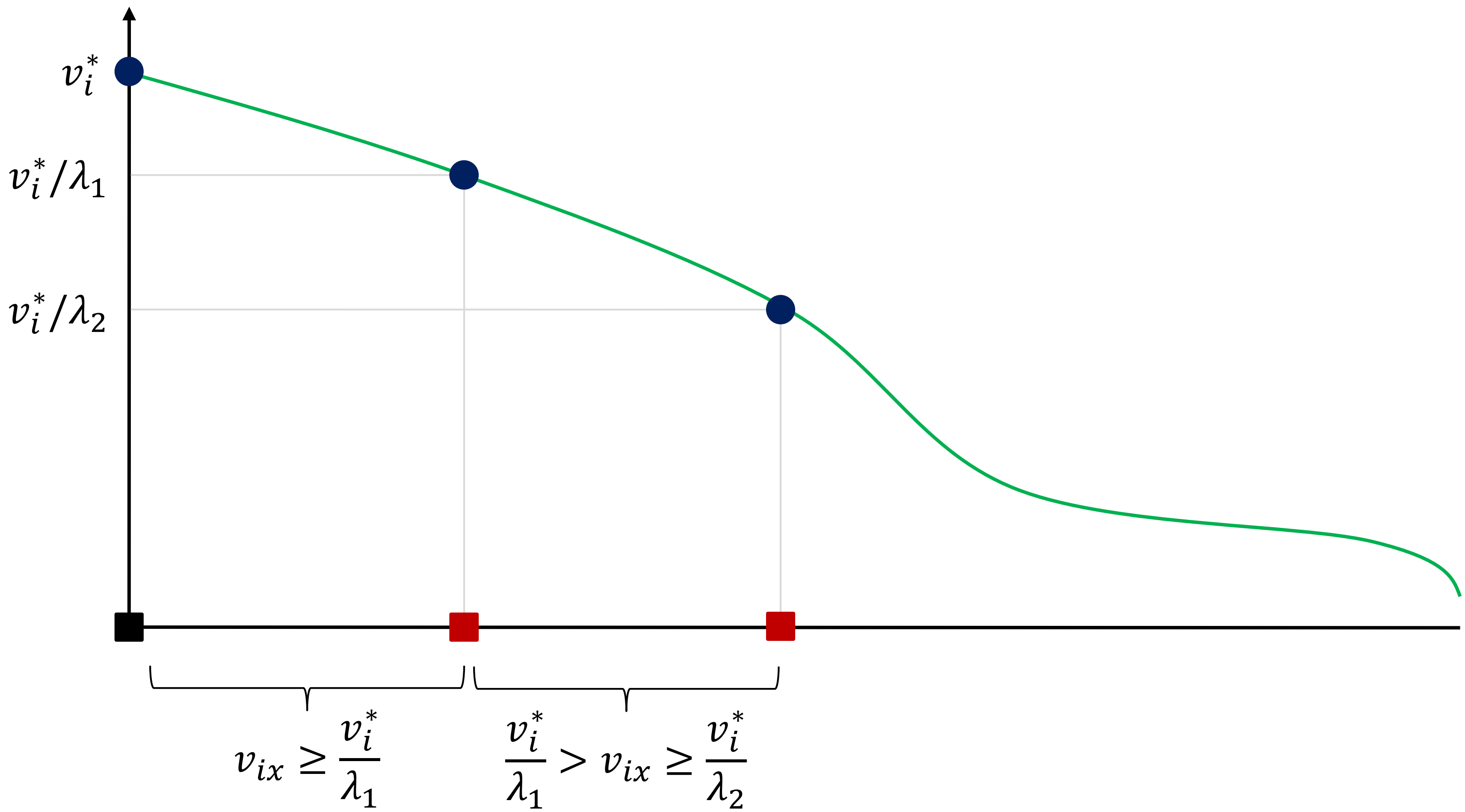
k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \dots, \lambda_k$



