Optimal Streaming Approximations for all Boolean Max 2-CSPs and Max k-SAT

Chi-Ning Chou Sasha Golonev Santhoshini Velusamy

Harvard University

FOCS 2020

Motivation and Spoiler

Motivation and Spoiler

- Constraint satisfaction problem (CSP) in the streaming model.

Motivation and Spoiler

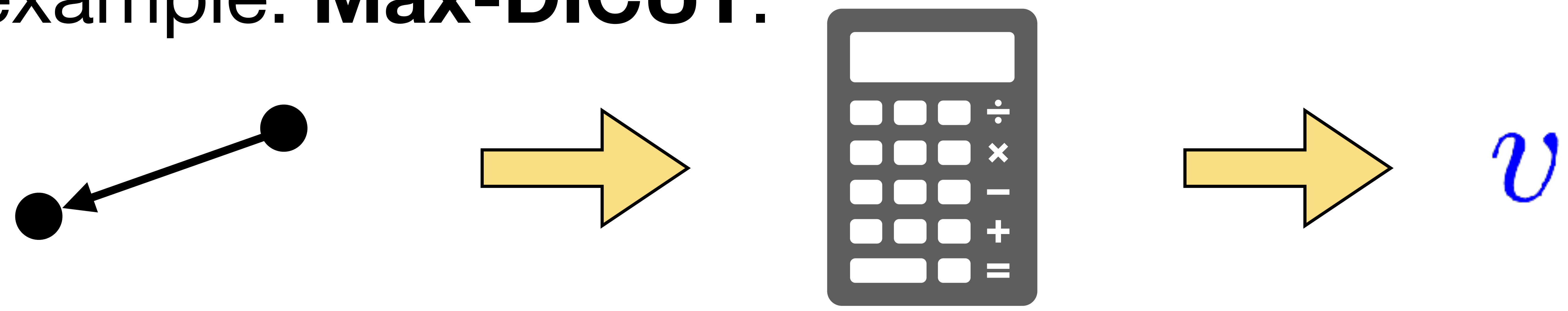
- Constraint satisfaction problem (CSP) in the streaming model.
  - CSP is one of the *central computational problems* in complexity theory.

Motivation and Spoiler

- Constraint satisfaction problem (CSP) in the streaming model.
  - ✦ CSP is one of the *central computational problems* in complexity theory.
  - ✦ Proving *unconditional hardness* in streaming model is more doable.

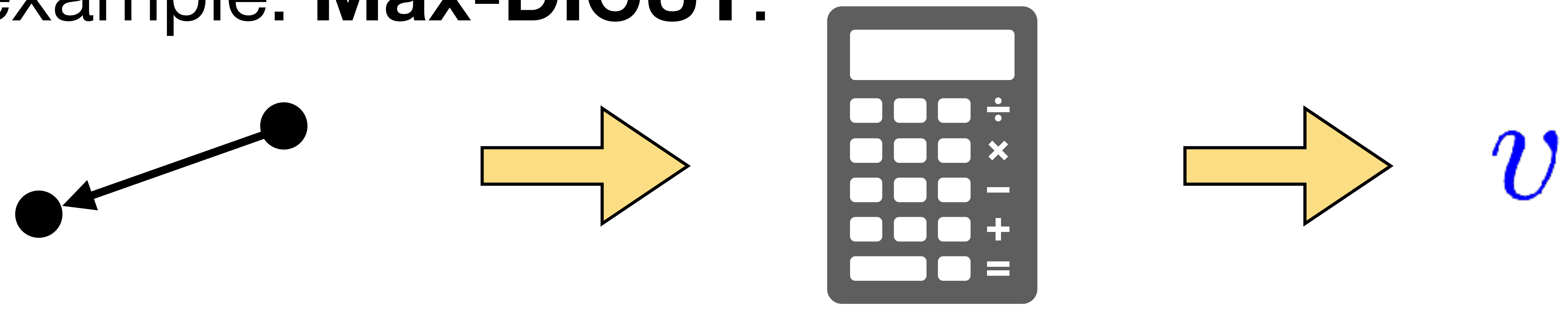
Motivation and Spoiler

- Constraint satisfaction problem (CSP) in the streaming model.
  - ✦ CSP is one of the *central computational problems* in complexity theory.
  - ✦ Proving *unconditional hardness* in streaming model is more doable.
- Motivating example: **Max-DICUT**.



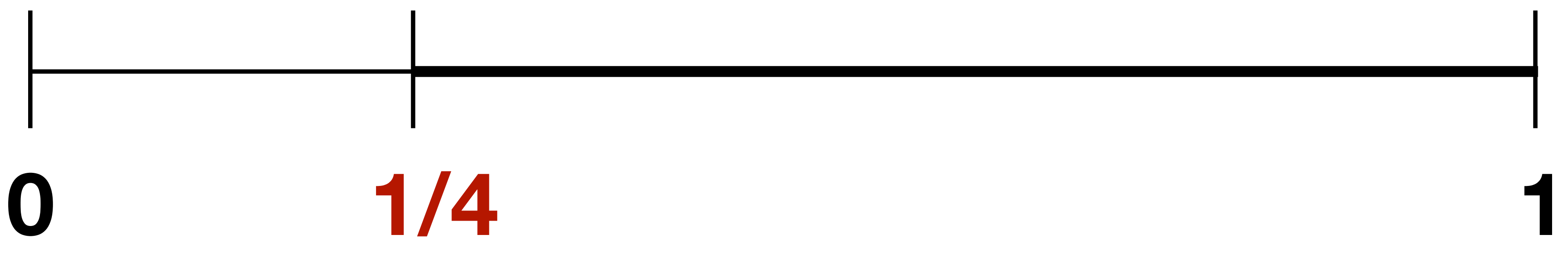
Motivation and Spoiler

- Constraint satisfaction problem (CSP) in the streaming model.
  - ✦ CSP is one of the *central computational problems* in complexity theory.
  - ✦ Proving *unconditional hardness* in streaming model is more doable.
- Motivating example: **Max-DICUT**.

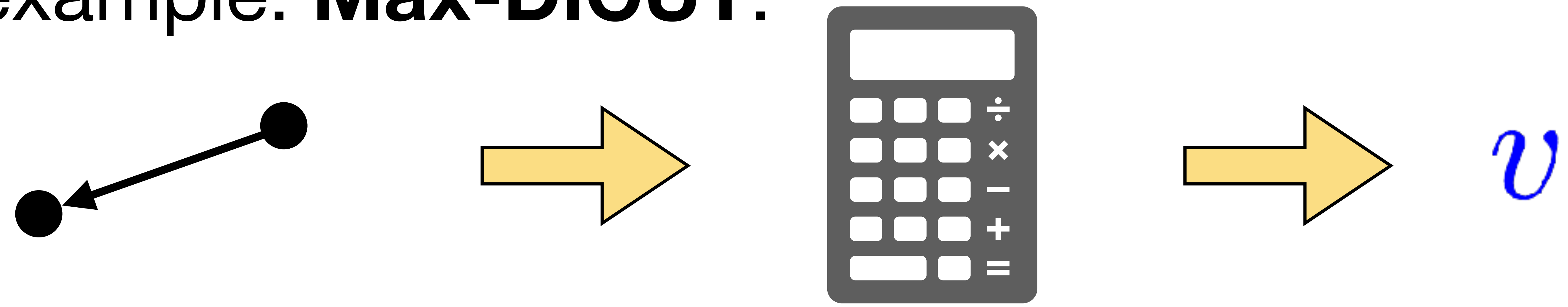


- Trivial algorithm gives **1/4**-approx.; [Guruswami-Velingker-Velusamy 17] showed **2/5**-approx.; [GVV17] + [Kapralov-Khanna-Sudan 15]: **1/2**-approx. is impossible.

Motivation and Spoiler

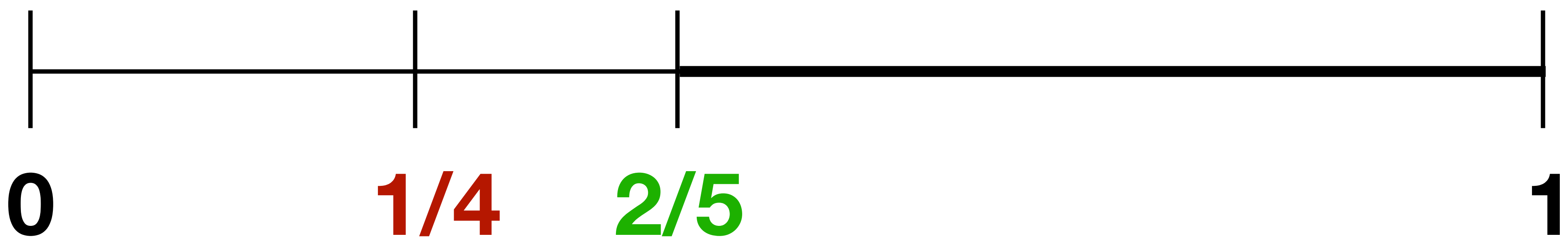


- Constraint satisfaction problem (CSP) in the streaming model.
  - ✦ CSP is one of the *central computational problems* in complexity theory.
  - ✦ Proving *unconditional hardness* in streaming model is more doable.
- Motivating example: **Max-DICUT**.

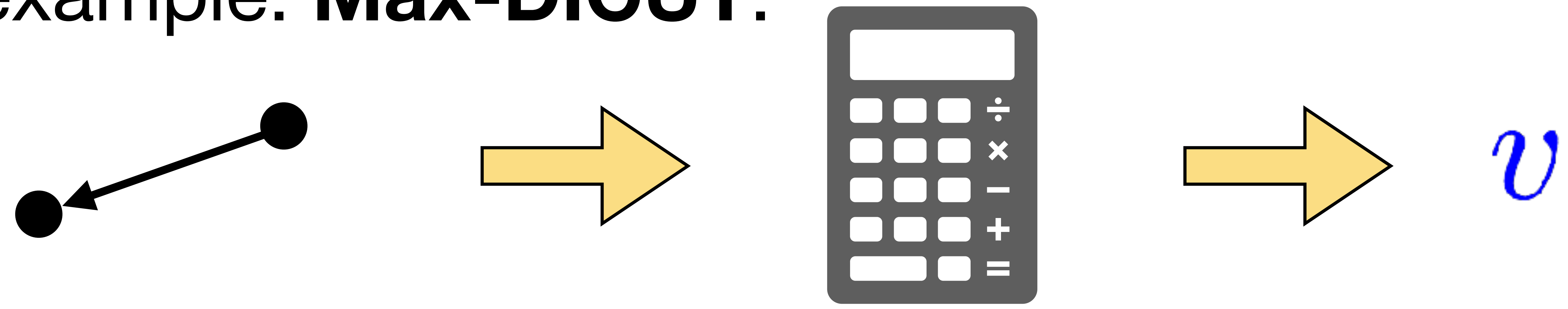


- Trivial algorithm gives **1/4**-approx.; [Guruswami-Velingker-Velusamy 17] showed **2/5**-approx.; [GVV17] + [Kapralov-Khanna-Sudan 15]: **1/2**-approx. is impossible.

Motivation and Spoiler

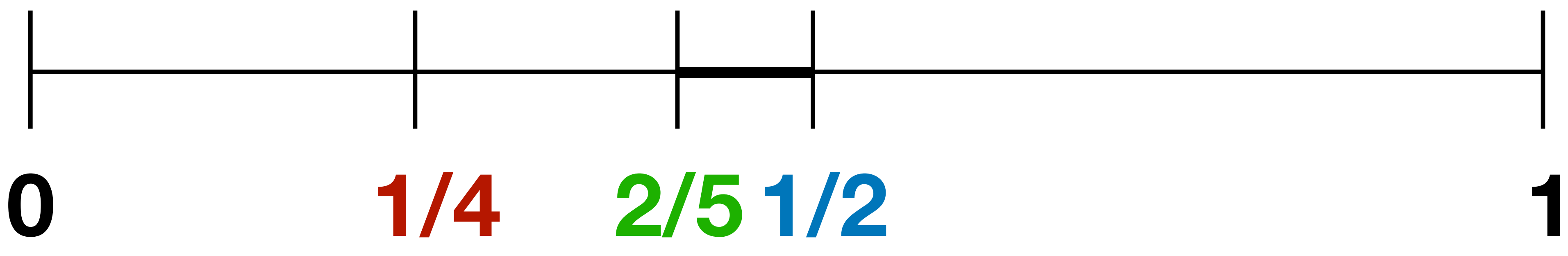


- Constraint satisfaction problem (CSP) in the streaming model.
  - ✦ CSP is one of the *central computational problems* in complexity theory.
  - ✦ Proving *unconditional hardness* in streaming model is more doable.
- Motivating example: **Max-DICUT**.

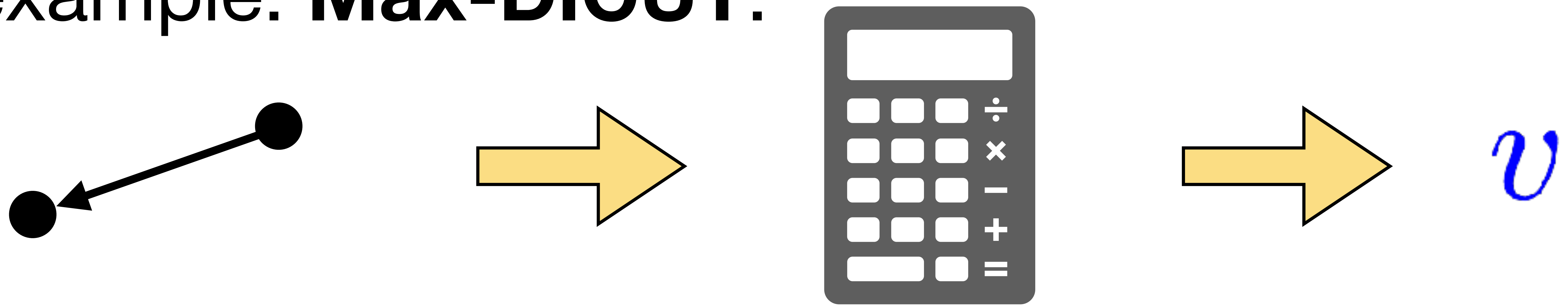


- Trivial algorithm gives **1/4**-approx.; [Guruswami-Velingker-Velusamy 17] showed **2/5**-approx.; [GVV17] + [Kapralov-Khanna-Sudan 15]: **1/2**-approx. is impossible.

Motivation and Spoiler

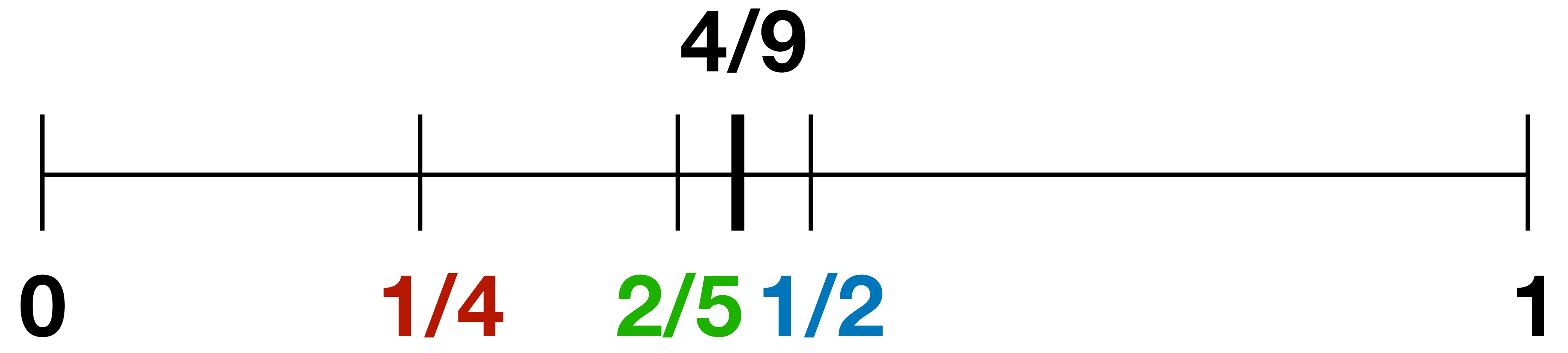


- Constraint satisfaction problem (CSP) in the streaming model.
  - ✦ CSP is one of the *central computational problems* in complexity theory.
  - ✦ Proving *unconditional hardness* in streaming model is more doable.
- Motivating example: **Max-DICUT**.

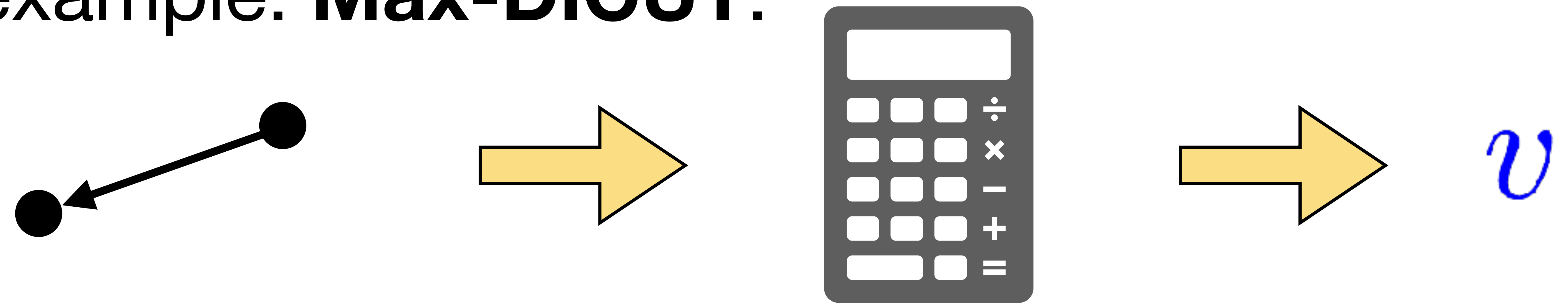


- Trivial algorithm gives **1/4**-approx.; [Guruswami-Velingker-Velusamy 17] showed **2/5**-approx.; [GVV17] + [Kapralov-Khanna-Sudan 15]: **1/2**-approx. is impossible.

Motivation and Spoiler

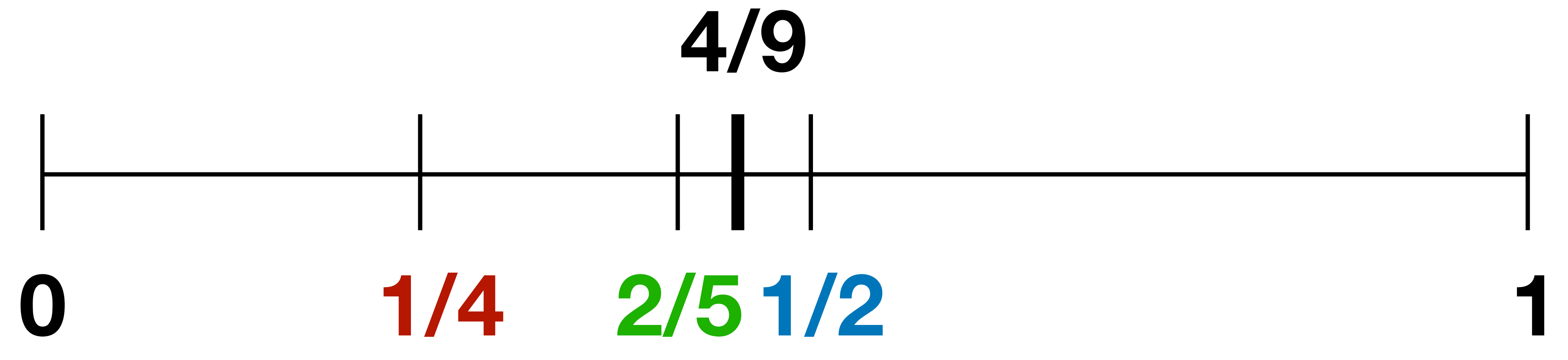


- Constraint satisfaction problem (CSP) in the streaming model.
  - ✦ CSP is one of the *central computational problems* in complexity theory.
  - ✦ Proving *unconditional hardness* in streaming model is more doable.
- Motivating example: **Max-DICUT**.

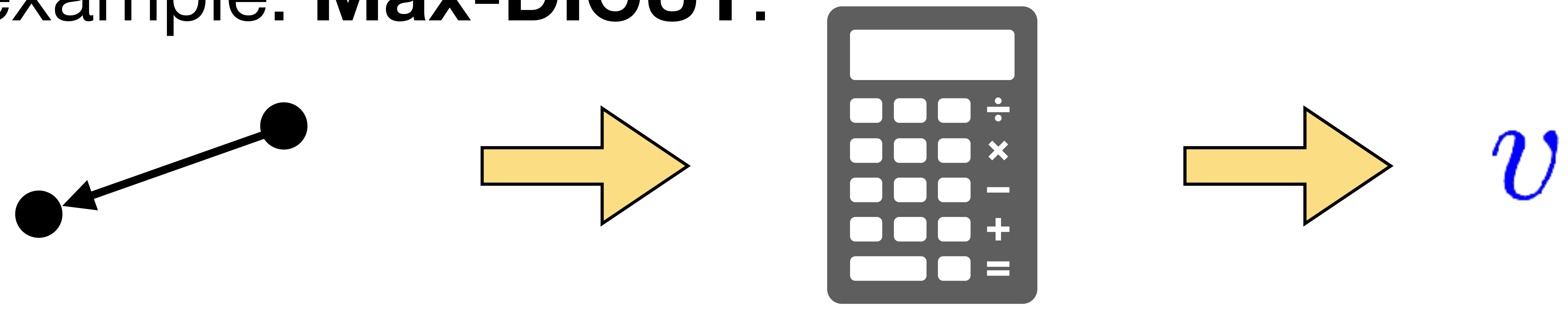


- Trivial algorithm gives **1/4**-approx.; [Guruswami-Velingker-Velusamy 17] showed **2/5**-approx.; [GVV17] + [Kapralov-Khanna-Sudan 15]: **1/2**-approx. is impossible.
- We show that **4/9**-approximation is the right answer!

Motivation and Spoiler



- Constraint satisfaction problem (CSP) in the streaming model.
    - ✦ CSP is one of the *central computational problems* in complexity theory.
    - ✦ Proving *unconditional hardness* in streaming model is more doable.
- Motivating example: **Max-DICUT**.



- Trivial algorithm gives **1/4**-approx.; [Guruswami-Velingker-Velusamy 17] showed **2/5**-approx.; [GVV17] + [Kapralov-Khanna-Sudan 15]: **1/2**-approx. is impossible.
- We show that **4/9**-approximation is the right answer!
- Further, we characterize the approximation ratio of every boolean 2-CSP!

Definitions

Constraint Satisfaction Problem (CSP)

Constraint Satisfaction Problem (CSP)

- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.

Constraint Satisfaction Problem (CSP)

- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0,1\}$ and $S \subset [n]$.

Constraint Satisfaction Problem (CSP)

- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0, 1\}$ and $S \subset [n]$.

  Example: $f(\cdot, \cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.

Constraint Satisfaction Problem (CSP)

- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0, 1\}$ and $S \subset [n]$.

  Example: $f(\cdot, \cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.
- **Input**: $\mathcal{C} = \{(f, S)\}$, number of constraints = m.

Constraint Satisfaction Problem (CSP)

- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0,1\}$ and $S \subset [n]$.

  Example: $f(\cdot,\cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.
- **Input**: $\mathcal{C} = \{(f, S)\}$, number of constraints = m.
- **Output**: The value of $\mathcal{C}$. Namely, the largest # of satisfied constraints.

Constraint Satisfaction Problem (CSP)

- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0,1\}$ and $S \subset [n]$.

  Example: $f(\cdot,\cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.
- **Input**: $\mathcal{C} = \{(f, S)\}$, number of constraints = m.
- **Output**: The value of $\mathcal{C}$. Namely, the largest # of satisfied constraints.

  Formally, define $\mathsf{val}_{\mathcal{C}} = \max_{\sigma:[n]\to\Sigma} |\{(f,S) \in \mathcal{C} : f(\sigma(x_S)) = 1\}| \in [0, m]$.

Constraint Satisfaction Problem (CSP)

- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0,1\}$ and $S \subset [n]$.

  Example: $f(\cdot,\cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.
- **Input**: $\mathcal{C} = \{(f, S)\}$, number of constraints = m.
- **Output**: The value of $\mathcal{C}$. Namely, the largest # of satisfied constraints.

  Formally, define $\mathsf{val}_{\mathcal{C}} = \max_{\sigma:[n]\to\Sigma} |\{(f,S) \in \mathcal{C} : f(\sigma(x_S)) = 1\}| \in [0, m]$.

  Assignment

  Restriction of the variables

Constraint Satisfaction Problem (CSP)

- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0,1\}$ and $S \subset [n]$.

  Example: $f(\cdot,\cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.
- **Input**: $\mathcal{C} = \{(f, S)\}$, number of constraints = m.
- **Output**: The value of $\mathcal{C}$. Namely, the largest # of satisfied constraints.

  Formally, define $\mathsf{val}_{\mathcal{C}} = \max_{\sigma:[n]\to\Sigma} |\{(f,S) \in \mathcal{C} : \ f(\sigma(x_S)) = 1\}| \in [0, m]$.

Max-CUT as a CSP

Constraint Satisfaction Problem (CSP)

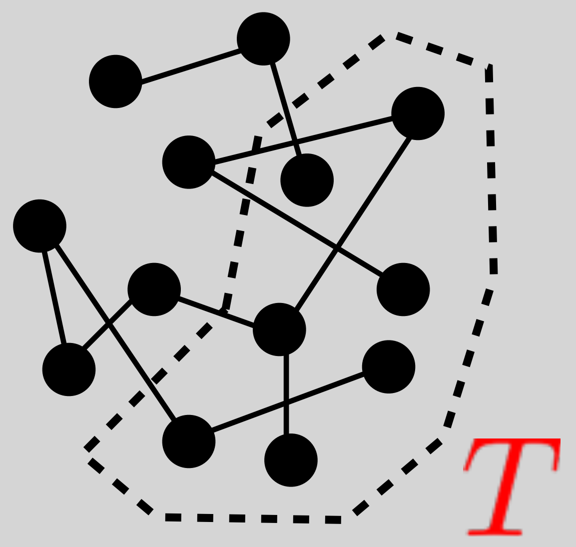
- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0,1\}$ and $S \subset [n]$.

  Example: $f(\cdot,\cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.
- **Input**: $\mathcal{C} = \{(f, S)\}$, number of constraints = m.
- **Output**: The value of $\mathcal{C}$. Namely, the largest # of satisfied constraints.

  Formally, define $\mathsf{val}_{\mathcal{C}} = \max_{\sigma:[n]\to\Sigma} |\{(f,S) \in \mathcal{C} : \ f(\sigma(x_S)) = 1\}| \in [0, m]$.

Max-CUT as a CSP

An undirected graph G.



Constraint Satisfaction Problem (CSP)

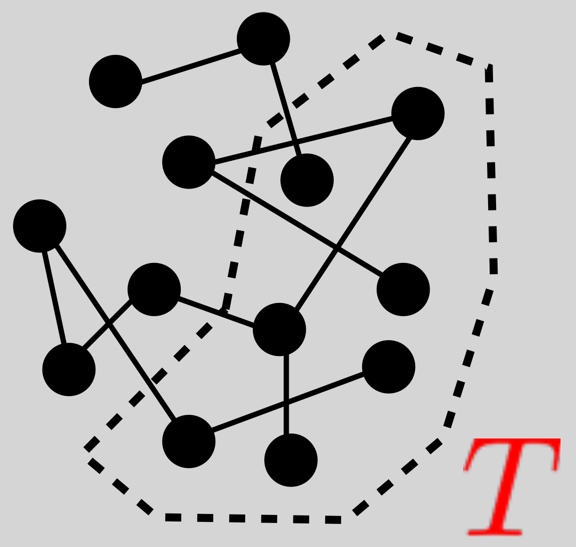
- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0,1\}$ and $S \subset [n]$.

  Example: $f(\cdot,\cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.
- **Input**: $\mathcal{C} = \{(f, S)\}$, number of constraints = m.
- **Output**: The value of $\mathcal{C}$. Namely, the largest # of satisfied constraints.

  Formally, define $\mathsf{val}_{\mathcal{C}} = \max_{\sigma:[n]\to\Sigma} |\{(f,S) \in \mathcal{C} : f(\sigma(x_S)) = 1\}| \in [0, m]$.

<u>Max-CUT as a CSP</u>

An undirected graph G.



- **Variables**: $x_i = 1 \Leftrightarrow i \in T$

Constraint Satisfaction Problem (CSP)

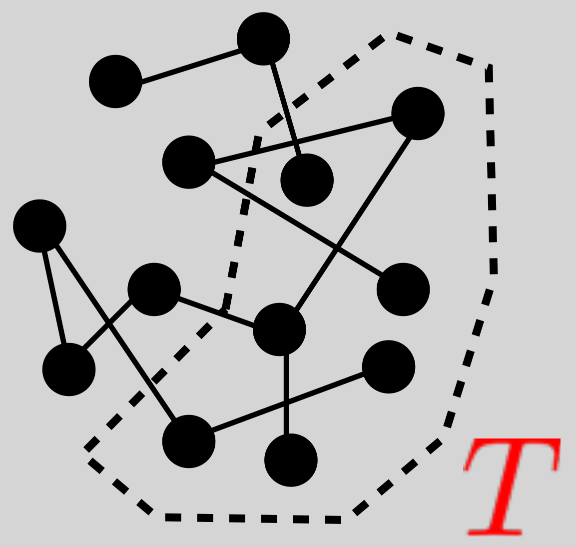
- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0,1\}$ and $S \subset [n]$.

  Example: $f(\cdot,\cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.
- **Input**: $\mathcal{C} = \{(f, S)\}$, number of constraints = m.
- **Output**: The value of $\mathcal{C}$. Namely, the largest # of satisfied constraints.

  Formally, define $\mathsf{val}_{\mathcal{C}} = \max_{\sigma:[n]\to\Sigma} |\{(f,S) \in \mathcal{C} : \ f(\sigma(x_S)) = 1\}| \in [0, m]$.

Max-CUT as a CSP

An undirected graph G.



- **Variables**: $x_i = 1 \Leftrightarrow i \in T$
- **Constraints**: $(i, j) \in E \Leftrightarrow x_i \oplus x_j \in \mathcal{C}$

Constraint Satisfaction Problem (CSP)

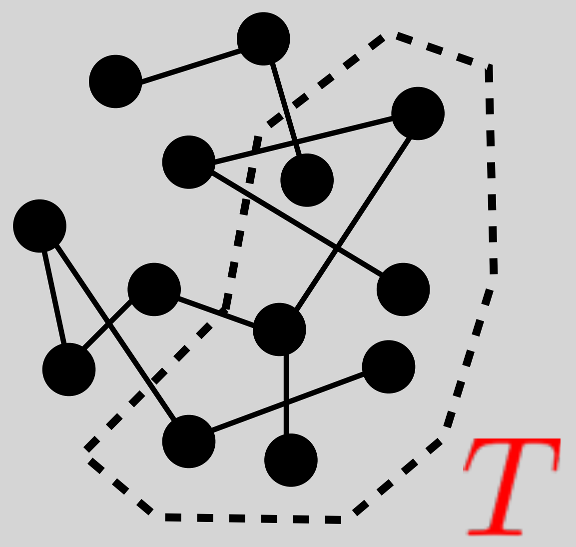
- **Variables**: $x_1, x_2, \ldots, x_n$ taking values in Σ.
- **Constraints**: (f, S) where $f : \Sigma^k \to \{0,1\}$ and $S \subset [n]$.

  Example: $f(\cdot,\cdot) = \cdot \wedge \cdot$ and $S = \{3, 8\}$, read as $x_3 \wedge x_8$.
- **Input**: $\mathcal{C} = \{(f, S)\}$, number of constraints = m.
- **Output**: The value of $\mathcal{C}$. Namely, the largest # of satisfied constraints.

  Formally, define $\mathsf{val}_{\mathcal{C}} = \max_{\sigma:[n]\to\Sigma} |\{(f,S) \in \mathcal{C} : f(\sigma(x_S)) = 1\}| \in [0, m]$.

Max-CUT as a CSP

An undirected graph G.



- **Variables**: $x_i = 1 \Leftrightarrow i \in T$
- **Constraints**: $(i,j) \in E \Leftrightarrow x_i \oplus x_j \in \mathcal{C}$
- **Value**: $\mathsf{val}_{\mathcal{C}} =$ max cut value

Constraint Satisfaction Problem (CSP)

Constraint Satisfaction Problem (CSP)

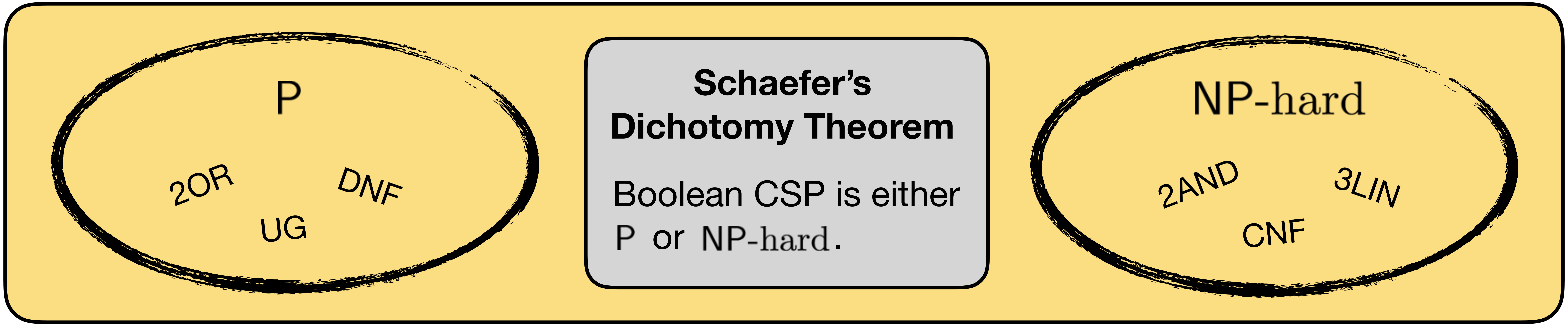
- CSP is ubiquitous and has been extremely well-studied!

Constraint Satisfaction Problem (CSP)

- CSP is ubiquitous and has been extremely well-studied!
- Some CSPs are easy and some are hard to solve **exactly**.

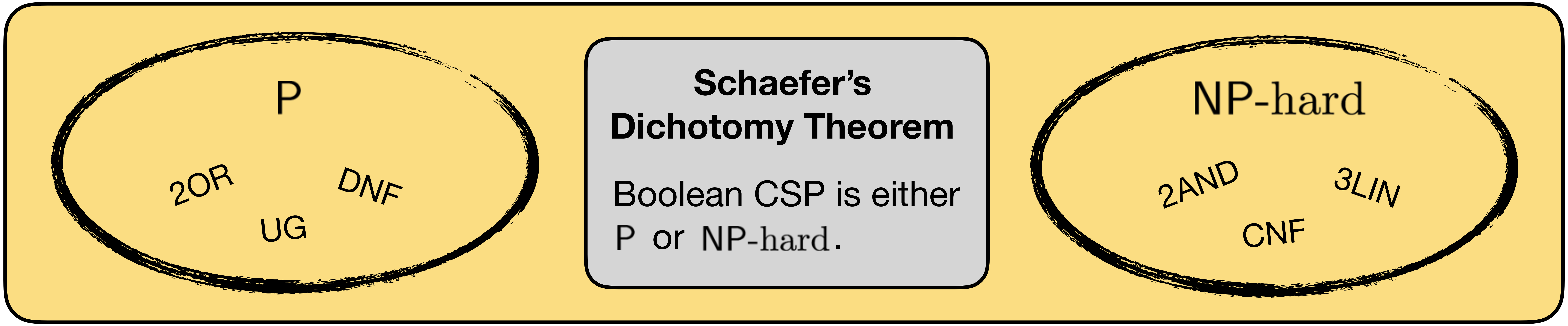
Constraint Satisfaction Problem (CSP)

- CSP is ubiquitous and has been extremely well-studied!
- Some CSPs are easy and some are hard to solve **exactly**.



Constraint Satisfaction Problem (CSP)

- CSP is ubiquitous and has been extremely well-studied!
- Some CSPs are easy and some are hard to solve **exactly**.



- What about solving CSP **approximately**?

Approximating CSP

Approximating CSP

- Approximation <=> Distinguishing instances with different values.

Approximating CSP

- Approximation <=> Distinguishing instances with different values.

α-approximation: Let $\alpha \in (0, 1]$. For any $v \in (0, m]$, can distinguish the following.

Approximating CSP

- Approximation <=> Distinguishing instances with different values.

α-approximation: Let $\alpha \in (0,1]$. For any $v \in (0,m]$, can distinguish the following.

Yes: $\mathsf{val}_{\mathcal{C}} \geq v$ **No:** $\mathsf{val}_{\mathcal{C}} < \alpha \cdot v$

Approximating CSP

- Approximation <=> Distinguishing instances with different values.

α-approximation: Let $\alpha \in (0, 1]$. For any $v \in (0, m]$, can distinguish the following.

Yes: $\mathsf{val}_{\mathcal{C}} \geq v$ **No:** $\mathsf{val}_{\mathcal{C}} < \alpha \cdot v$

- $\alpha = 1$: the exact version; $\alpha = 1 - \epsilon, \forall \epsilon > 0$: fully approximation.

Approximating CSP

- Approximation <=> Distinguishing instances with different values.

α-approximation: Let $\alpha \in (0, 1]$. For any $v \in (0, m]$, can distinguish the following.

Yes: $\mathsf{val}_{\mathcal{C}} \geq v$ **No:** $\mathsf{val}_{\mathcal{C}} < \alpha \cdot v$

- $\alpha = 1$: the exact version; $\alpha = 1 - \epsilon, \forall \epsilon > 0$: fully approximation.
- **Algorithmic side**: Random sampling, SDP-based algorithms.

Approximating CSP

- Approximation <=> Distinguishing instances with different values.

α-approximation: Let $\alpha \in (0, 1]$. For any $v \in (0, m]$, can distinguish the following.

Yes: $\text{val}_{\mathcal{C}} \geq v$ **No:** $\text{val}_{\mathcal{C}} < \alpha \cdot v$

- $\alpha = 1$: the exact version; $\alpha = 1 - \epsilon, \forall \epsilon > 0$: fully approximation.
- **Algorithmic side**: Random sampling, SDP-based algorithms.
- **Hardness side**: NP-hardness or UG-hardness (through PCP theorem).

Approximating CSP

Max-CUT: 0 1

- Approximation <=> Distinguishing instances with different values.

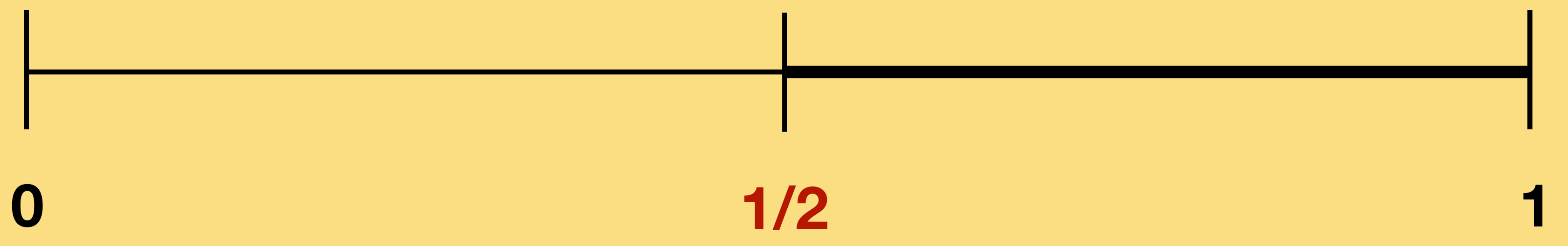
α-approximation: Let $\alpha \in (0,1]$. For any $v \in (0,m]$, can distinguish the following.

Yes: $\text{val}_{\mathcal{C}} \geq v$ **No:** $\text{val}_{\mathcal{C}} < \alpha \cdot v$

- $\alpha = 1$: the exact version; $\alpha = 1-\epsilon, \forall \epsilon > 0$: fully approximation.
- **Algorithmic side**: Random sampling, SDP-based algorithms.
- **Hardness side**: NP-hardness or UG-hardness (through PCP theorem).

Approximating CSP

Max-CUT:



- Approximation <=> Distinguishing instances with different values.

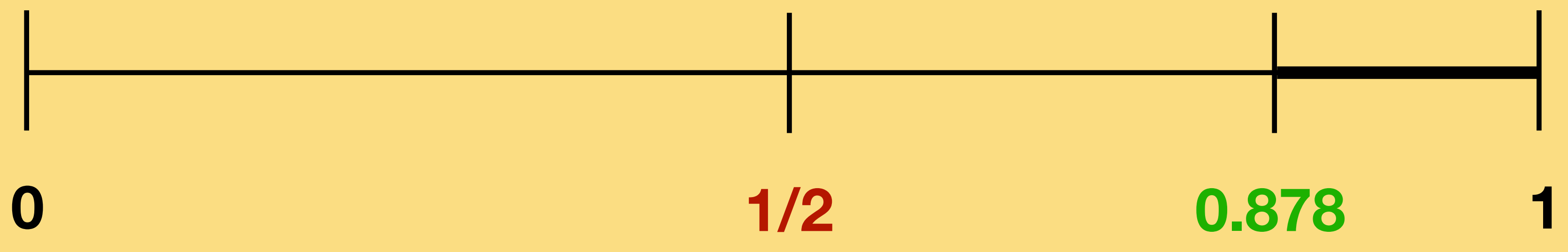
α-approximation: Let $\alpha \in (0, 1]$. For any $v \in (0, m]$, can distinguish the following.

Yes: $\text{val}_{\mathcal{C}} \geq v$ **No:** $\text{val}_{\mathcal{C}} < \alpha \cdot v$

- $\alpha = 1$: the exact version; $\alpha = 1 - \epsilon, \forall \epsilon > 0$: fully approximation.
- **Algorithmic side**: Random sampling, SDP-based algorithms.
- **Hardness side**: NP-hardness or UG-hardness (through PCP theorem).

Approximating CSP

Max-CUT:



- Approximation <=> Distinguishing instances with different values.

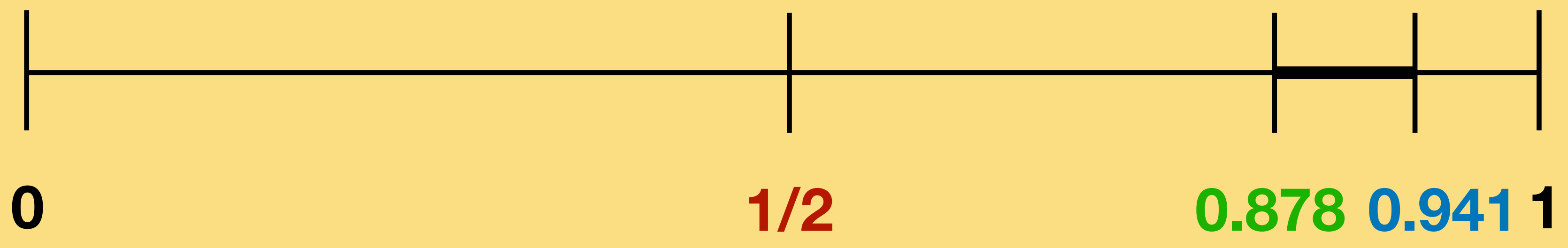
α-approximation: Let $\alpha \in (0, 1]$. For any $v \in (0, m]$, can distinguish the following.

Yes: $\text{val}_{\mathcal{C}} \geq v$ **No:** $\text{val}_{\mathcal{C}} < \alpha \cdot v$

- $\alpha = 1$: the exact version; $\alpha = 1 - \epsilon, \forall \epsilon > 0$: fully approximation.
- **Algorithmic side**: Random sampling, SDP-based algorithms.
- **Hardness side**: NP-hardness or UG-hardness (through PCP theorem).

Approximating CSP

Max-CUT:



- Approximation <=> Distinguishing instances with different values.

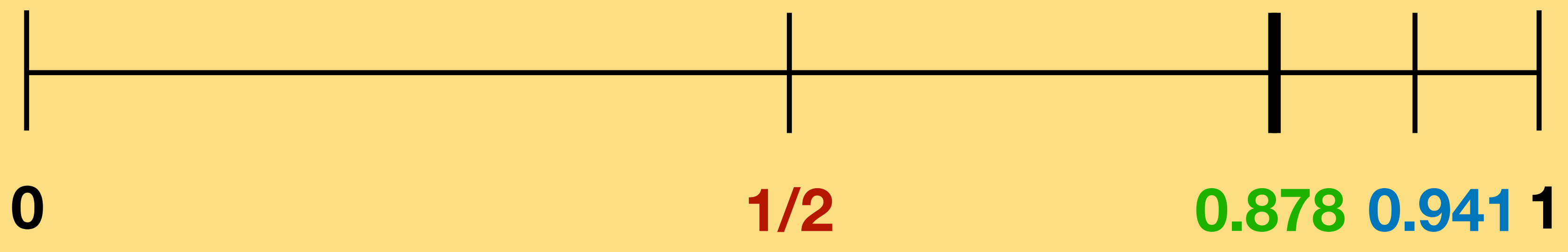
α-approximation: Let $\alpha \in (0, 1]$. For any $v \in (0, m]$, can distinguish the following.

Yes: $\mathsf{val}_\mathcal{C} \geq v$ **No:** $\mathsf{val}_\mathcal{C} < \alpha \cdot v$

- $\alpha = 1$: the exact version; $\alpha = 1 - \epsilon, \forall \epsilon > 0$: fully approximation.
- **Algorithmic side**: Random sampling, SDP-based algorithms.
- **Hardness side**: NP-hardness or UG-hardness (through PCP theorem).

Approximating CSP

Max-CUT:



- Approximation <=> Distinguishing instances with different values.