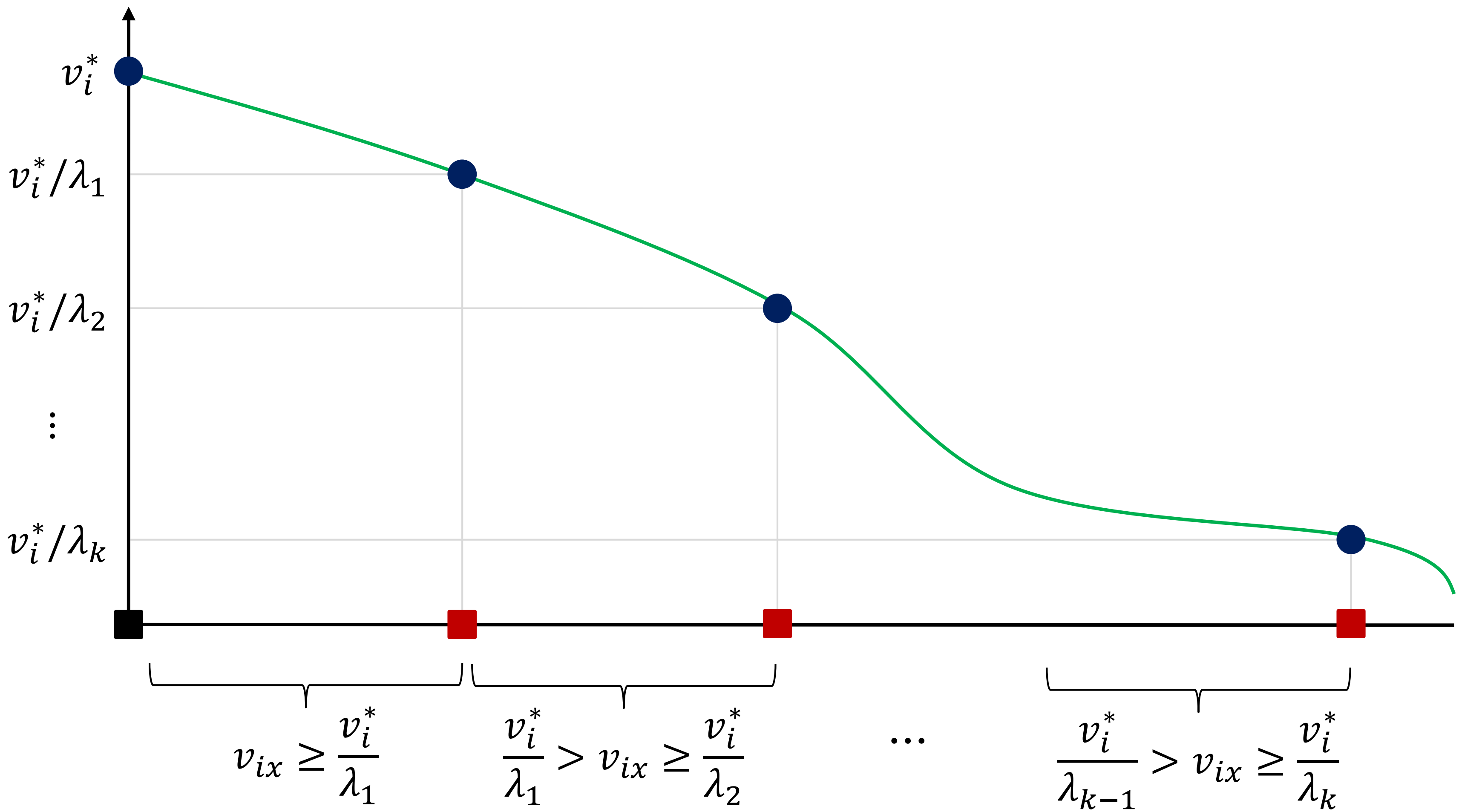
k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \ldots, \lambda_k$