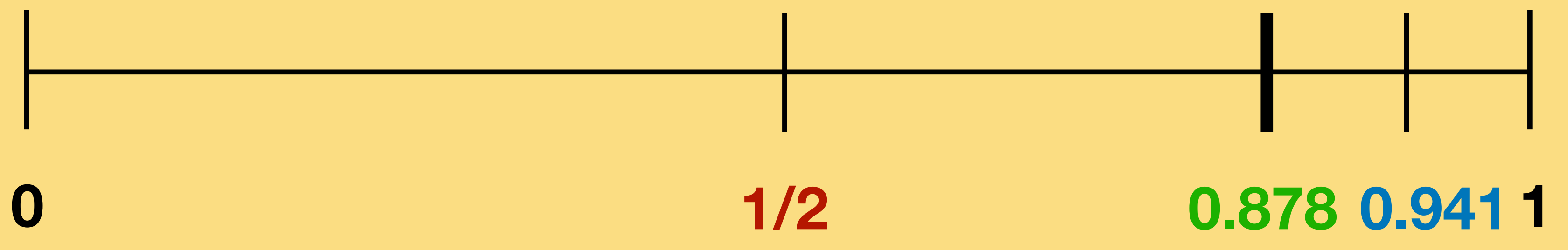
α-approximation: Let $\alpha \in (0,1]$. For any $v \in (0, m]$, can distinguish the following.

Yes: $\text{val}_C \geq v$ **No:** $\text{val}_C < \alpha \cdot v$

- $\alpha = 1$: the exact version; $\alpha = 1 - \epsilon, \forall \epsilon > 0$: fully approximation.
- **Algorithmic side**: Random sampling, SDP-based algorithms.
- **Hardness side**: NP-hardness or UG-hardness (through PCP theorem).

Approximating CSP

Max-CUT:



- Approximation <=> Distinguishing instances with different values.

α-approximation: Let $\alpha \in (0,1]$. For any $v \in (0,m]$, can distinguish the following.

Yes: $\text{val}_C \geq v$ **No:** $\text{val}_C < \alpha \cdot v$

- $\alpha = 1$: the exact version; $\alpha = 1 - \epsilon, \forall \epsilon > 0$: fully approximation.
- **Algorithmic side**: Random sampling, SDP-based algorithms.
- **Hardness side**: NP-hardness or UG-hardness (through PCP theorem).
- Many fascinating results and open problems!

Unifying Theory for Approx. CSP!?

Unifying Theory for Approx. CSP!?

Through the Lens of Streaming Model

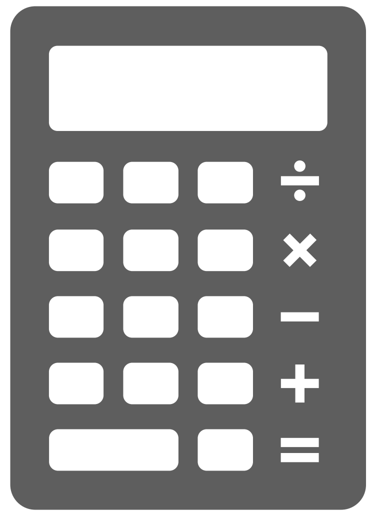
CSP in the Streaming Model

CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

CSP in the Streaming Model

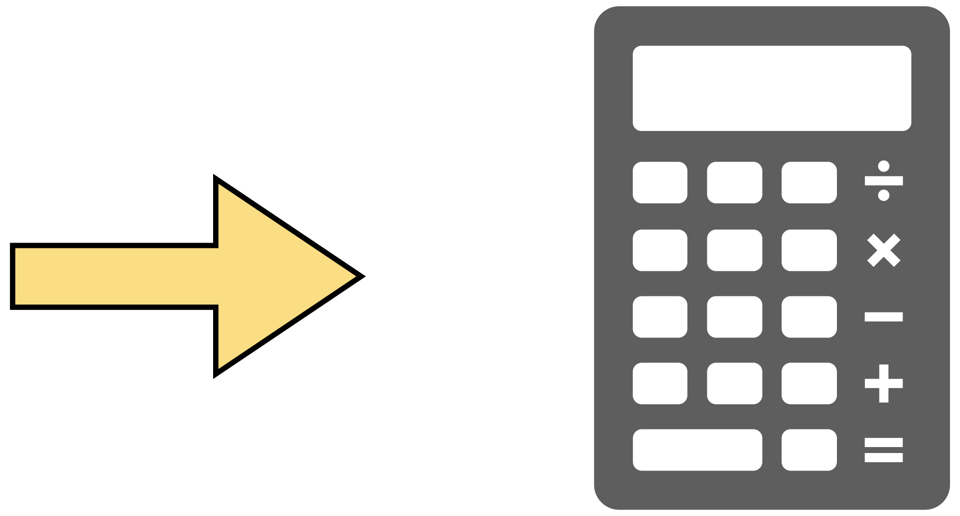
- The input (each constraint) arrives in a stream.



CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

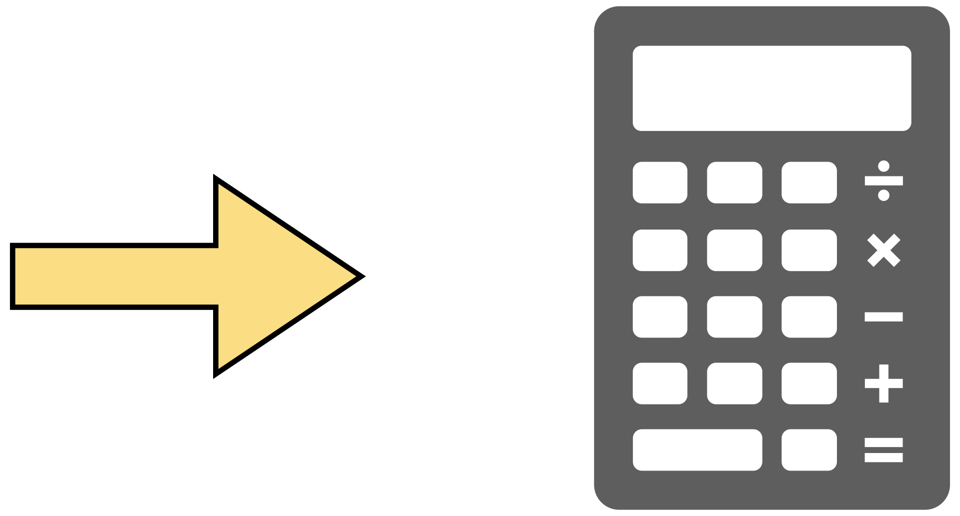
$$x_1 \wedge x_6 \wedge \neg x_4$$



CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

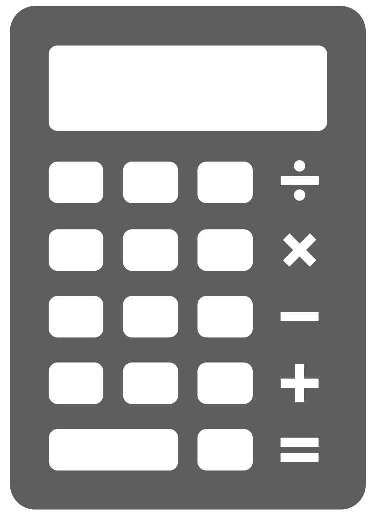
$$\neg x_5 \wedge x_2 \wedge \neg x_{23}$$



CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

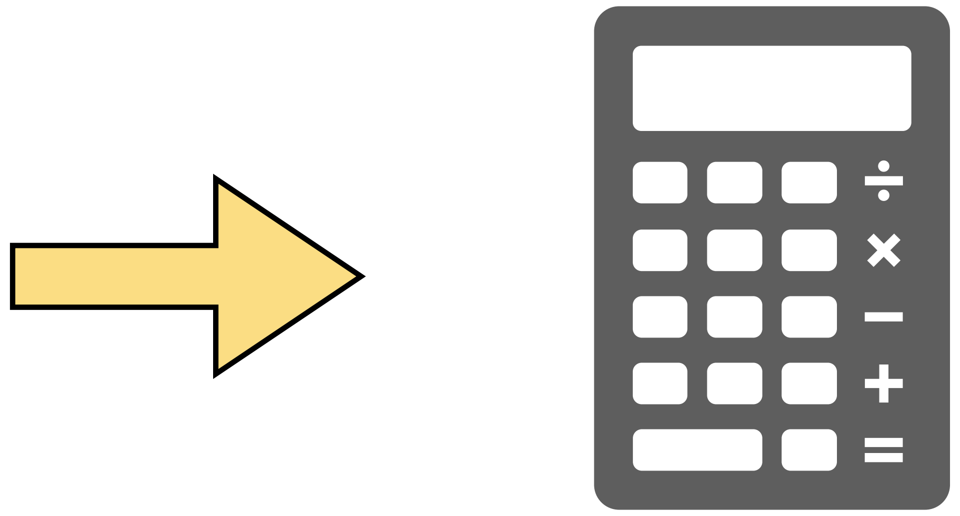
$$x_2 \wedge x_7 \wedge \neg x_{11}$$



CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

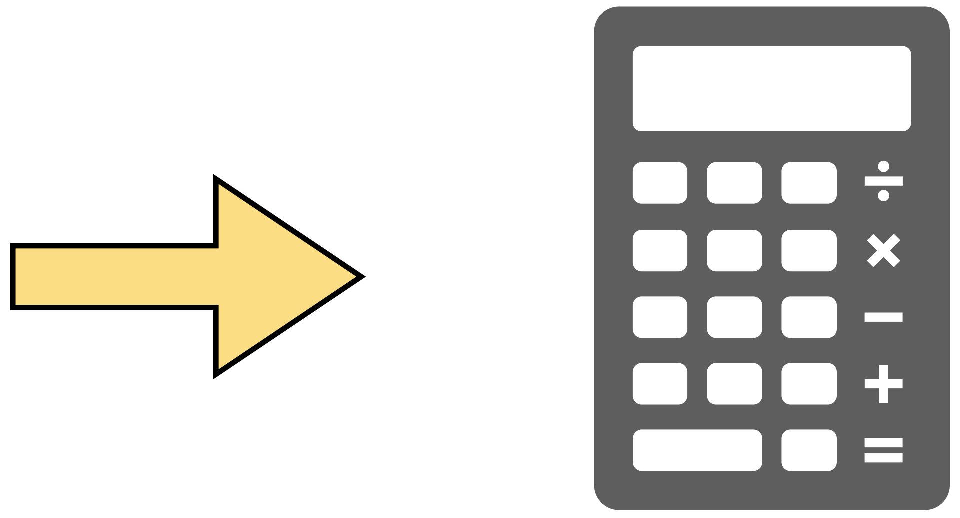
$$\neg x_7 \wedge x_{15} \wedge \neg x_{31}$$



CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

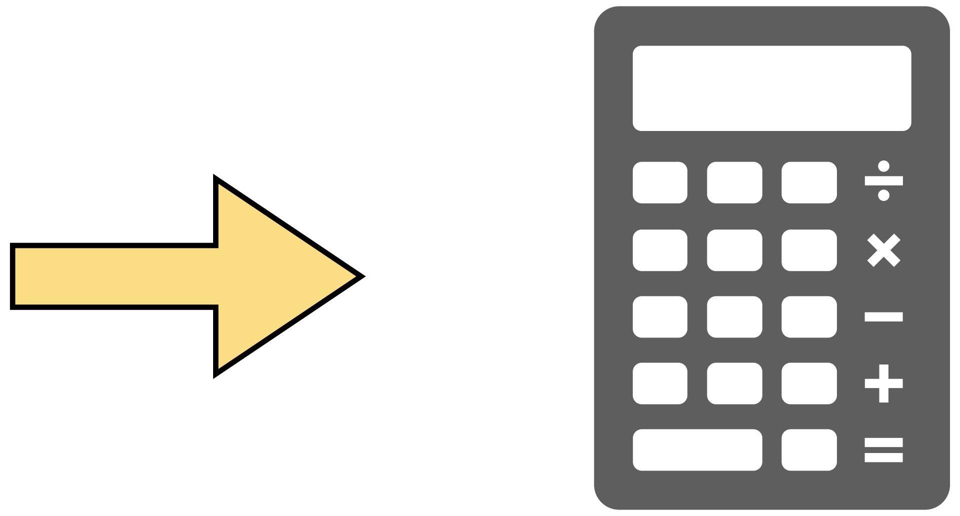
$$\neg x_2 \wedge \neg x_8 \wedge \neg x_{12}$$



CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

$$\neg x_2 \wedge \neg x_8 \wedge \neg x_{12}$$

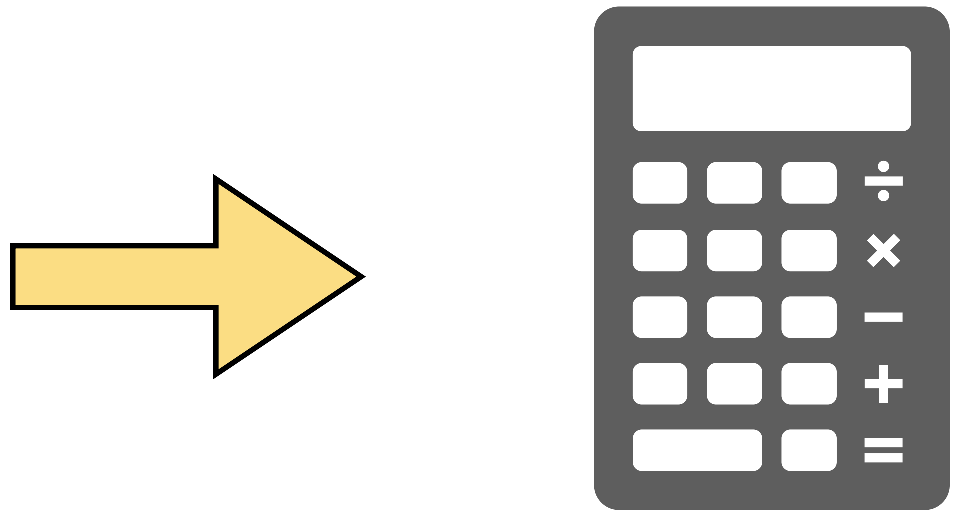


- Only having $o(n)$ or even $O(\log n)$ space.

CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

$$\neg x_2 \wedge \neg x_8 \wedge \neg x_{12}$$

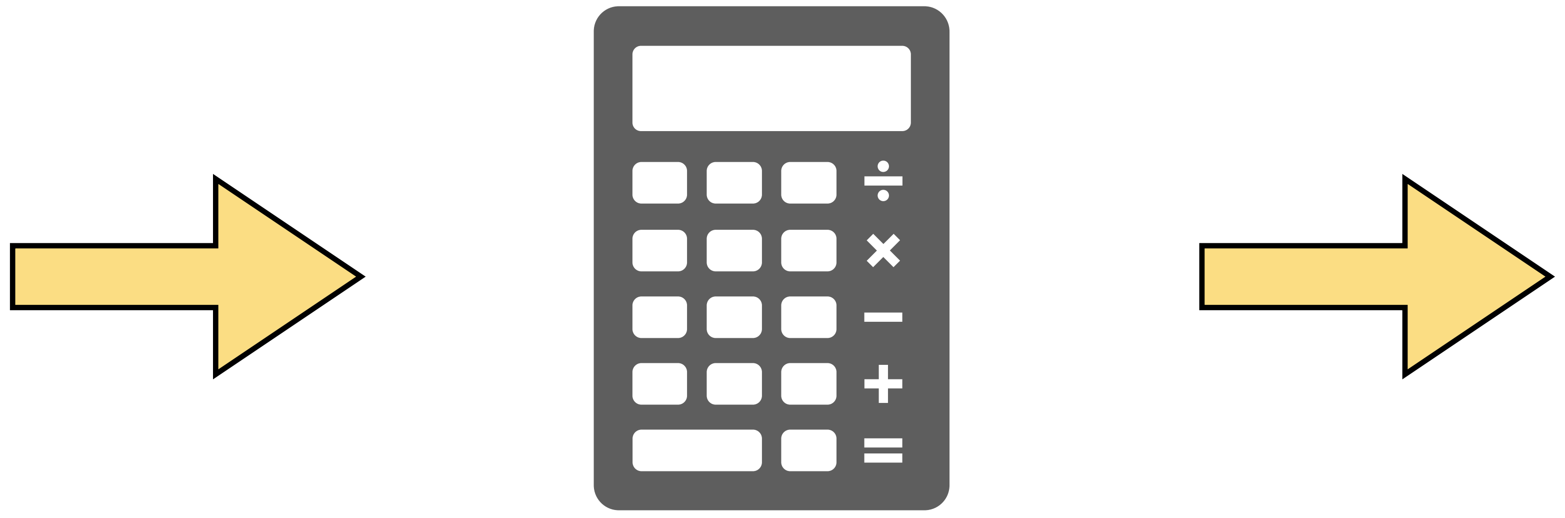


- Only having $o(n)$ or even $O(\log n)$ space.
- **Observation**: Cannot even store an assignment (which requires n bits)!

CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

$\neg x_2 \wedge \neg x_8 \wedge \neg x_{12}$



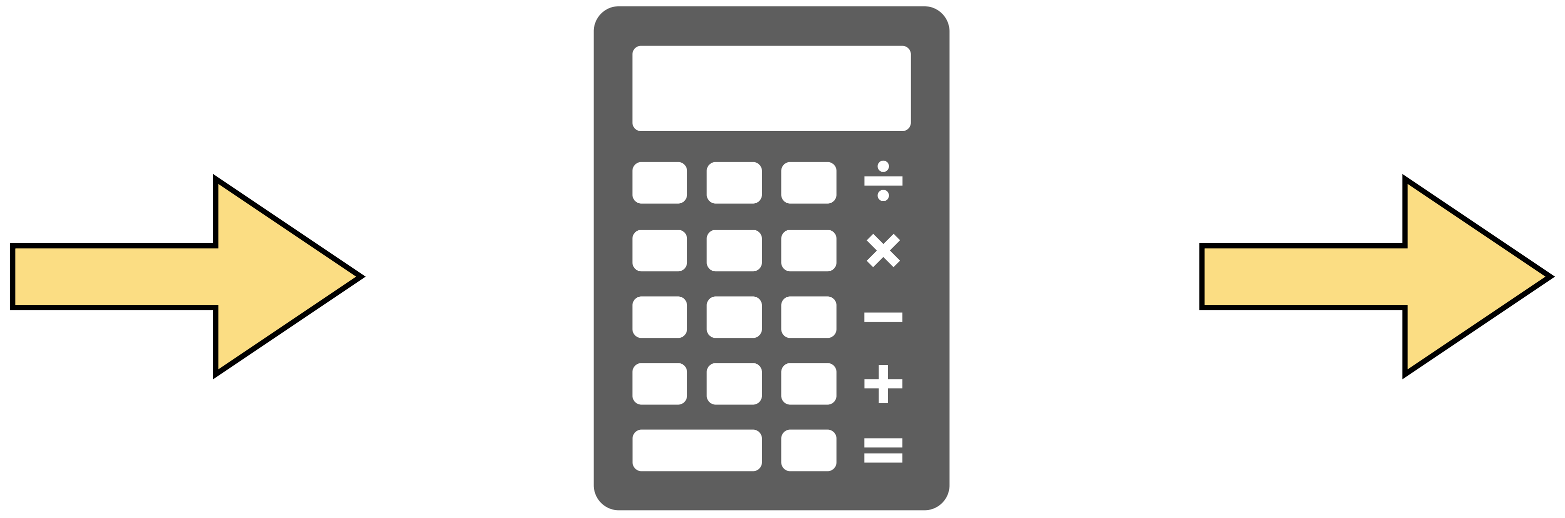
v

- Only having $o(n)$ or even $O(\log n)$ space.
- **Observation**: Cannot even store an assignment (which requires n bits)!
- α **-approximation**: Output an integer v such that

CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

$\neg x_2 \wedge \neg x_8 \wedge \neg x_{12}$



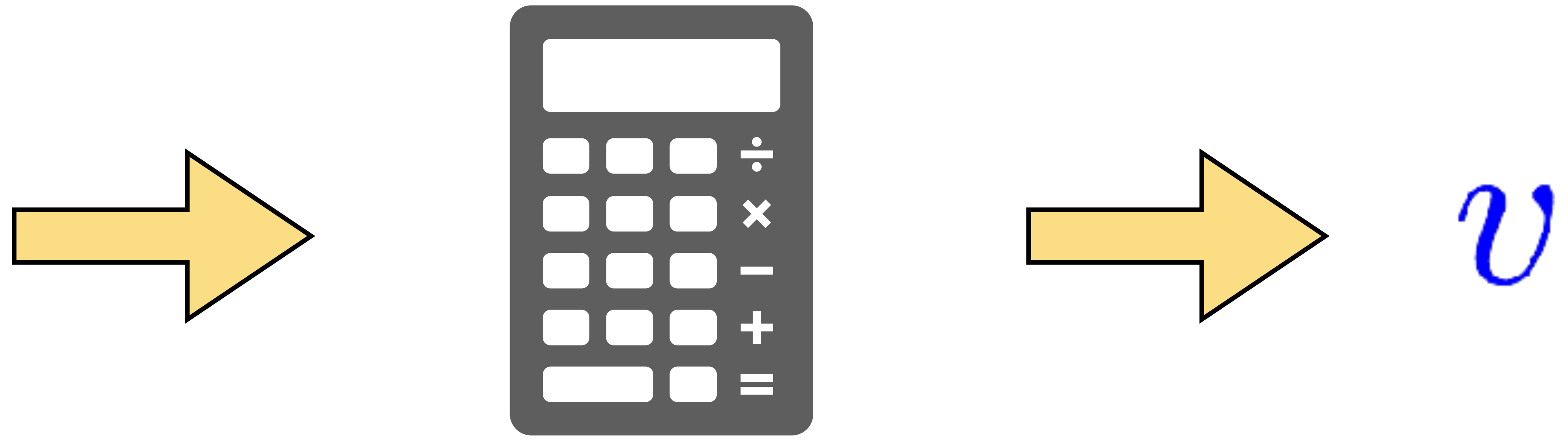
v

- Only having $o(n)$ or even $O(\log n)$ space.
- **Observation**: Cannot even store an assignment (which requires n bits)!
- α**-approximation**: Output an integer v such that
  - there exists an assignment satisfying v constraints and

CSP in the Streaming Model

- The input (each constraint) arrives in a stream.

$\neg x_2 \wedge \neg x_8 \wedge \neg x_{12}$

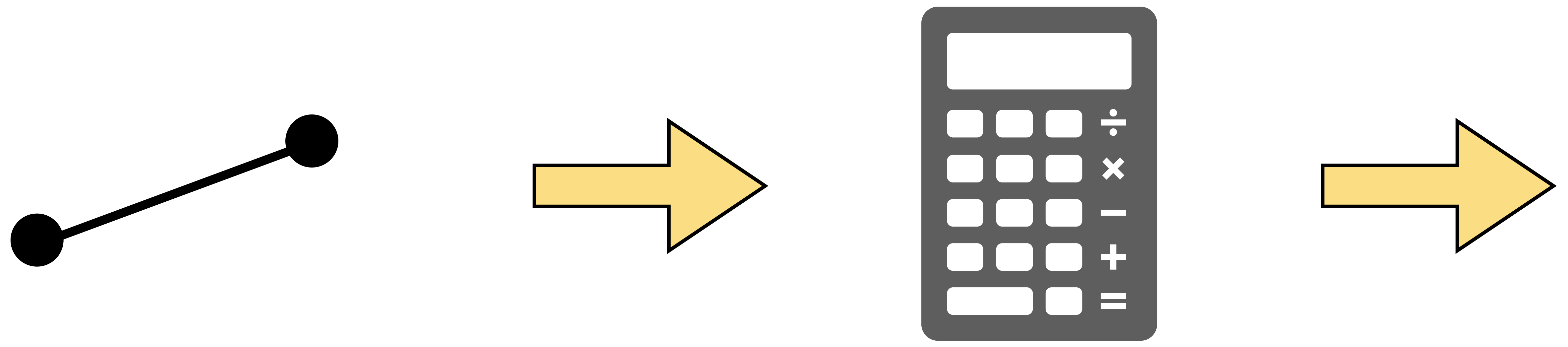


- Only having $o(n)$ or even $O(\log n)$ space.
- **Observation**: Cannot even store an assignment (which requires n bits)!
- α**-approximation**: Output an integer v such that
  - there exists an assignment satisfying v constraints and
  - $v \geq \alpha \cdot \mathsf{val}_C$.

Trivial Random Sampling is Optimal for Max-CUT!

Trivial Random Sampling is Optimal for Max-CUT!

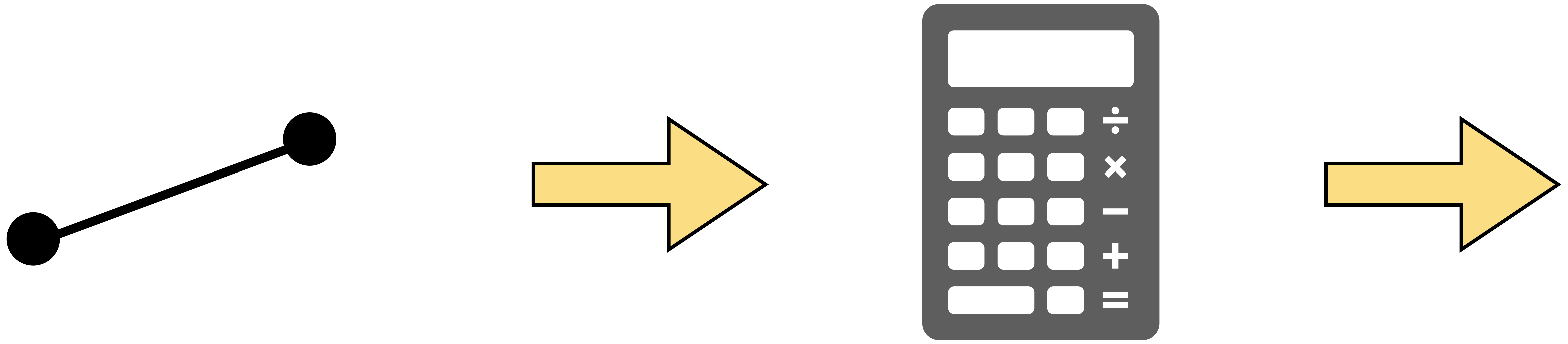
- Trivial random sampling gives **1/2**-approximation using $O(\log n)$ space.



$$\frac{\#\ \text{edges}}{2}$$

Trivial Random Sampling is Optimal for Max-CUT!

- Trivial random sampling gives **1/2**-approximation using $O(\log n)$ space.

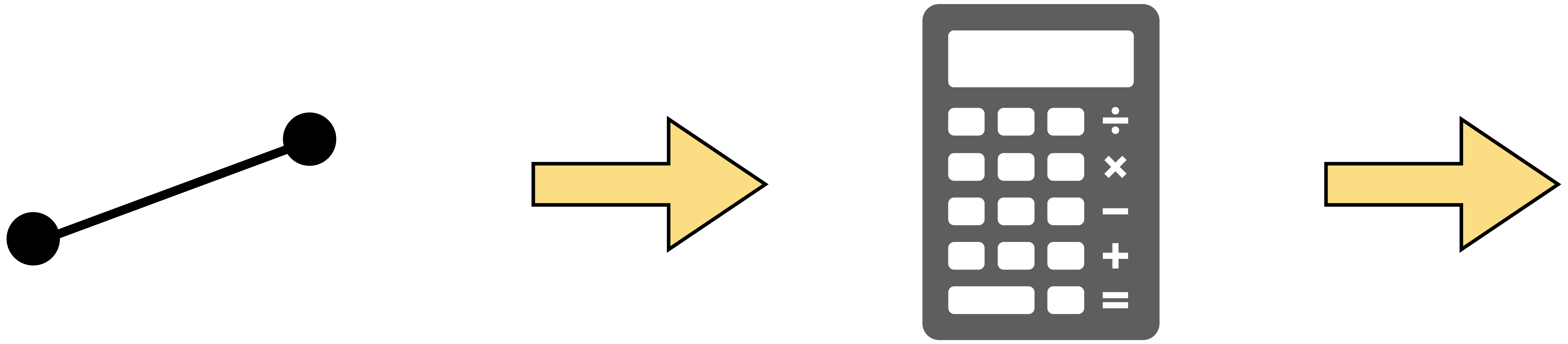


$$\frac{\#\ \text{edges}}{2}$$

- $\forall \epsilon > 0$, there's no (**1/2**+ϵ)-approximation streaming algorithm for Max-CUT!

Trivial Random Sampling is Optimal for Max-CUT!

- Trivial random sampling gives **1/2**-approximation using $O(\log n)$ space.

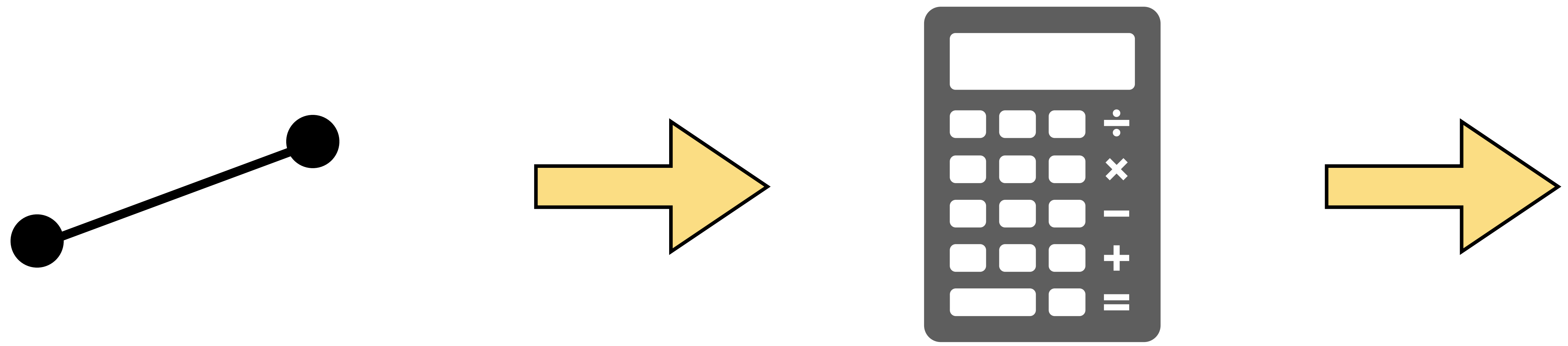


$$\frac{\#\ \text{edges}}{2}$$

- $\forall \epsilon > 0$, there's no (**1/2**$+\epsilon$)-approximation streaming algorithm for Max-CUT!
    - [Kapralov-Khanna-Sudan 15]: $\tilde{\Omega}(\sqrt{n})$ space.

Trivial Random Sampling is Optimal for Max-CUT!

- Trivial random sampling gives **1/2**-approximation using $O(\log n)$ space.

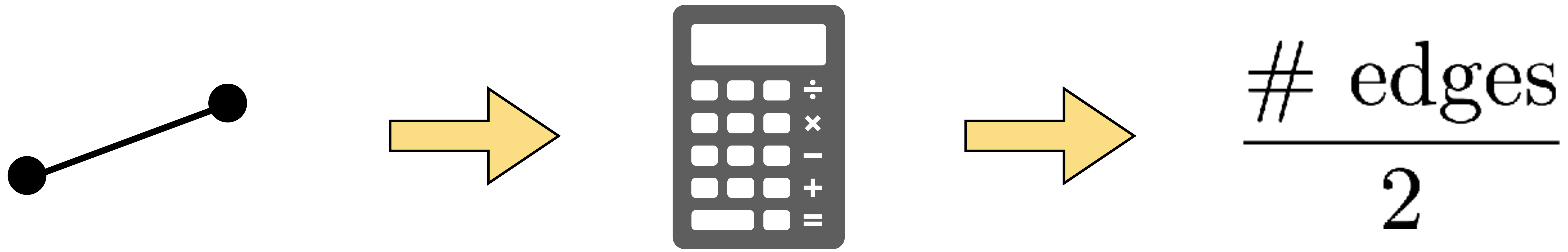


$$\frac{\#\ \text{edges}}{2}$$

- $\forall \epsilon > 0$, there's no (**1/2**+ϵ)-approximation streaming algorithm for Max-CUT!
  - ✦ [Kapralov-Khanna-Sudan 15]: $\tilde{\Omega}(\sqrt{n})$ space.
  - ✦ [Kapralov-Khanna-Sudan-Velingker 17]: 0.99-approx. needs $\Omega(n)$ space.

Trivial Random Sampling is Optimal for Max-CUT!

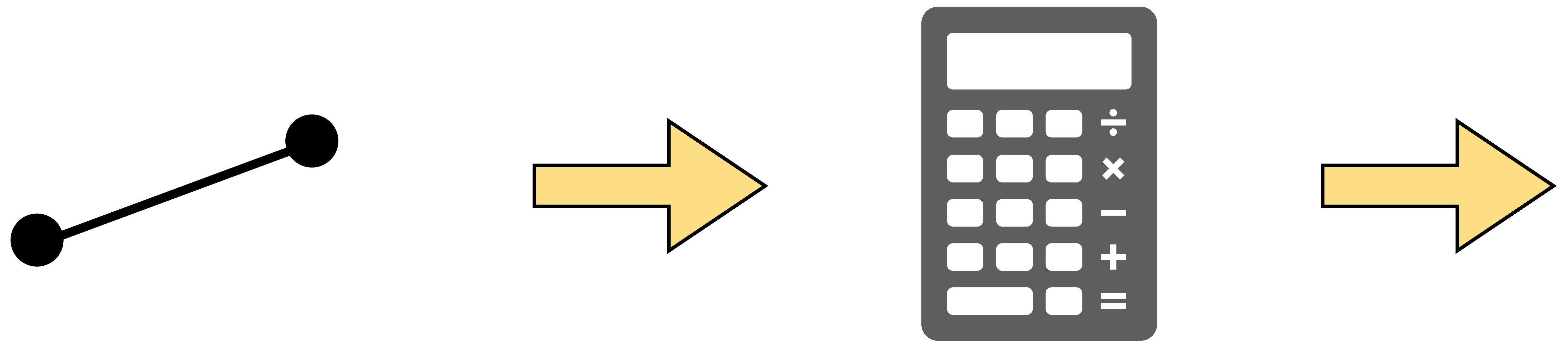
- Trivial random sampling gives **1/2**-approximation using $O(\log n)$ space.



- $\forall \epsilon > 0$, there's no (**1/2**+ϵ)-approximation streaming algorithm for Max-CUT!
  - ✦ [Kapralov-Khanna-Sudan 15]: $\tilde{\Omega}(\sqrt{n})$ space.
  - ✦ [Kapralov-Khanna-Sudan-Velingker 17]: 0.99-approx. needs $\Omega(n)$ space.
  - ✦ [Kapralov-Krachun 19]: $\Omega(n)$ space.

Trivial Random Sampling is Optimal for Max-CUT!

- Trivial random sampling gives **1/2**-approximation using $O(\log n)$ space.



$$\frac{\#\ \text{edges}}{2}$$

- $\forall \epsilon > 0$, there's no (**1/2**+ϵ)-approximation streaming algorithm for Max-CUT!
  - ✦ [Kapralov-Khanna-Sudan 15]: $\tilde{\Omega}(\sqrt{n})$ space.
  - ✦ [Kapralov-Khanna-Sudan-Velingker 17]: 0.99-approx. needs $\Omega(n)$ space.
  - ✦ [Kapralov-Krachun 19]: $\Omega(n)$ space.

There's a SDP-based algorithm which gives **0.878**-approx.

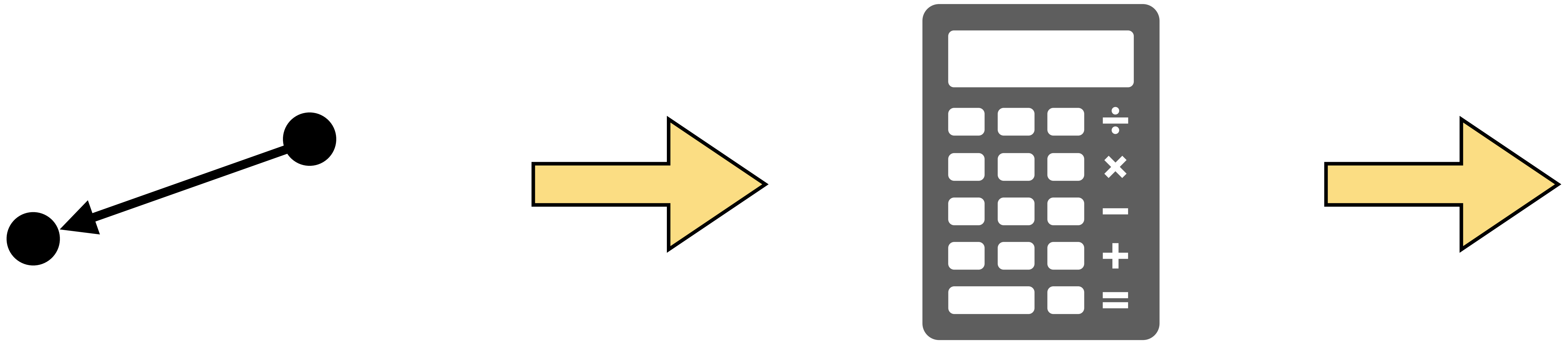
Max-DICUT in the Streaming Model

Max-DICUT in the Streaming Model

- Trivial random sampling now gives **1/4**-approximation using $O(\log n)$ space.

Max-DICUT in the Streaming Model

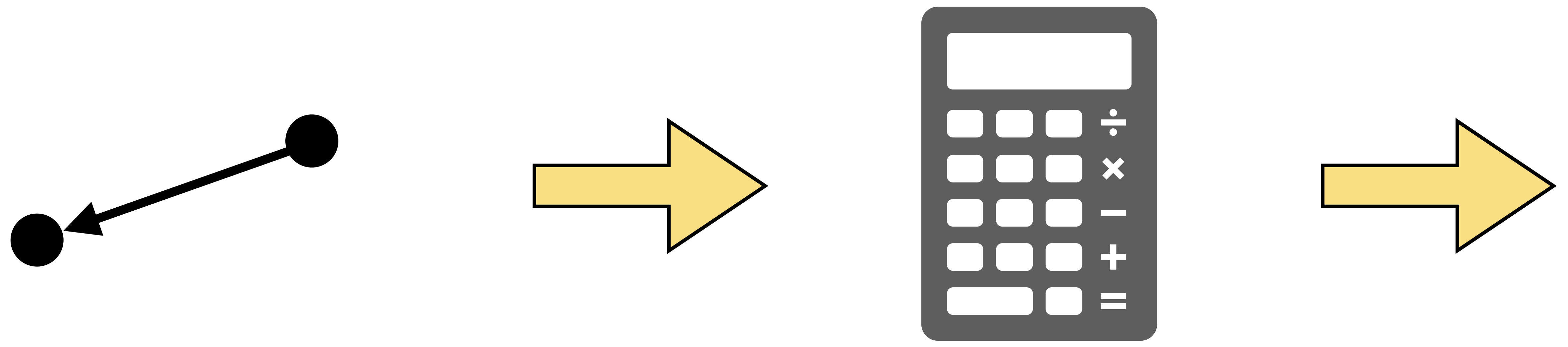
- Trivial random sampling now gives **1/4**-approximation using $O(\log n)$ space.



$$\frac{\#\ \text{edges}}{4}$$

Max-DICUT in the Streaming Model

- Trivial random sampling now gives **1/4**-approximation using $O(\log n)$ space.

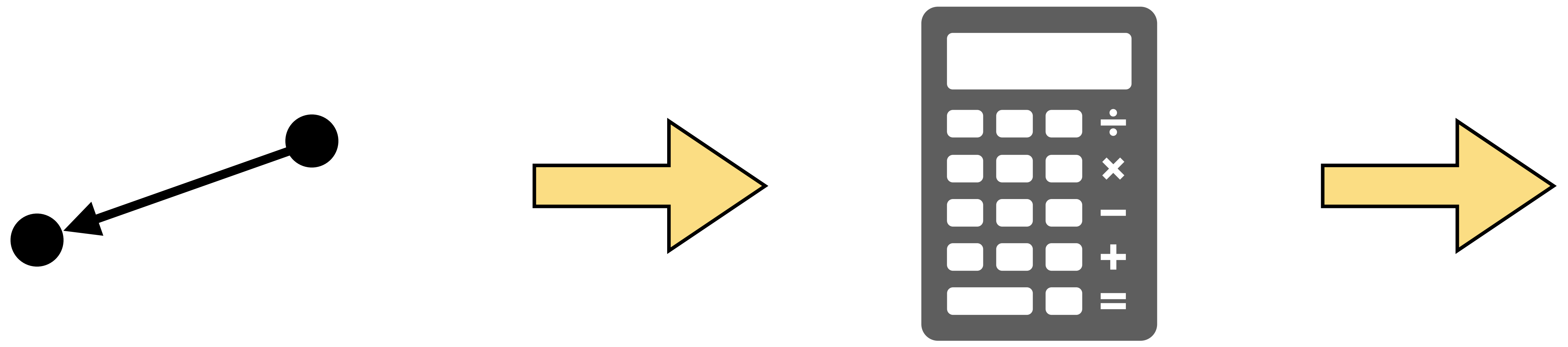


$$\frac{\#\ \text{edges}}{4}$$

- **Hardness side**: no (**1/2**+ϵ)-approximation using $o(n)$ space (from Max-CUT).

Max-DICUT in the Streaming Model

- Trivial random sampling now gives **1/4**-approximation using $O(\log n)$ space.

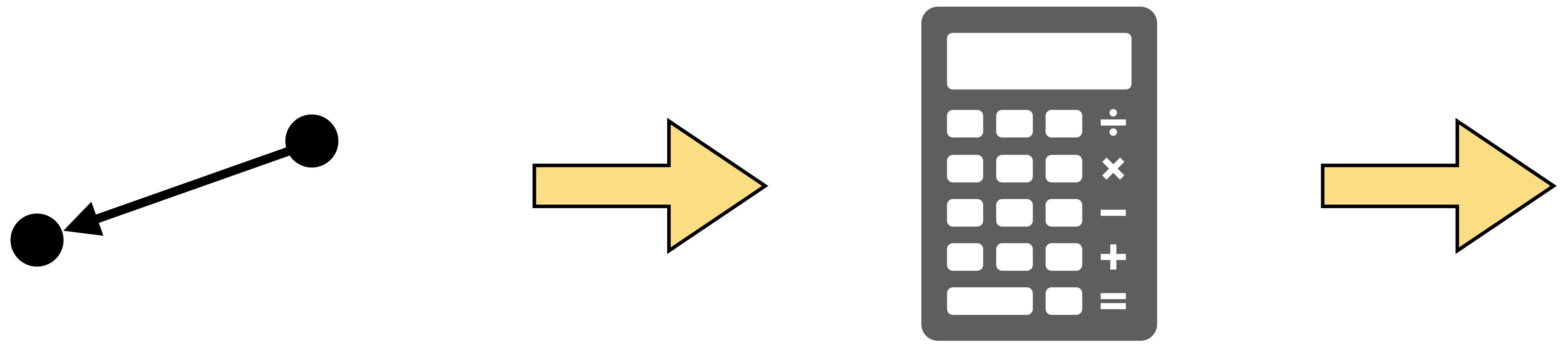


$$\frac{\#\ \text{edges}}{4}$$

- **Hardness side**: no (**1/2**+ϵ)-approximation using $o(n)$ space (from Max-CUT).
- **Algorithm side**: [Guruswami-Velingker-Velusamy 17] gave a **2/5**-approximation.

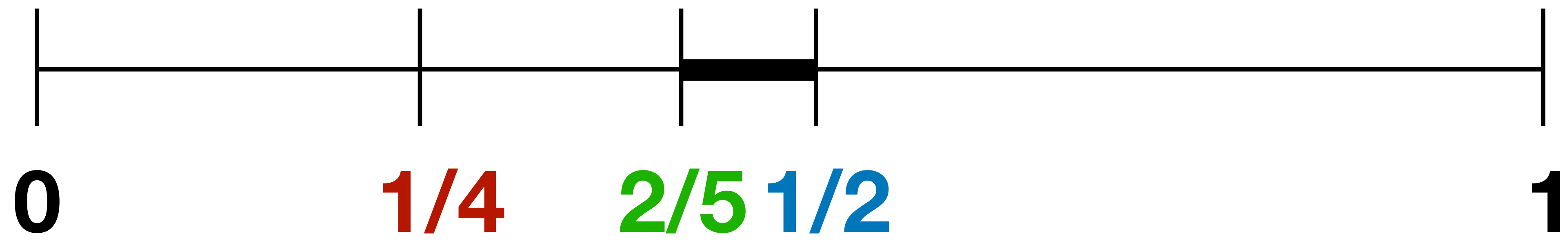
Max-DICUT in the Streaming Model

- Trivial random sampling now gives **1/4**-approximation using $O(\log n)$ space.



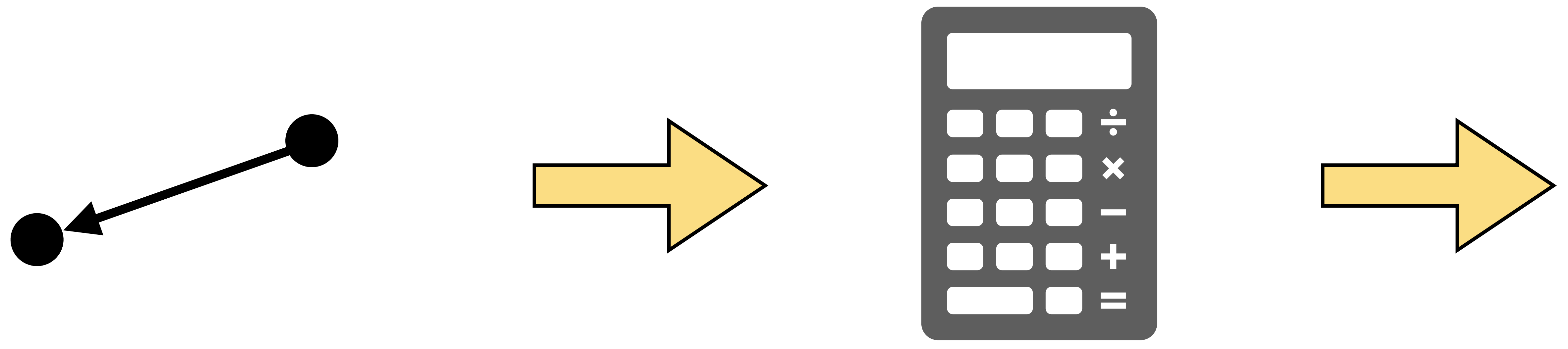
$$\frac{\#\ \text{edges}}{4}$$

- **Hardness side**: no (**1/2**+ϵ)-approximation using $o(n)$ space (from Max-CUT).
- **Algorithm side**: [Guruswami-Velingker-Velusamy 17] gave a **2/5**-approximation.