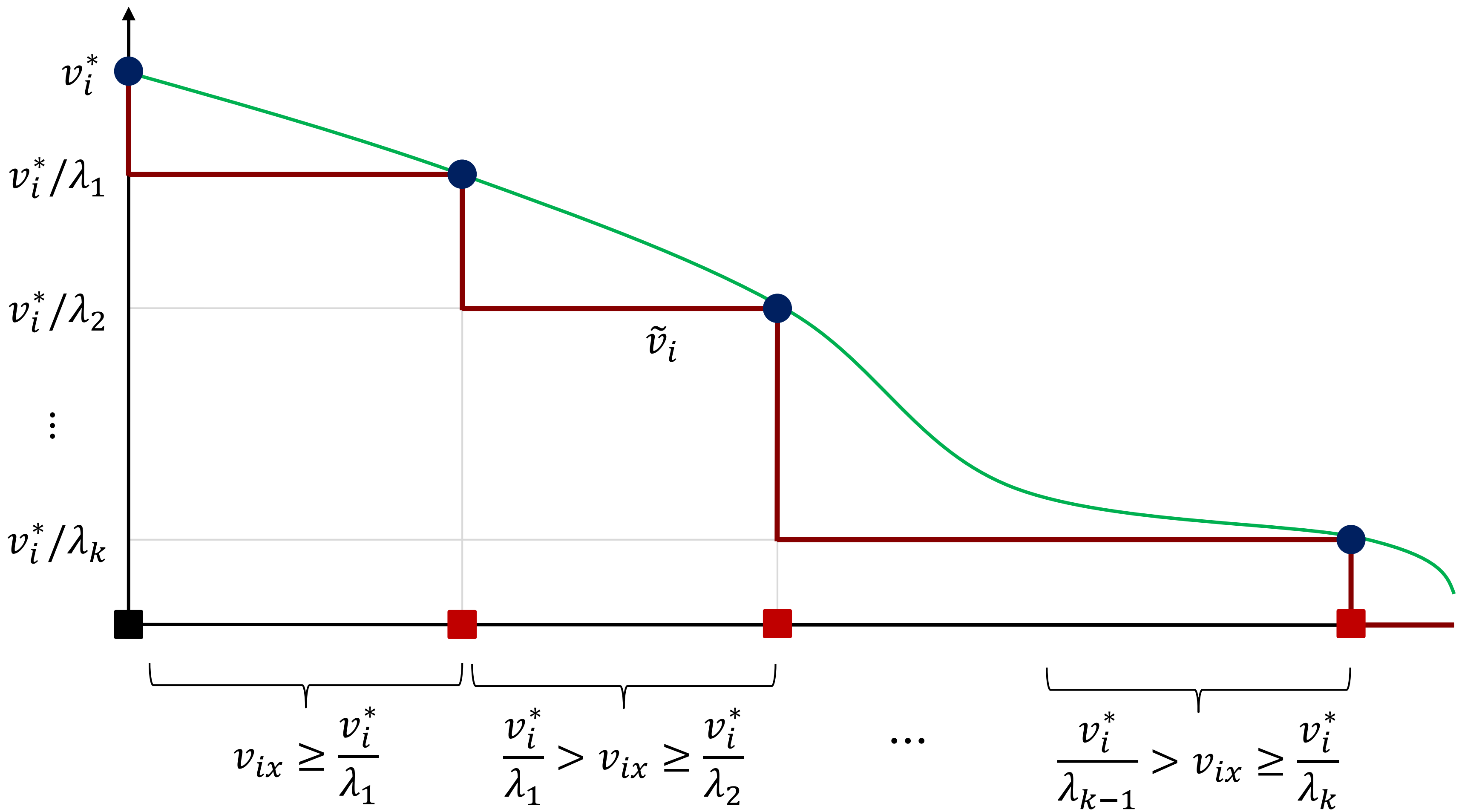
k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \dots, \lambda_k$