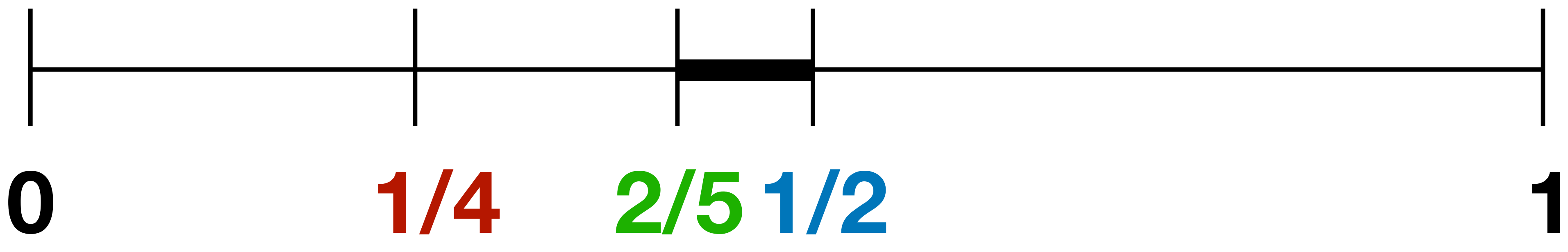
Max-DICUT in the Streaming Model

- Trivial random sampling now gives **1/4**-approximation using $O(\log n)$ space.



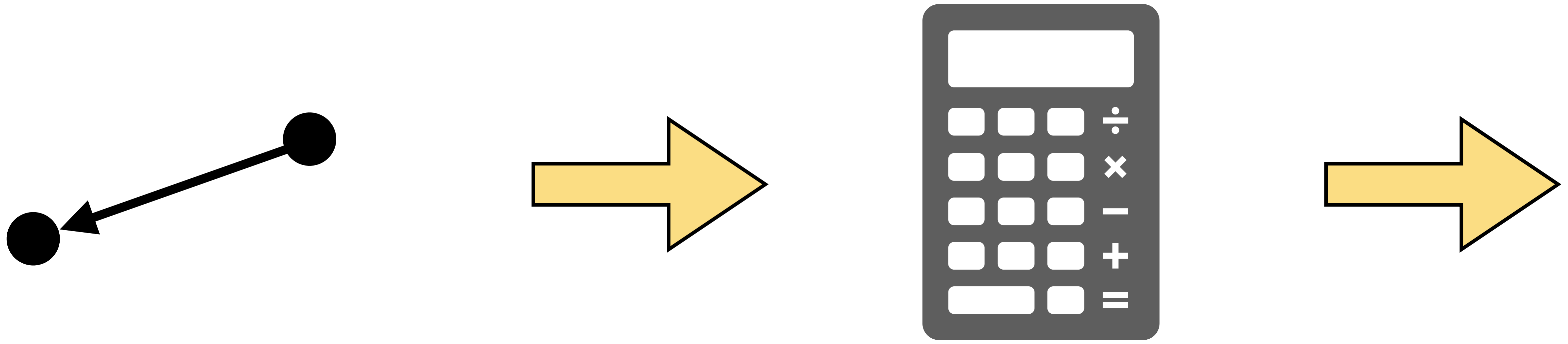
$$\frac{\#\ \text{edges}}{4}$$

- **Hardness side**: no (**1/2**+ϵ)-approximation using $o(n)$ space (from Max-CUT).
- **Algorithm side**: [Guruswami-Velingker-Velusamy 17] gave a **2/5**-approximation.
- What's the "right approximation ratio"?



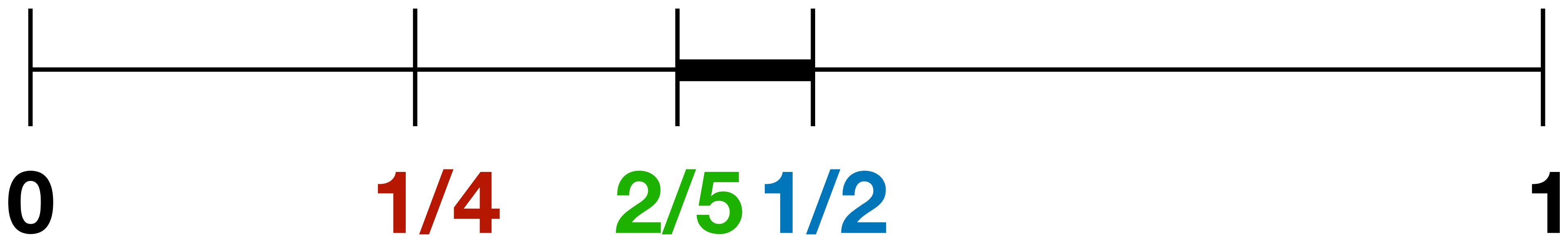
Max-DICUT in the Streaming Model

- Trivial random sampling now gives **1/4**-approximation using $O(\log n)$ space.



$$\frac{\#\ \text{edges}}{4}$$

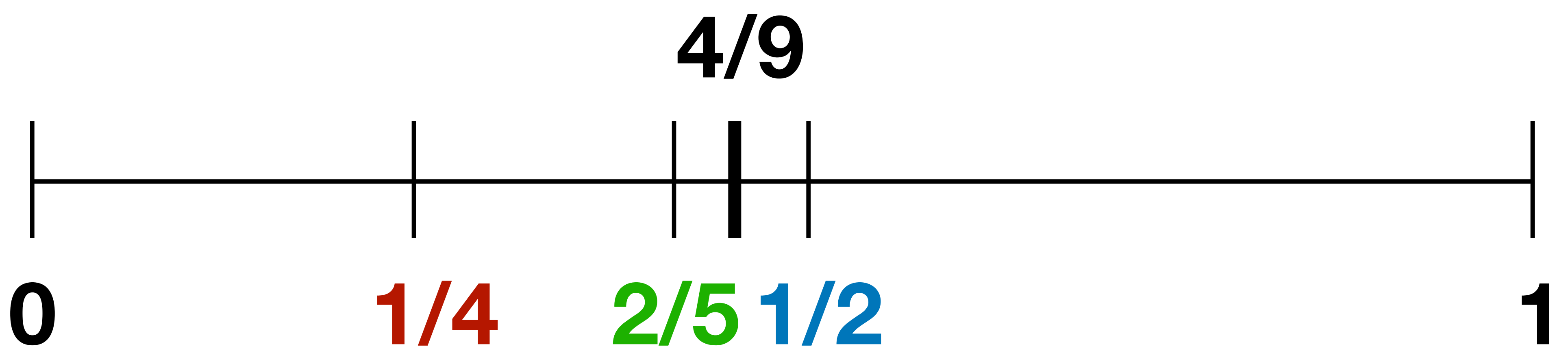
- **Hardness side**: no (**1/2**+ϵ)-approximation using $o(n)$ space (from Max-CUT).
- **Algorithm side**: [Guruswami-Velingker-Velusamy 17] gave a **2/5**-approximation.
- What's the "right approximation ratio"?
- What about other CSP?



Our Results

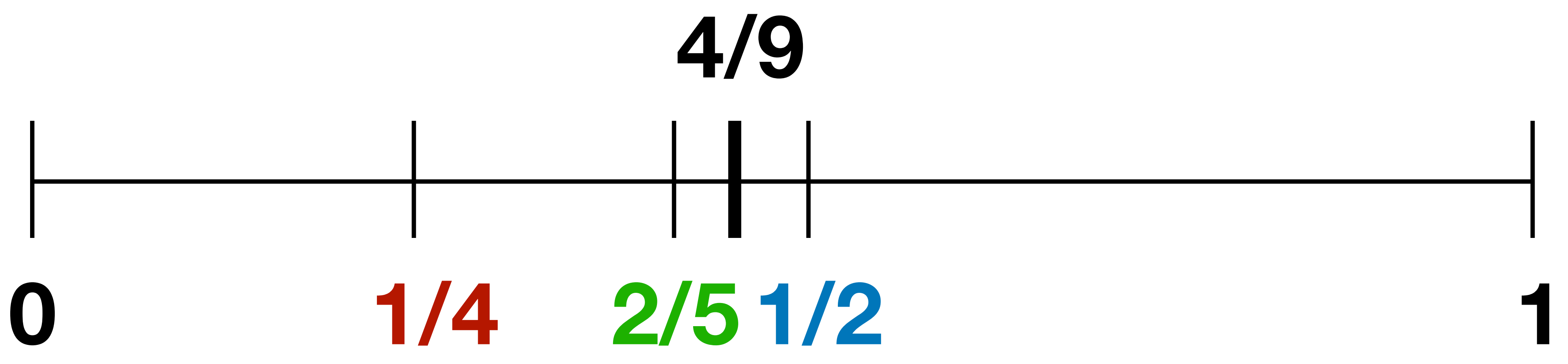
Our Results

- The answer of Max-DICUT is **4/9** 😳



Our Results

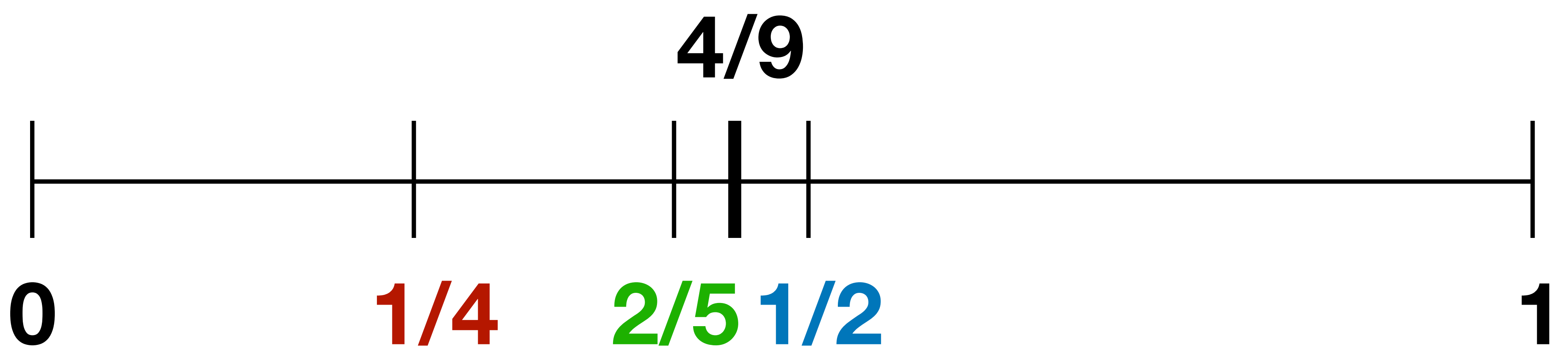
- The answer of Max-DICUT is **4/9** 😳



- Further, we characterize the approximation ratio of every boolean 2CSP!

Our Results

- The answer of Max-DICUT is **4/9** 😳

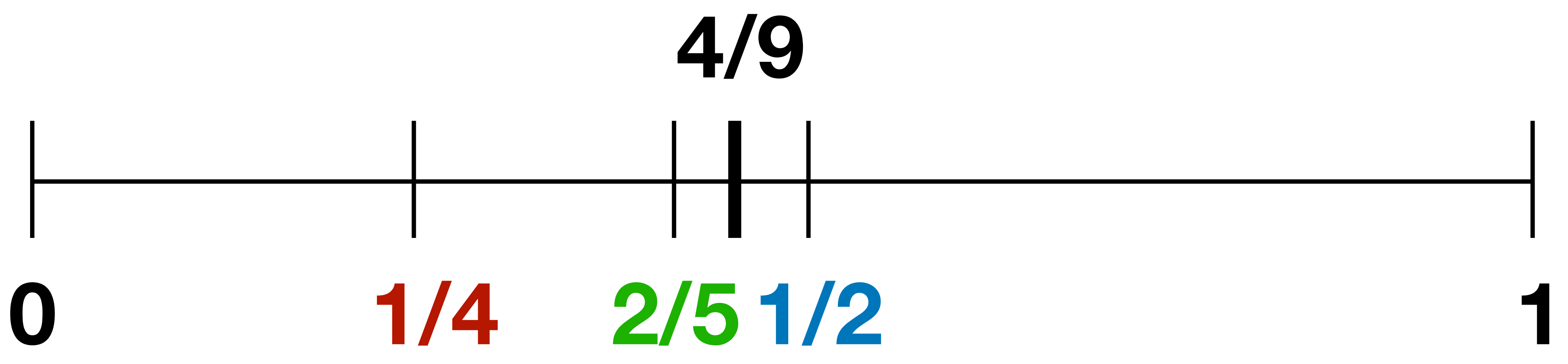


- Further, we characterize the approximation ratio of every boolean 2CSP!

Theorem (main).

Our Results

- The answer of Max-DICUT is **4/9** 😳

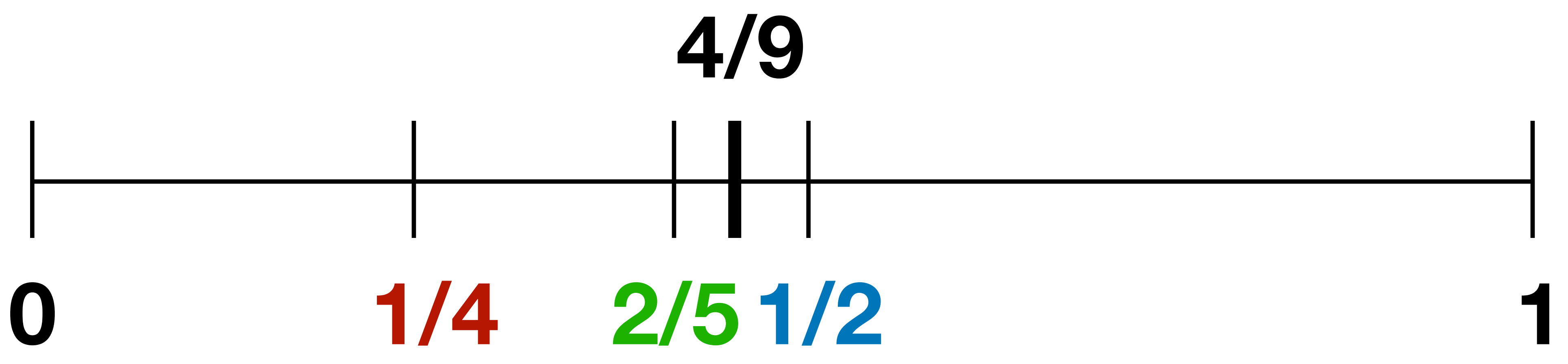


- Further, we characterize the approximation ratio of every boolean 2CSP!

Theorem (main).

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda \in (0,1]$ such that $\forall \epsilon > 0$,

Our Results



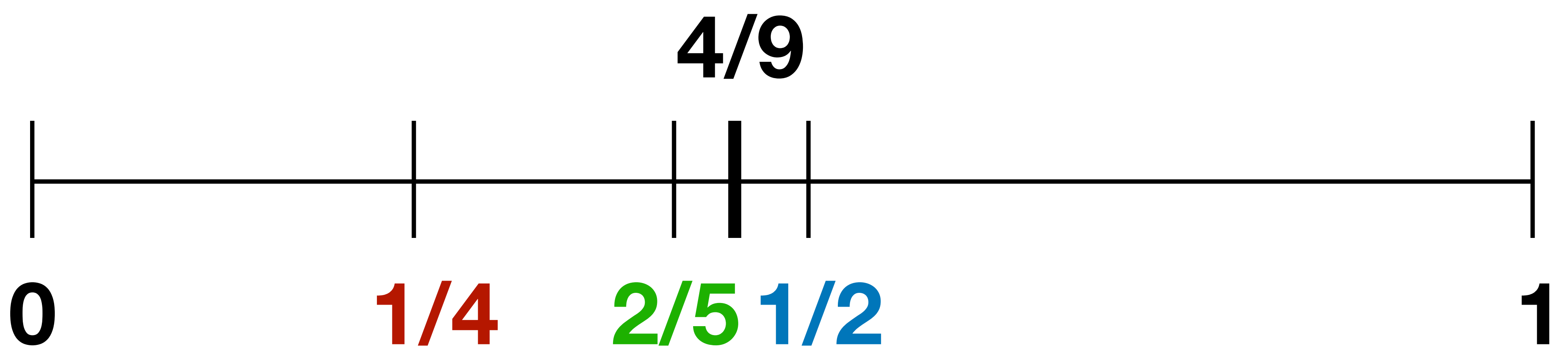
- The answer of Max-DICUT is **4/9** 😳
- Further, we characterize the approximation ratio of every boolean 2CSP!

Theorem (main).

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda \in (0,1]$ such that $\forall \epsilon > 0$,

(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

Our Results



- The answer of Max-DICUT is **4/9** 😳
- Further, we characterize the approximation ratio of every boolean 2CSP!

<u>Theorem (main).</u>

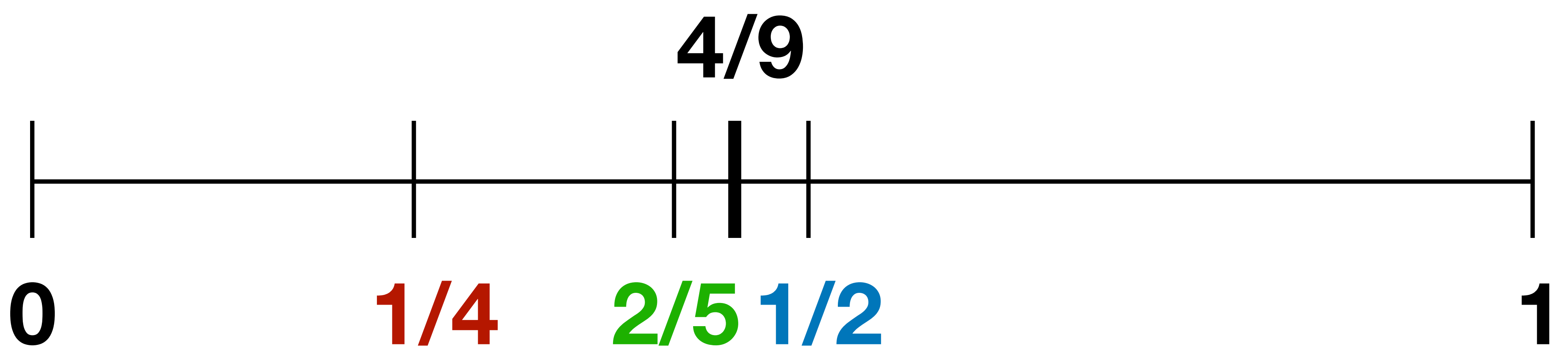
For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda \in (0,1]$ such that $\forall \epsilon > 0$,

(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Our Results

- The answer of Max-DICUT is **4/9** 😳



- Further, we characterize the approximation ratio of every boolean 2CSP!

Theorem (main).

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda \in (0,1]$ such that $\forall \epsilon > 0$,

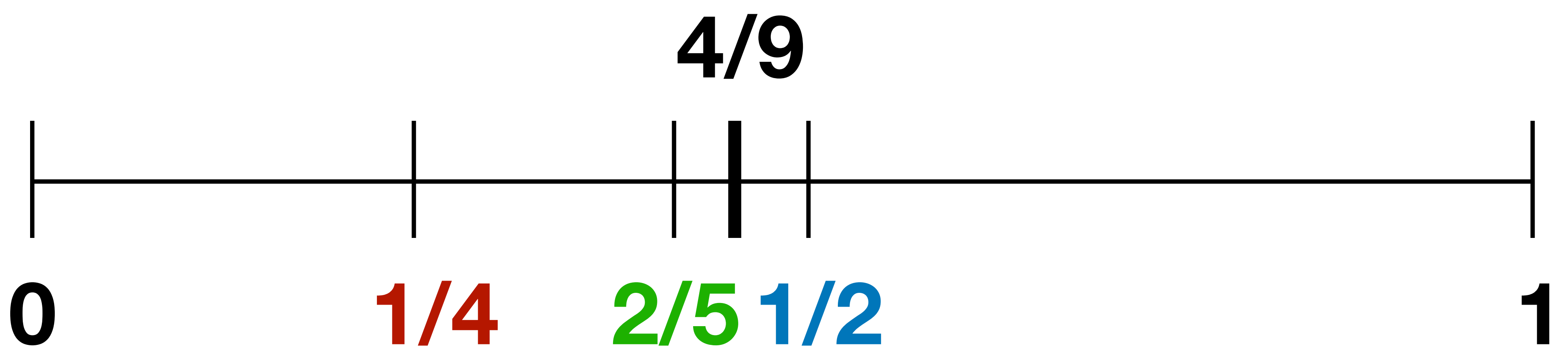
(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Λ	α_Λ	τ_Λ	Reference

Our Results

- The answer of Max-DICUT is **4/9** 😳



- Further, we characterize the approximation ratio of every boolean 2CSP!

Theorem (main).

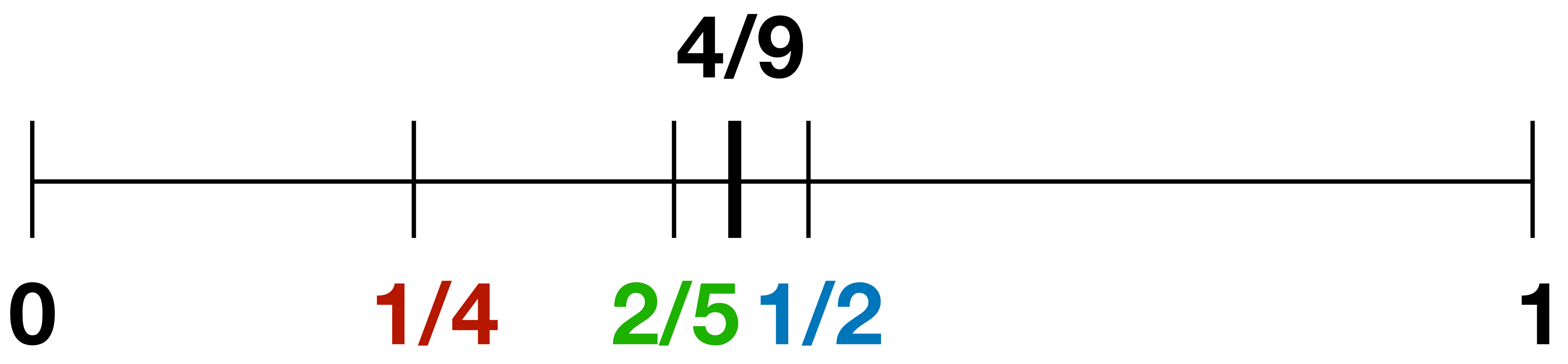
For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda \in (0,1]$ such that $\forall \epsilon > 0$,

(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Λ	α_Λ	τ_Λ	Reference
XOR	$\frac{1}{2}$	1	[KK19]

Our Results



- The answer of Max-DICUT is **4/9** 😳
- Further, we characterize the approximation ratio of every boolean 2CSP!

Theorem (main).

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda \in (0,1]$ such that $\forall \epsilon > 0$,

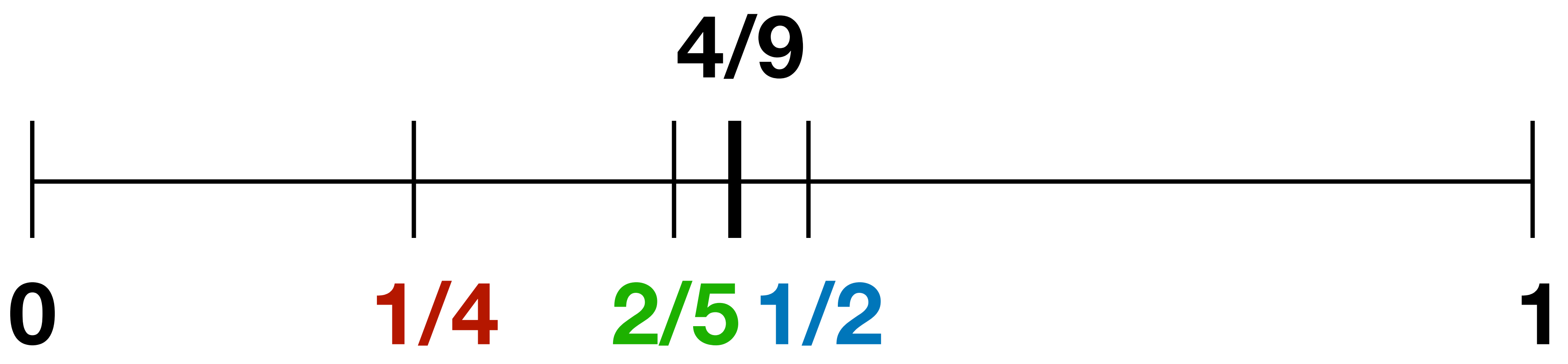
(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Λ	α_Λ	τ_Λ	Reference
XOR	$\frac{1}{2}$	1	[KK19]
AND	$\frac{4}{9}$	$\frac{1}{2}$	This work

Our Results

- The answer of Max-DICUT is **4/9** 😳



- Further, we characterize the approximation ratio of every boolean 2CSP!

Theorem (main).

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda \in (0,1]$ such that $\forall \epsilon > 0$,

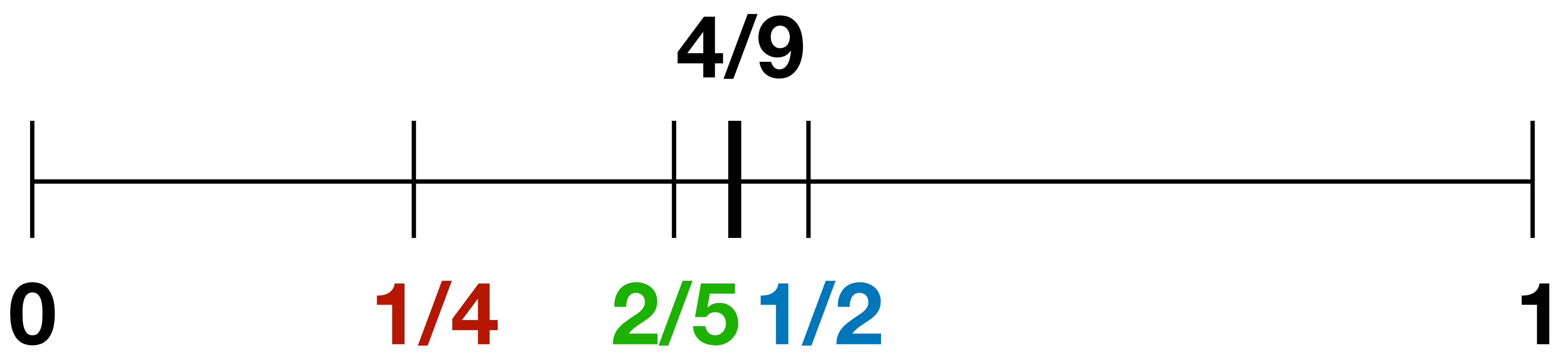
(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Λ	α_Λ	τ_Λ	Reference
XOR	$\frac{1}{2}$	1	[KK19]
AND	$\frac{4}{9}$	$\frac{1}{2}$	This work
EOR	$\frac{3}{4}$	$\frac{1}{2}$	This work

Our Results

- The answer of Max-DICUT is **4/9** 😳



- Further, we characterize the approximation ratio of every boolean 2CSP!

Theorem (main).

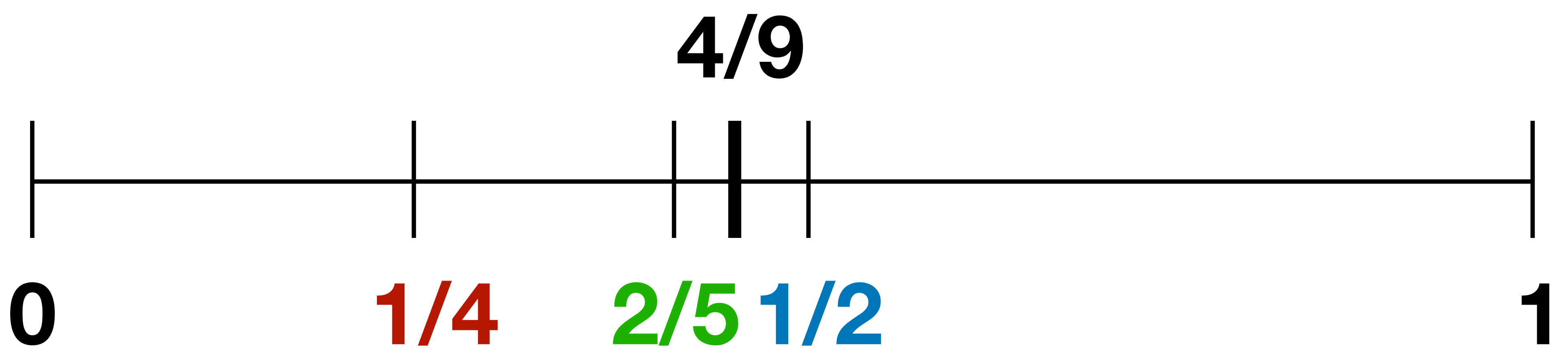
For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda \in (0,1]$ such that $\forall \epsilon > 0$,

(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Λ	α_Λ	τ_Λ	Reference
XOR	$\frac{1}{2}$	1	[KK19]
AND	$\frac{4}{9}$	$\frac{1}{2}$	This work
EOR	$\frac{3}{4}$	$\frac{1}{2}$	This work
OR	$\frac{\sqrt{2}}{2}$	$\frac{1}{2}$	This work

Our Results



- The answer of Max-DICUT is **4/9** 😳
- Further, we characterize the approximation ratio of every boolean 2CSP!

Theorem (main).

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda \in (0,1]$ such that $\forall \epsilon > 0$,

(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Λ	α_Λ	τ_Λ	Reference
XOR	$\frac{1}{2}$	1	[KK19]
AND	$\frac{4}{9}$	$\frac{1}{2}$	This work
EOR	$\frac{3}{4}$	$\frac{1}{2}$	This work
OR	$\frac{\sqrt{2}}{2}$	$\frac{1}{2}$	This work

Can be extended to Max k-SAT!

Algorithms

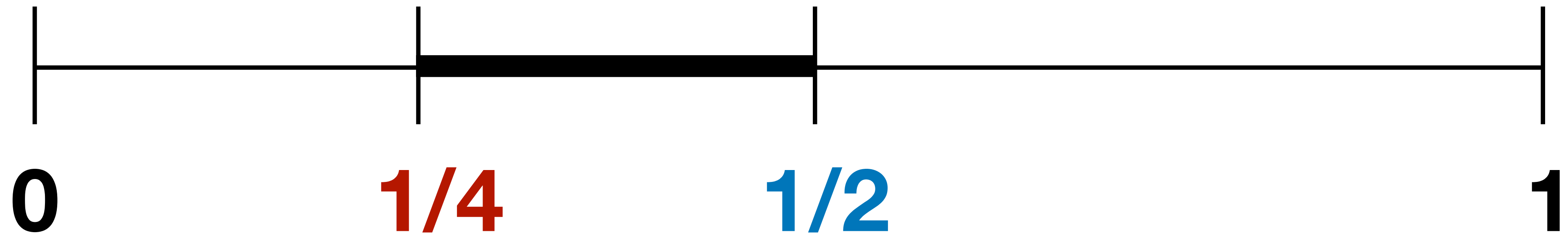
Algorithms

with a focus on Max-DICUT

Warm-up: 2/5-Approximation by [GVV17]

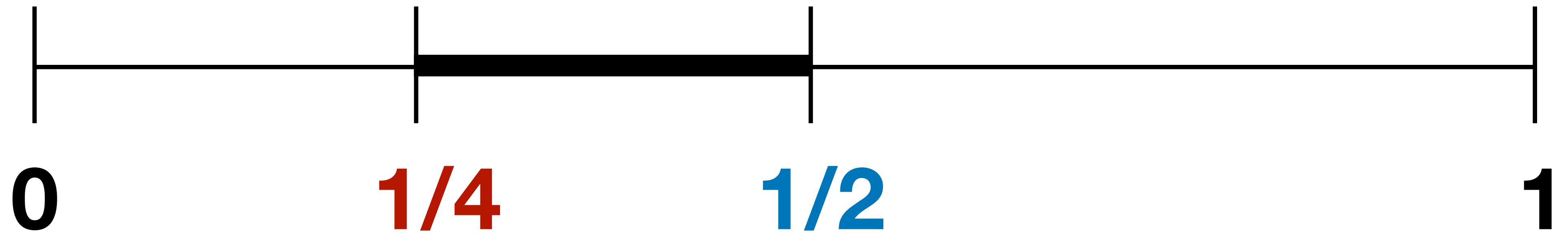
Warm-up: 2/5-Approximation by [GVV17]

- **Recall**: Trivial algorithm gives **1/4**-approx. while **1/2**-approx. is hard.



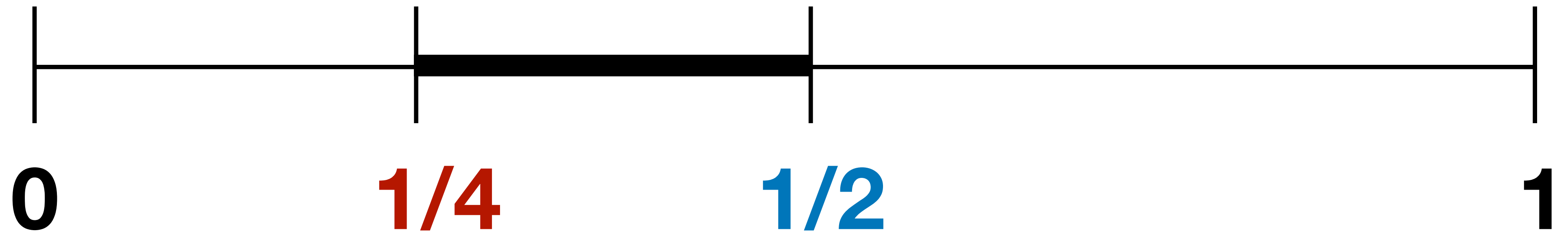
Warm-up: 2/5-Approximation by [GVV17]

- **Recall**: Trivial algorithm gives **1/4**-approx. while **1/2**-approx. is hard.
- **Idea**: Consider the ***bias*** of each vertex.



Warm-up: 2/5-Approximation by [GVV17]

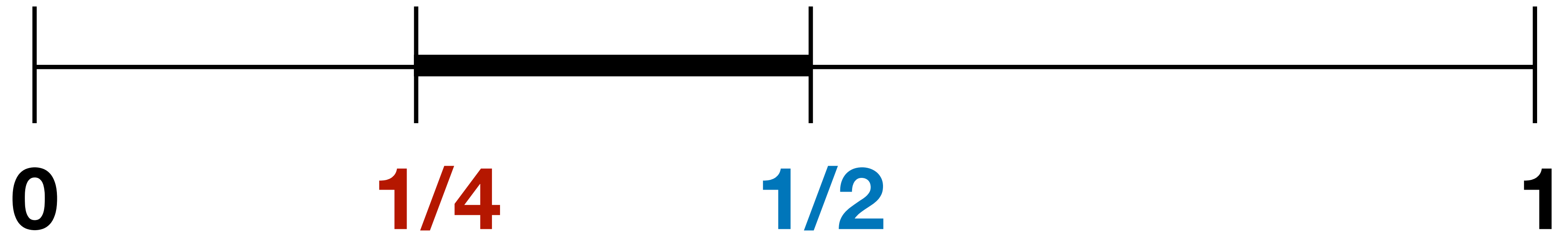
- **Recall**: Trivial algorithm gives **1/4**-approx. while **1/2**-approx. is hard.
- **Idea**: Consider the ***bias*** of each vertex.



Definition (bias and total bias):

Warm-up: 2/5-Approximation by [GVV17]

- **Recall**: Trivial algorithm gives **1/4**-approx. while **1/2**-approx. is hard.
- **Idea**: Consider the ***bias*** of each vertex.

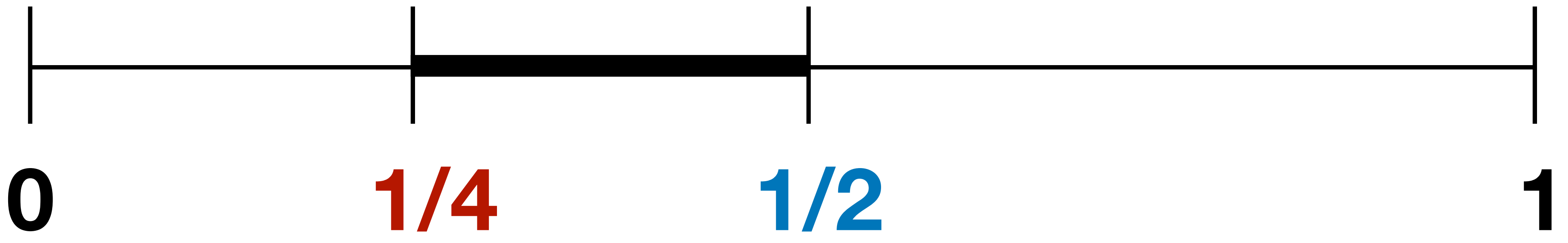


Definition (bias and total bias):

- $\text{bias}(v) = \text{in-degree} - \text{out-degree}$

Warm-up: 2/5-Approximation by [GVV17]

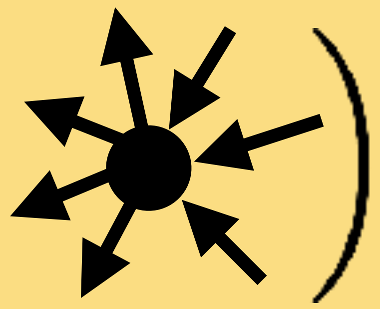
- **Recall**: Trivial algorithm gives **1/4**-approx. while **1/2**-approx. is hard.
- **Idea**: Consider the ***bias*** of each vertex.



<u>Definition</u> <u>(bias and total bias):</u>

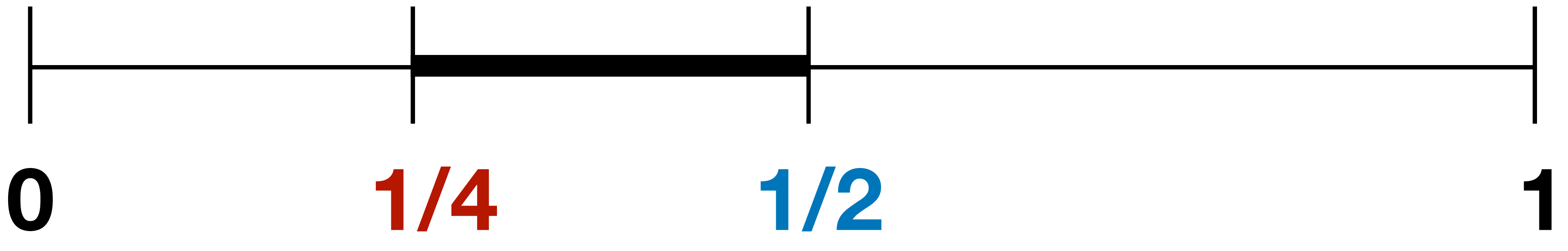
- $\text{bias}(v) = \text{in-degree} - \text{out-degree}$

<u>Example:</u>

$\text{bias}\Big($  $\Big) = -1$

Warm-up: 2/5-Approximation by [GVV17]

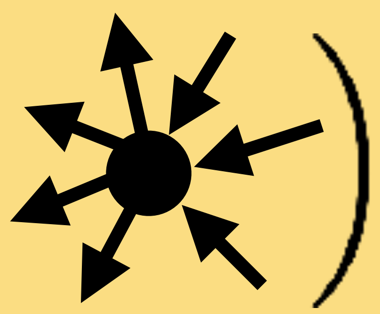
- **Recall**: Trivial algorithm gives **1/4**-approx. while **1/2**-approx. is hard.
- **Idea**: Consider the ***bias*** of each vertex.



Definition (bias and total bias):

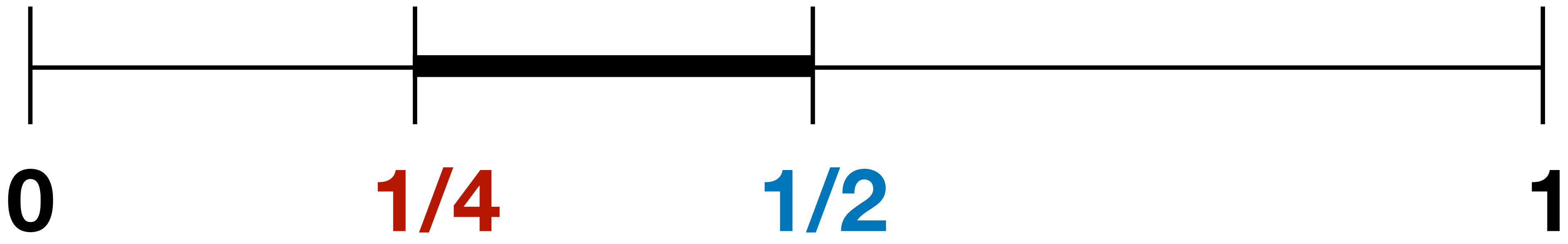
- $\text{bias}(v) = \text{in-degree} - \text{out-degree}$
- Total bias: $B = \sum_v |\text{bias}(v)| \in [0, 2m]$

Example:

bias() $= -1$

Warm-up: 2/5-Approximation by [GVV17]

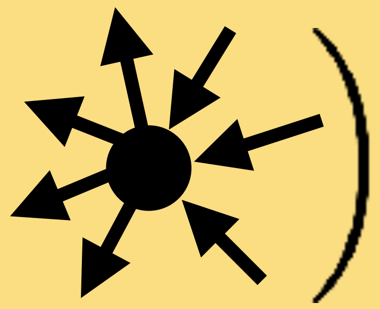
- **Recall**: Trivial algorithm gives **1/4**-approx. while **1/2**-approx. is hard.
- **Idea**: Consider the ***bias*** of each vertex.



Definition (bias and total bias):

- $\text{bias}(v) = \text{in-degree} - \text{out-degree}$
- Total bias: $B = \sum_v |\text{bias}(v)| \in [0, 2m]$

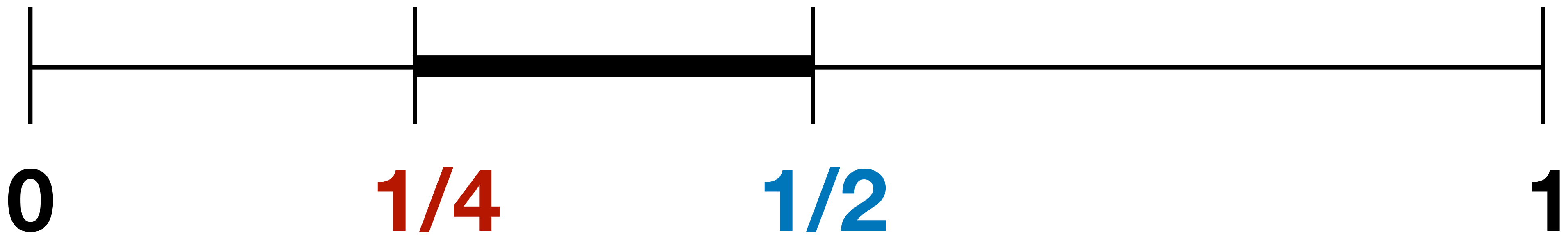
Example:

$\text{bias}\Big($  $\Big) = -1$

Total bias B can be estimated in $O(\log n)$ space!

Warm-up: 2/5-Approximation by [GVV17]

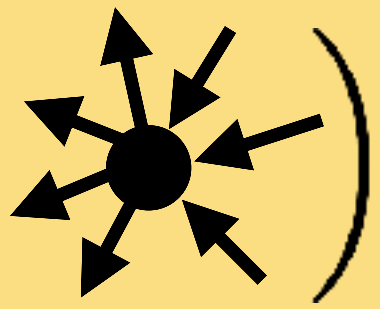
- **Recall**: Trivial algorithm gives **1/4**-approx. while **1/2**-approx. is hard.
- **Idea**: Consider the ***bias*** of each vertex.



<u>Definition</u> <u>(bias and total bias):</u>

- $\text{bias}(v) = \text{in-degree} - \text{out-degree}$
- Total bias: $B = \sum_v |\text{bias}(v)| \in [0, 2m]$

<u>Example:</u>

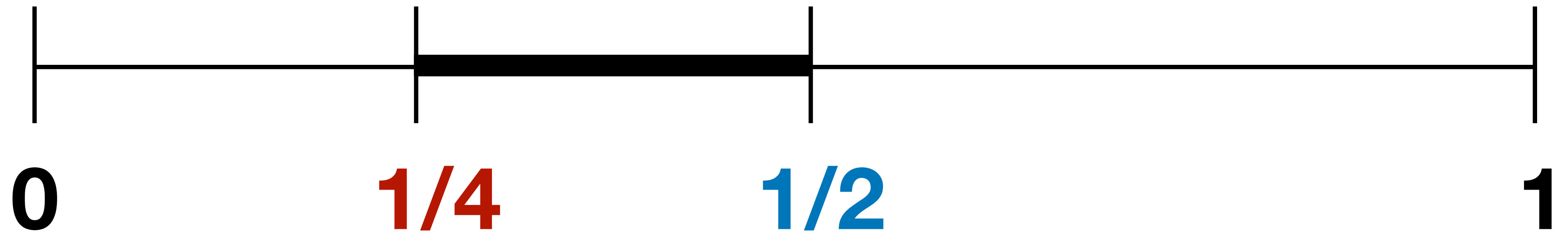
$\text{bias}($  $) = -1$

Total bias B can be estimated in $O(\log n)$ space!

- Can you see **why**?

Warm-up: 2/5-Approximation by [GVV17]

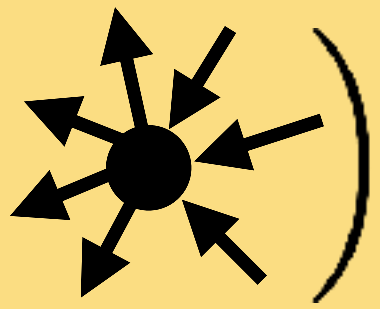
- **Recall**: Trivial algorithm gives **1/4**-approx. while **1/2**-approx. is hard.
- **Idea**: Consider the ***bias*** of each vertex.