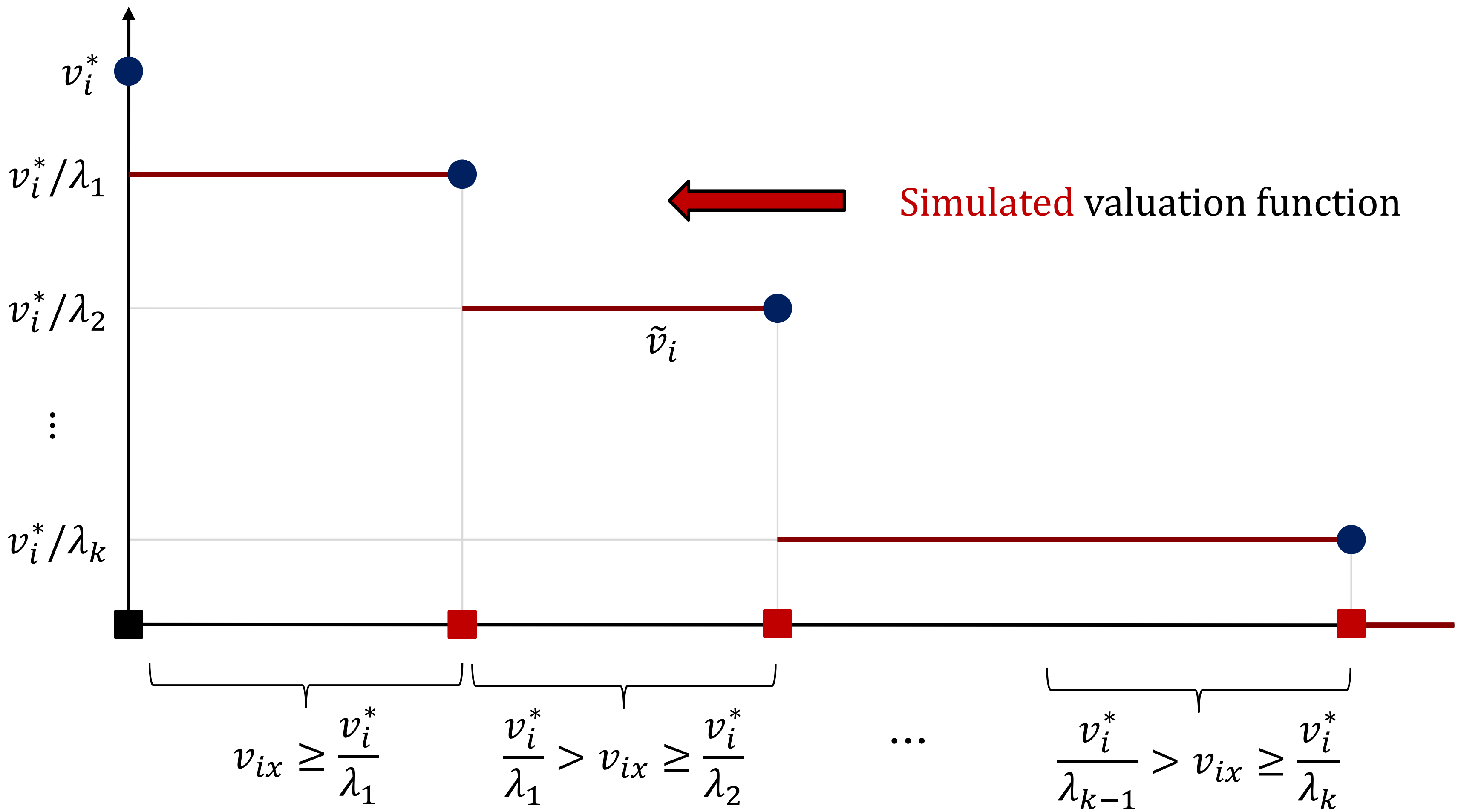
k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \dots, \lambda_k$



k-Acceptable Range Voting

- Define k threshold values $\lambda_1, \dots, \lambda_k$



k-Acceptable Range Voting

- Set $\lambda_\ell = m^{\ell/(k+1)}$ for $\ell \in [k]$
- Compute the simulated valuation function for every agent
- Return the alternative with maximum simulated social welfare

k-Acceptable Range Voting

- Set $\lambda_\ell = m^{\ell/(k+1)}$ for $\ell \in [k]$
- Compute the simulated valuation function for every agent
- Return the alternative with maximum simulated social welfare

Theorem
k-ARV makes $O(k \cdot \log m)$ values queries per agent, and has distortion $O(\sqrt[k+1]{m})$, even for unrestricted values

k-Acceptable Range Voting

- Set $\lambda_\ell = m^{\ell/(k+1)}$ for $\ell \in [k]$
- Compute the simulated valuation function for every agent
- Return the alternative with maximum simulated social welfare

Theorem
k-ARV makes $O(k \cdot \log m)$ values queries per agent, and has distortion $O(\sqrt[k+1]{m})$, even for unrestricted values

- 1-ARV has distortion $O(\sqrt{m})$ using $O(\log m)$ queries per agent
- $\log m$-ARV has distortion $O(1)$ using $O(\log^2 m)$ queries per agent

Remark 1

- $O(\sqrt{m})$ distortion
  - $\Theta(\sqrt{m})$ queries ➡ $O(\log m)$ queries
- $O(1)$ distortion
  - $\Theta(m)$ queries ➡ $O(\log^2 m)$ queries

Remark 2

- $\log m$-ARV has distortion $O(1)$ using $O(\log^2 m)$ queries per agent
- Can be also achieved by using comparison queries under the unit-sum assumption
  - The assumption is needed in order to approximate via comparison queries the value of the alternative at the first position

Remark 3

- $O(\sqrt{m})$ distortion
  - $O(\log m)$ queries
  - Lower bound: Constant number of queries per agent

- $O(1)$ distortion
  - $O(\log^2 m)$ queries
  - Lower bound: $\log m$ queries per agent

Thank You!!!!