Definition (bias and total bias):

- $\text{bias}(v) = \text{in-degree} - \text{out-degree}$
- Total bias: $B = \sum_v |\text{bias}(v)| \in [0, 2m]$

Example:

bias() $= -1$

Total bias B can be estimated in $O(\log n)$ space!

- Can you see **why**?
- Understand the relation between B and $\text{val}_{\mathcal{C}}$ could give approximation.

Relation Between Total Bias and Cut Value

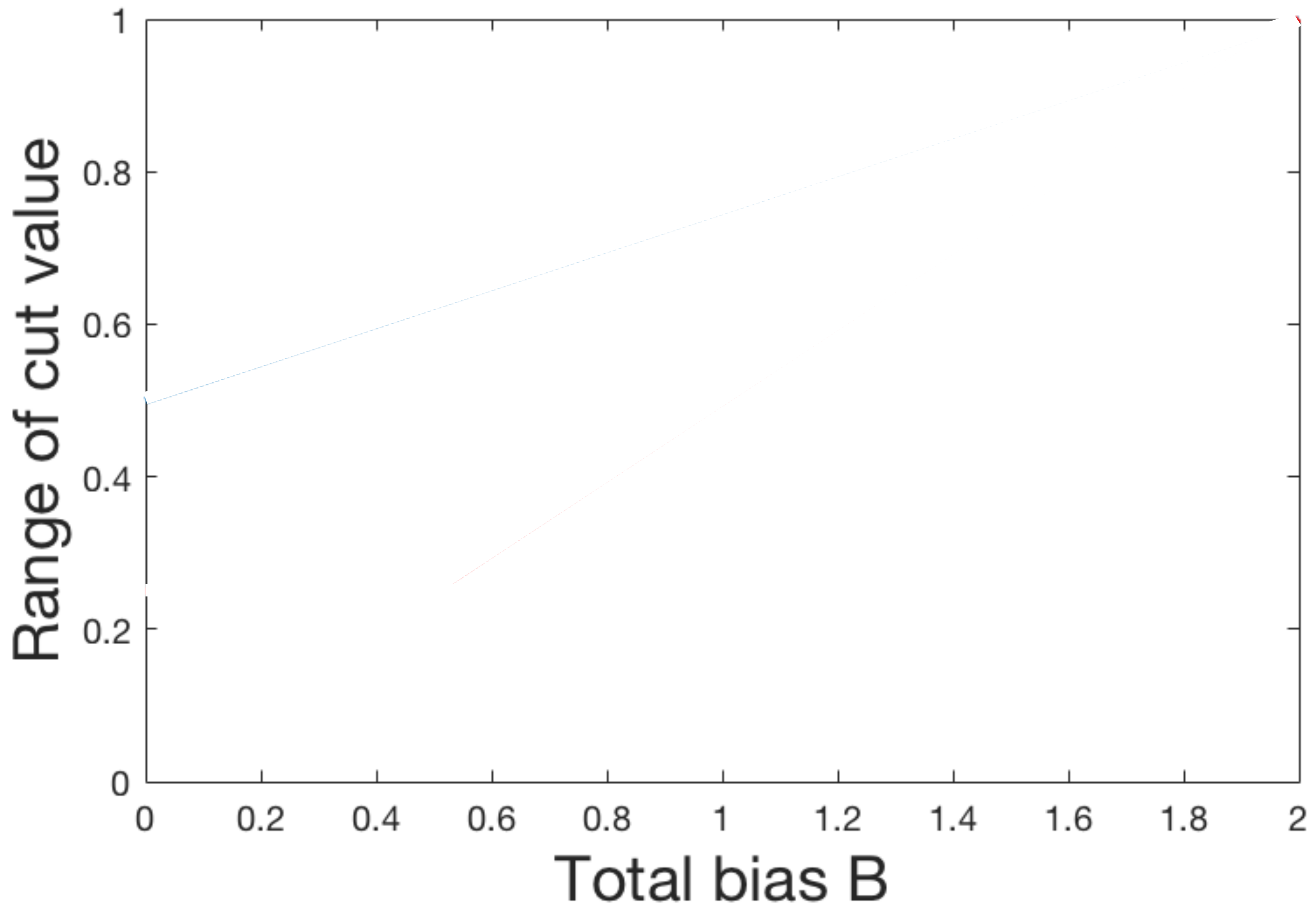
Definition (bias and total bias):

$$\mathsf{bias}(v) = \text{in-degree} - \text{out-degree} \quad \text{and} \quad B = \sum_v |\mathsf{bias}(v)|$$

Relation Between Total Bias and Cut Value

<u>**Definition** (bias and total bias):</u>

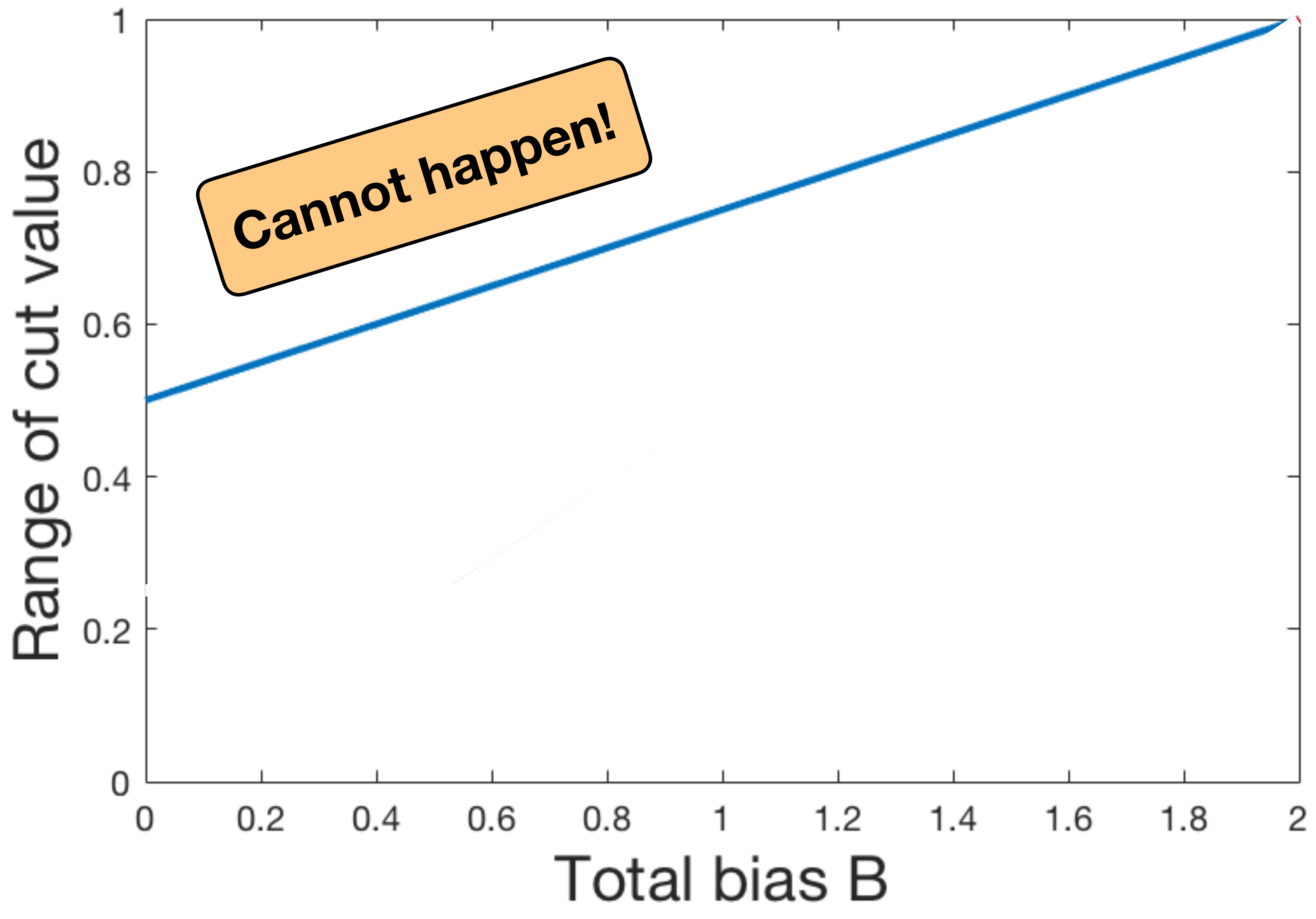
$$\text{bias}(v) = \text{in-degree} - \text{out-degree} \quad \text{and} \quad B = \sum_v |\text{bias}(v)|$$



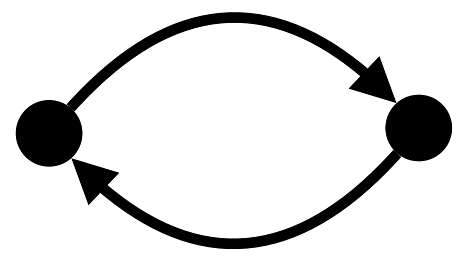
Relation Between Total Bias and Cut Value

Definition (bias and total bias):

$$\text{bias}(v) = \text{in-degree} - \text{out-degree} \quad \text{and} \quad B = \sum_v |\text{bias}(v)|$$



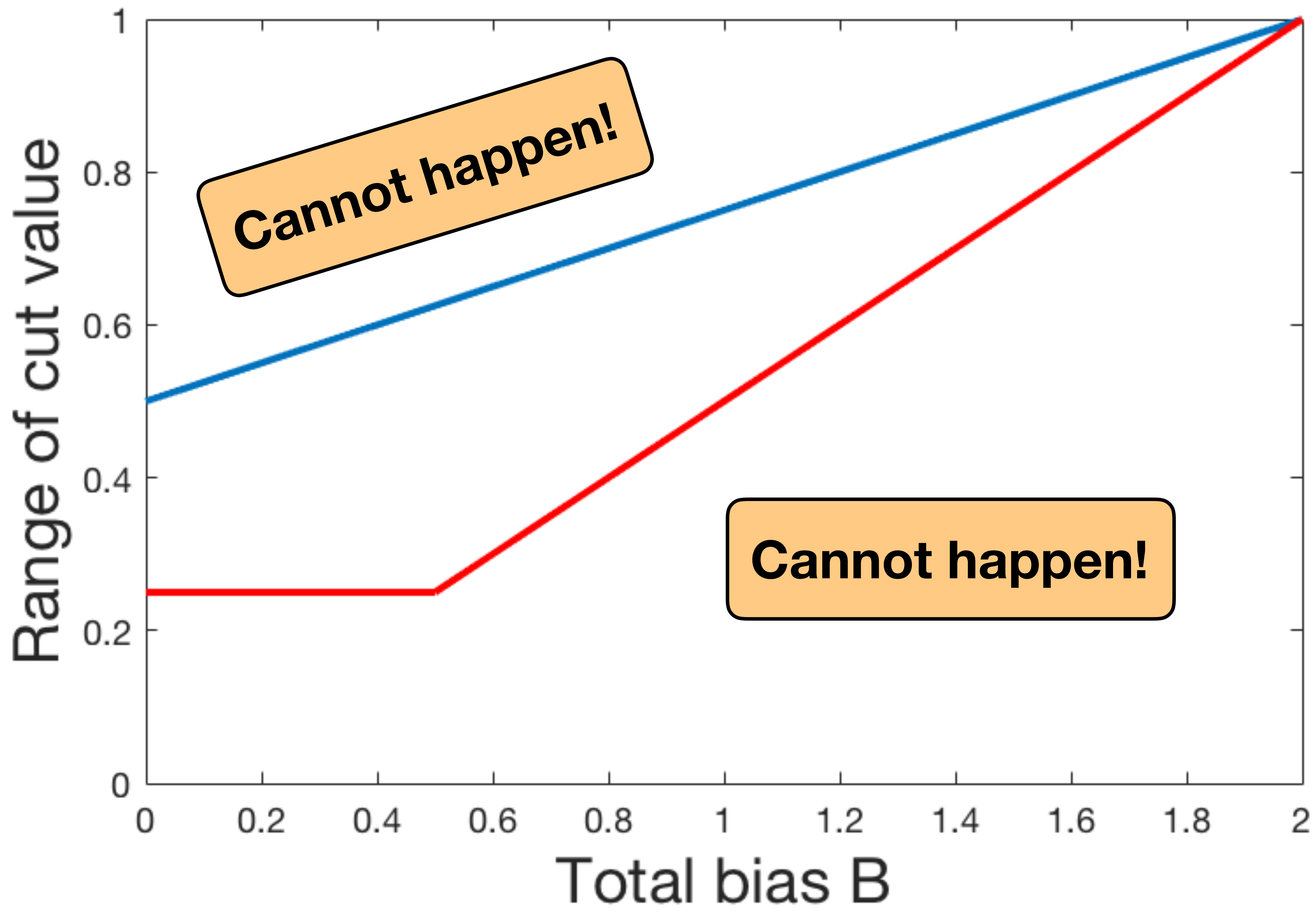
- **Blue line** (cut value upper bound):

  Small bias => ∃ 

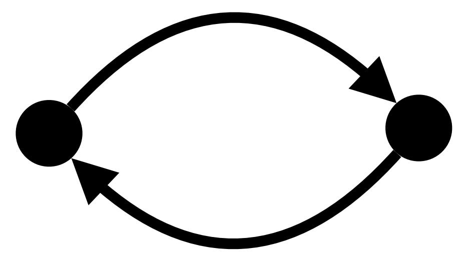
Relation Between Total Bias and Cut Value

Definition (bias and total bias):

$$\text{bias}(v) = \text{in-degree} - \text{out-degree} \quad \text{and} \quad B = \sum_v |\text{bias}(v)|$$



- **Blue line** (cut value upper bound):

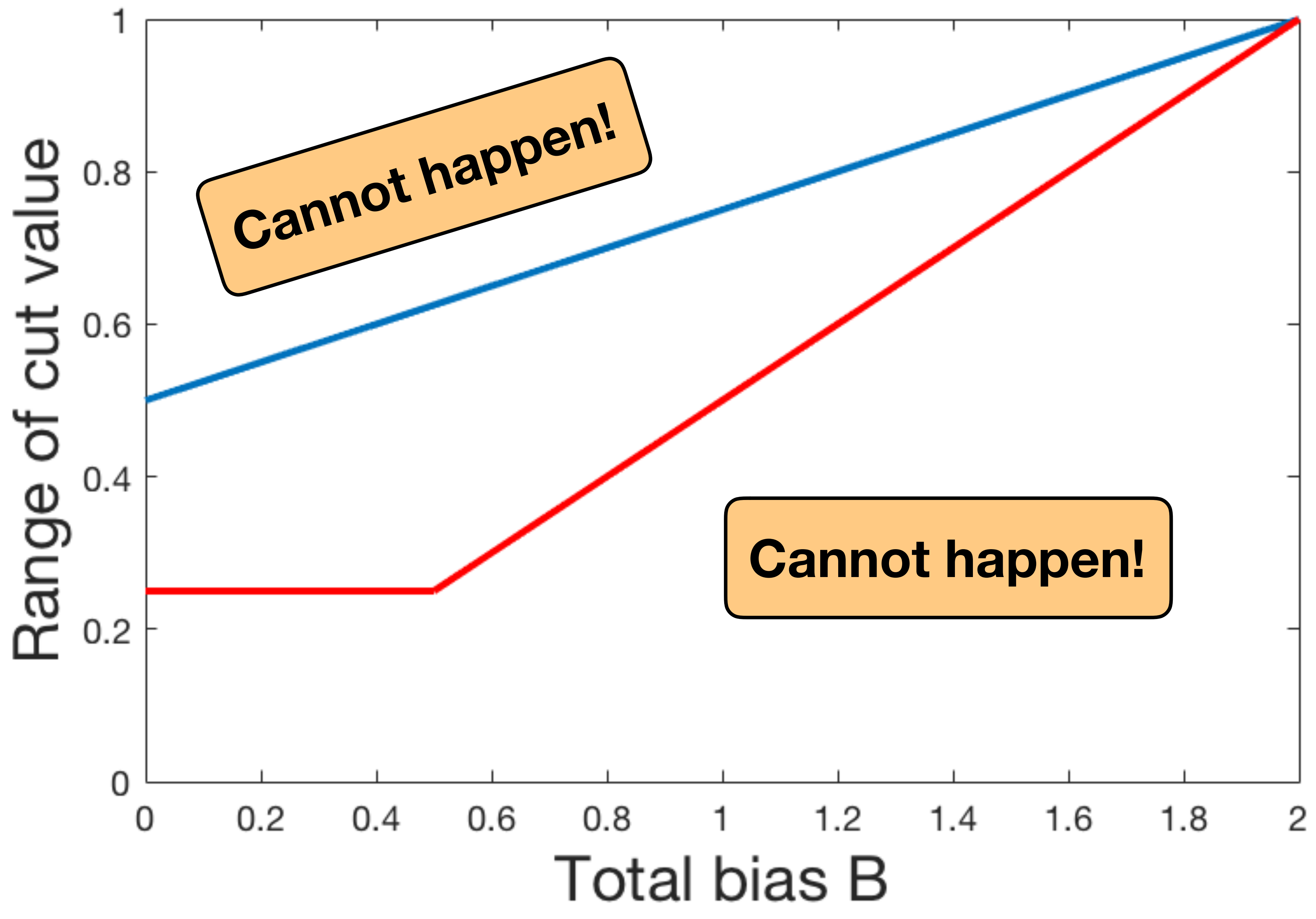
  Small bias => ∃ 

- **Red line**: The cut value of greedy cut.

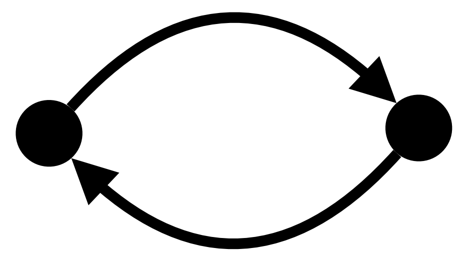
Relation Between Total Bias and Cut Value

Definition (bias and total bias):

$$\text{bias}(v) = \text{in-degree} - \text{out-degree} \quad \text{and} \quad B = \sum_v |\text{bias}(v)|$$

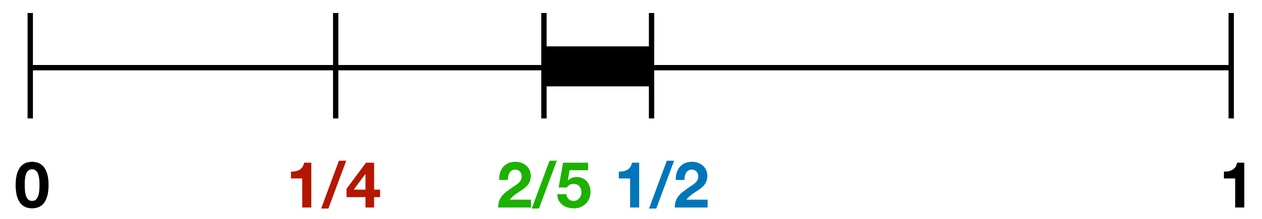


- **Blue line** (cut value upper bound):

  Small bias => $\exists$ 

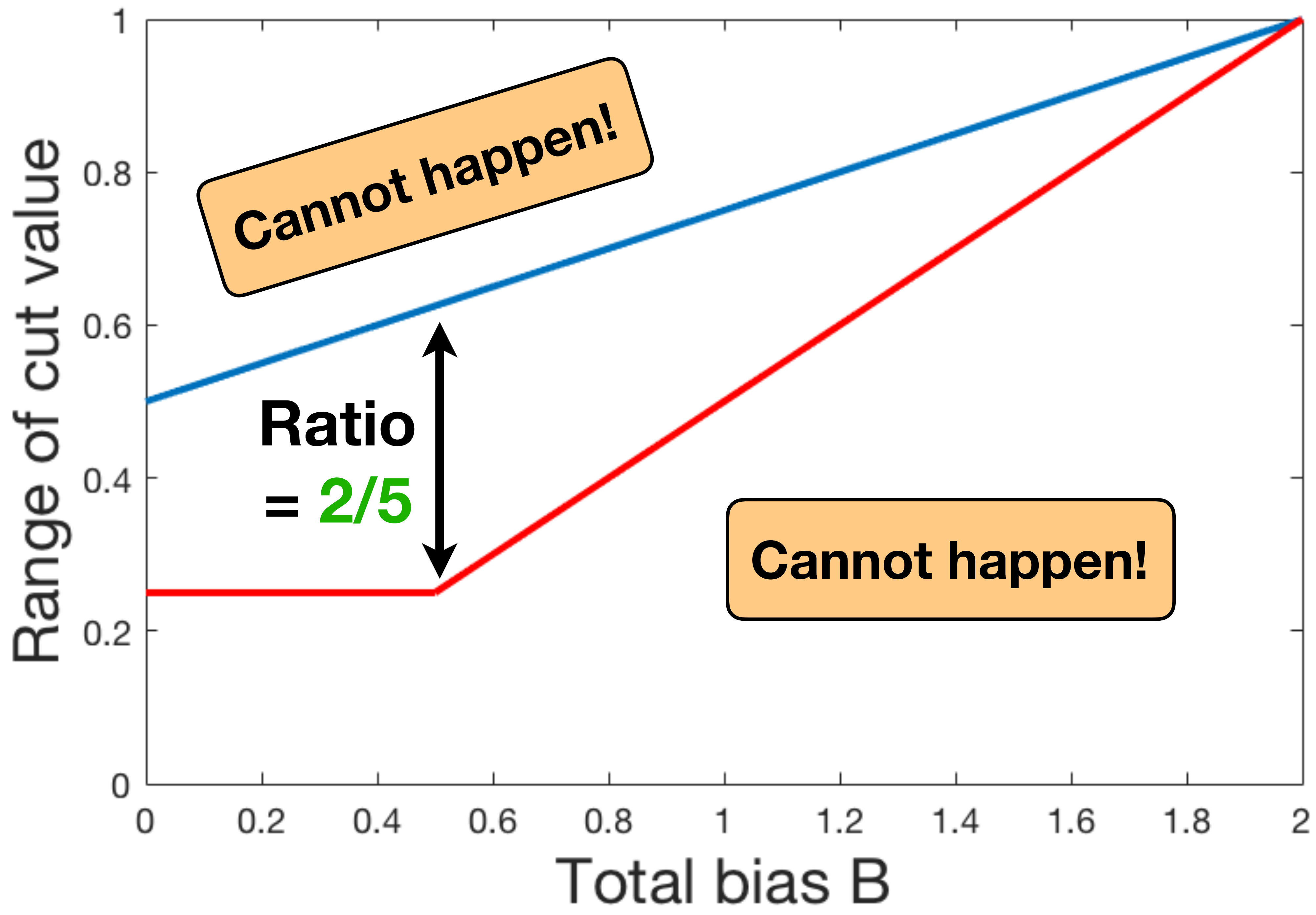
- **Red line**: The cut value of greedy cut.
- **Streaming algorithm**: Estimate *B* and output the **red line**.

Relation Between Total Bias and Cut Value

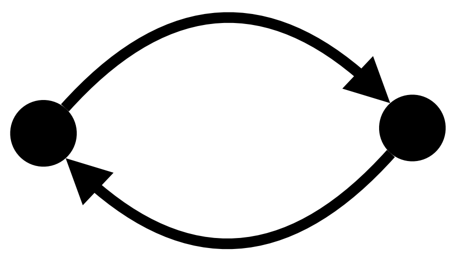


Definition (bias and total bias):

$$\text{bias}(v) = \text{in-degree} - \text{out-degree} \quad \text{and} \quad B = \sum_v |\text{bias}(v)|$$



- **Blue line** (cut value upper bound):

  Small bias => $\exists$ 

- **Red line**: The cut value of greedy cut.
- **Streaming algorithm**: Estimate *B* and output the **red line**.
- **Ratio**: When *B* = 1/2, the ratio is **2/5**.

New Idea: Random Sampling

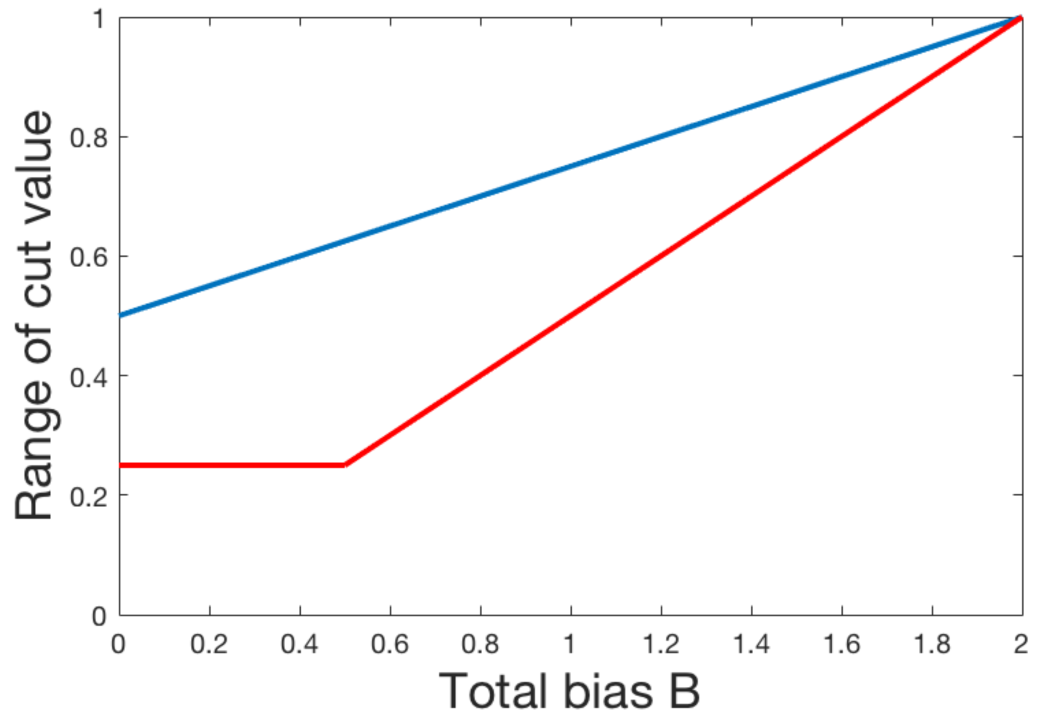
New Idea: Random Sampling with Bias

New Idea: Random Sampling with Bias

- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.

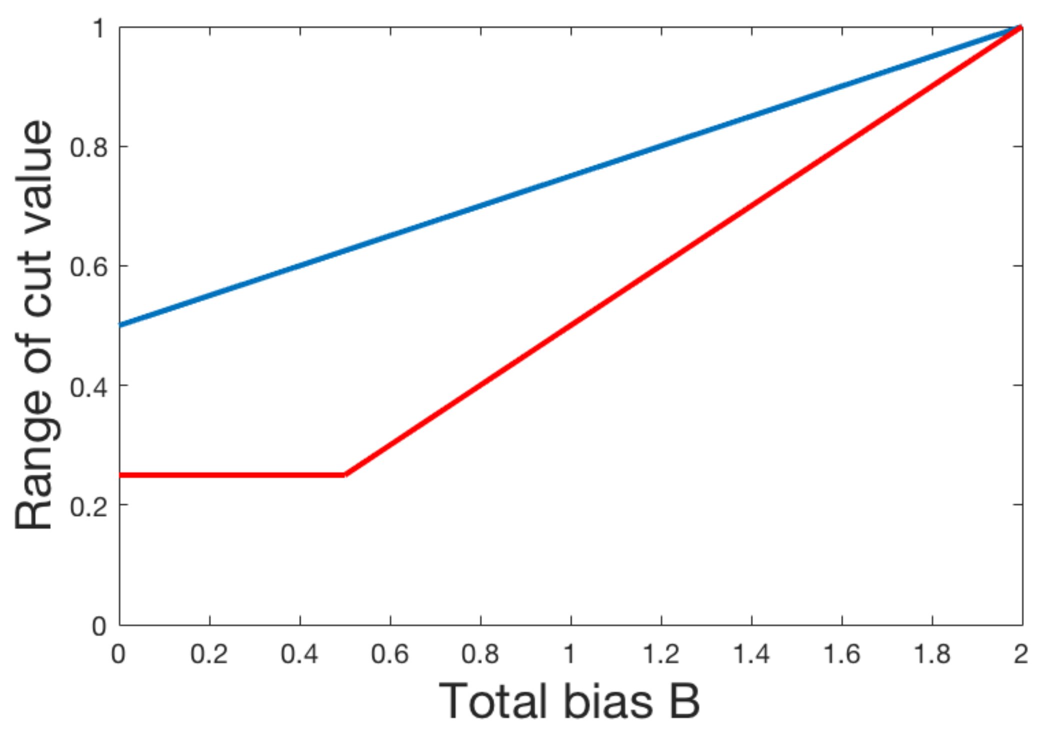
New Idea: Random Sampling with Bias

- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because



New Idea: Random Sampling with Bias

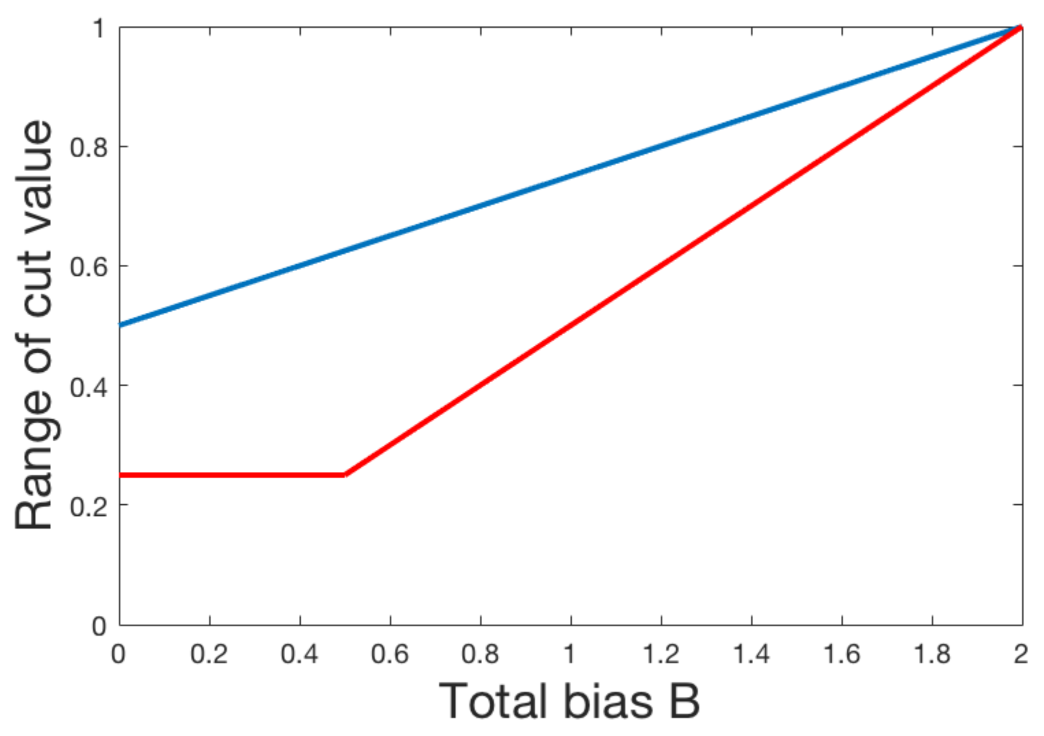
- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because



(i) there's a graph with optimal cut value **#edges** and

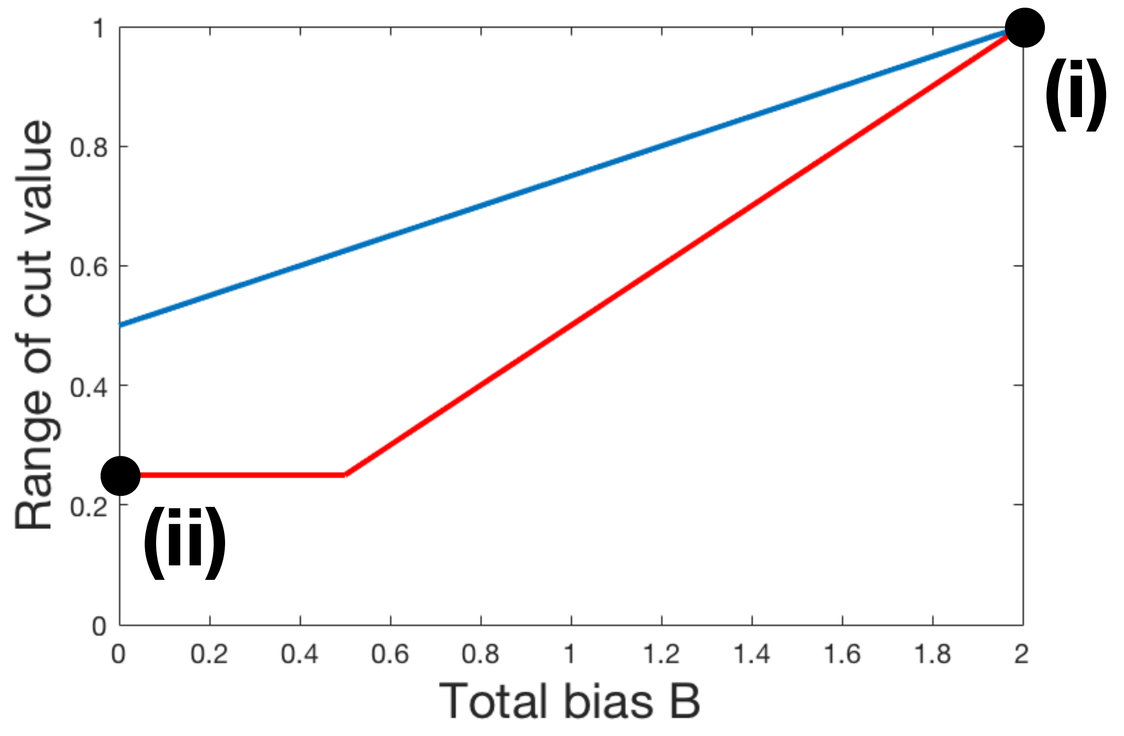
New Idea: Random Sampling with Bias

- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because
  - (i) there's a graph with optimal cut value **#edges** and
  - (ii) there's a graph with optimal cut value **#edges/4.**



New Idea: Random Sampling with Bias

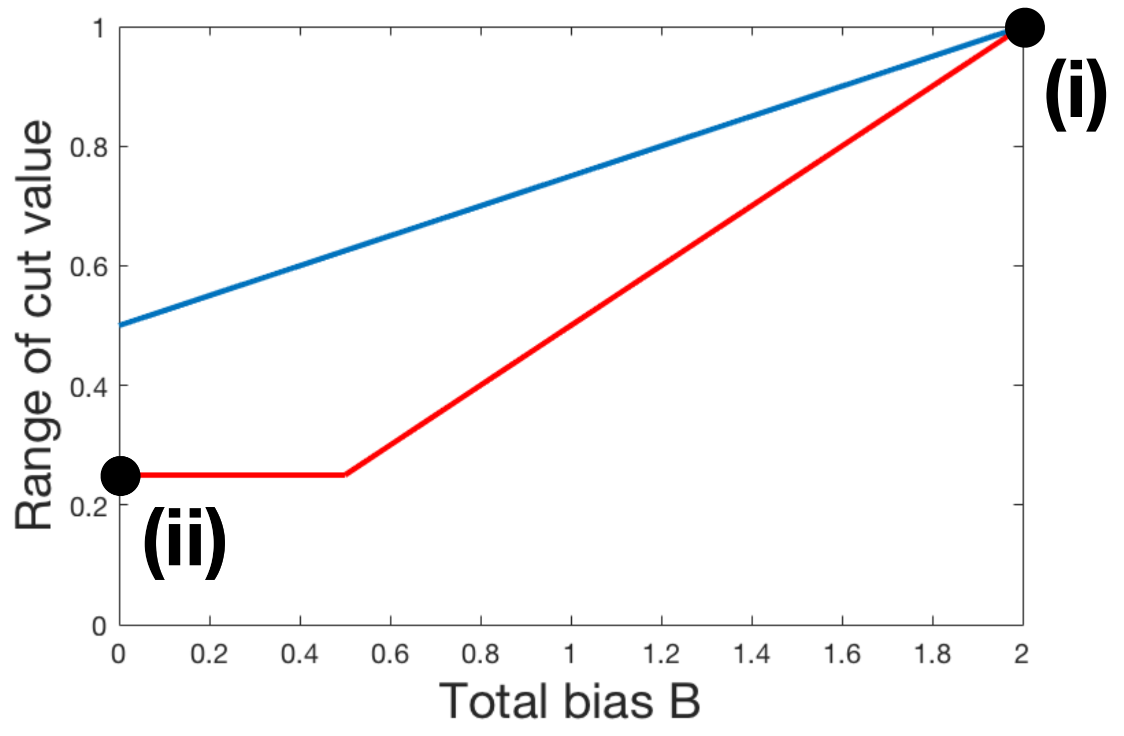
- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because
  - (i) there's a graph with optimal cut value **#edges** and
  - (ii) there's a graph with optimal cut value **#edges/4.**



- However, by [GVV17], the total bias *B* of (i) and (ii) are in different ranges!

New Idea: Random Sampling with Bias

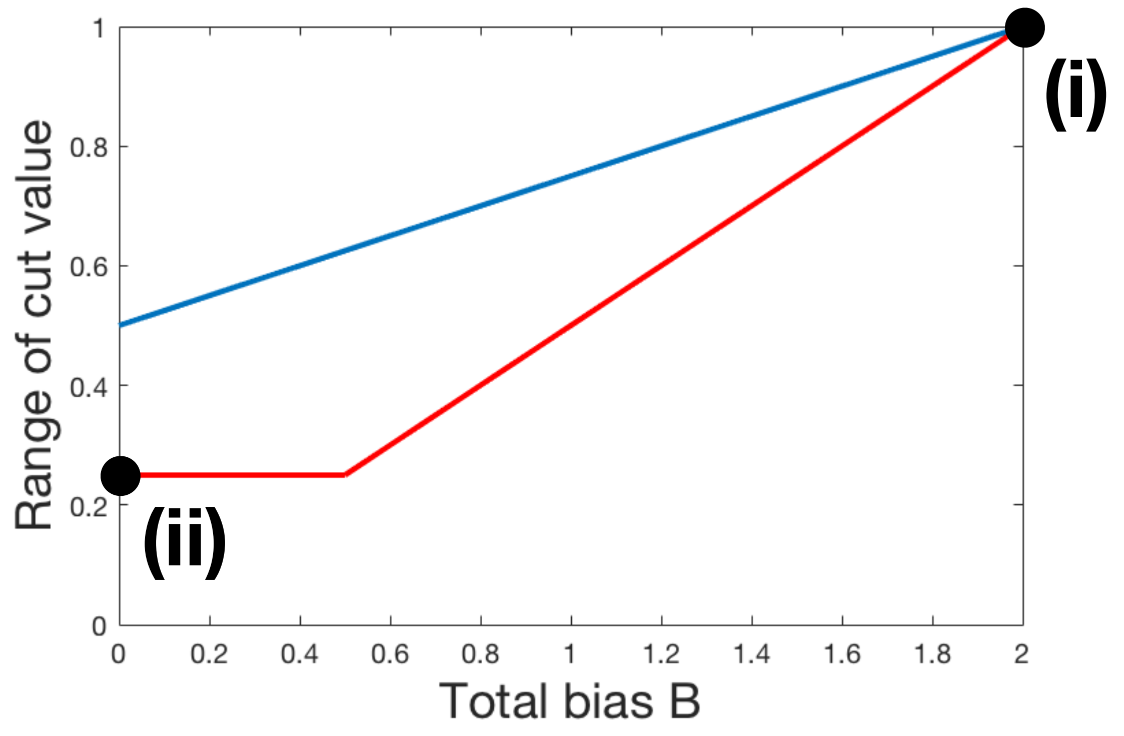
- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because
  - (i) there's a graph with optimal cut value **#edges** and
  - (ii) there's a graph with optimal cut value **#edges/4.**



- However, by [GVV17], the total bias B of (i) and (ii) are in different ranges!
- **Random sampling with bias**: Let δ be a const. chosen later. For each i,

New Idea: Random Sampling with Bias

- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because
  - (i) there's a graph with optimal cut value **#edges** and
  - (ii) there's a graph with optimal cut value **#edges/4.**

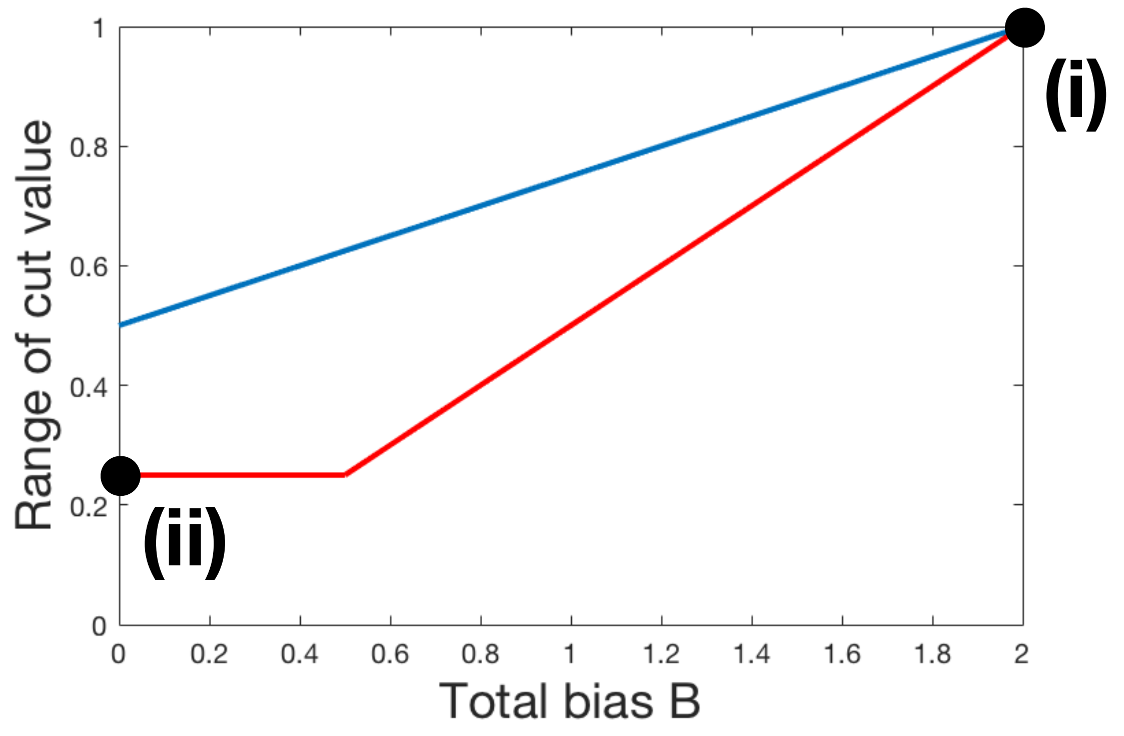


- However, by [GVV17], the total bias *B* of (i) and (ii) are in different ranges!
- **Random sampling with bias**: Let δ be a const. chosen later. For each i,

- i

New Idea: Random Sampling with Bias

- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because
  - (i) there's a graph with optimal cut value **#edges** and
  - (ii) there's a graph with optimal cut value **#edges/4.**

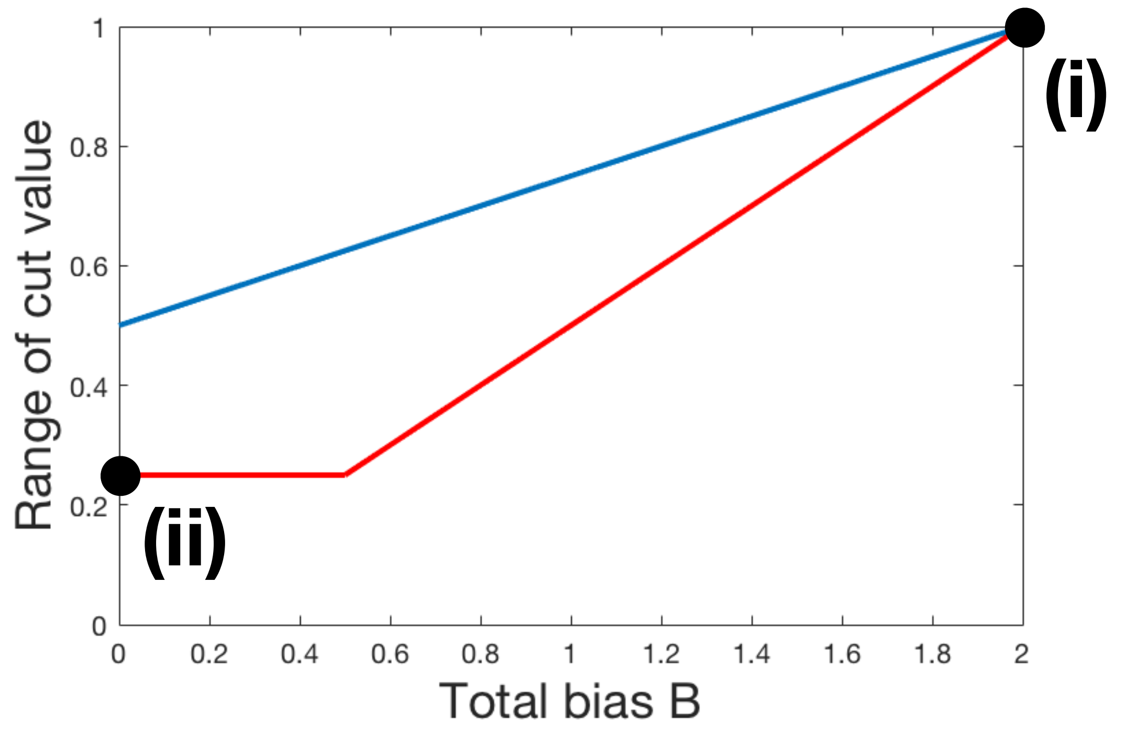


- However, by [GVV17], the total bias *B* of (i) and (ii) are in different ranges!
- **Random sampling with bias**: Let δ be a const. chosen later. For each i,

$\bullet_i$ → In-going set with probability $\frac{1}{2} + \delta \cdot \text{sgn}\left(\text{bias}\left(\bullet_i\right)\right)$

New Idea: Random Sampling with Bias

- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because
  - (i) there's a graph with optimal cut value **#edges** and
  - (ii) there's a graph with optimal cut value **#edges/4.**



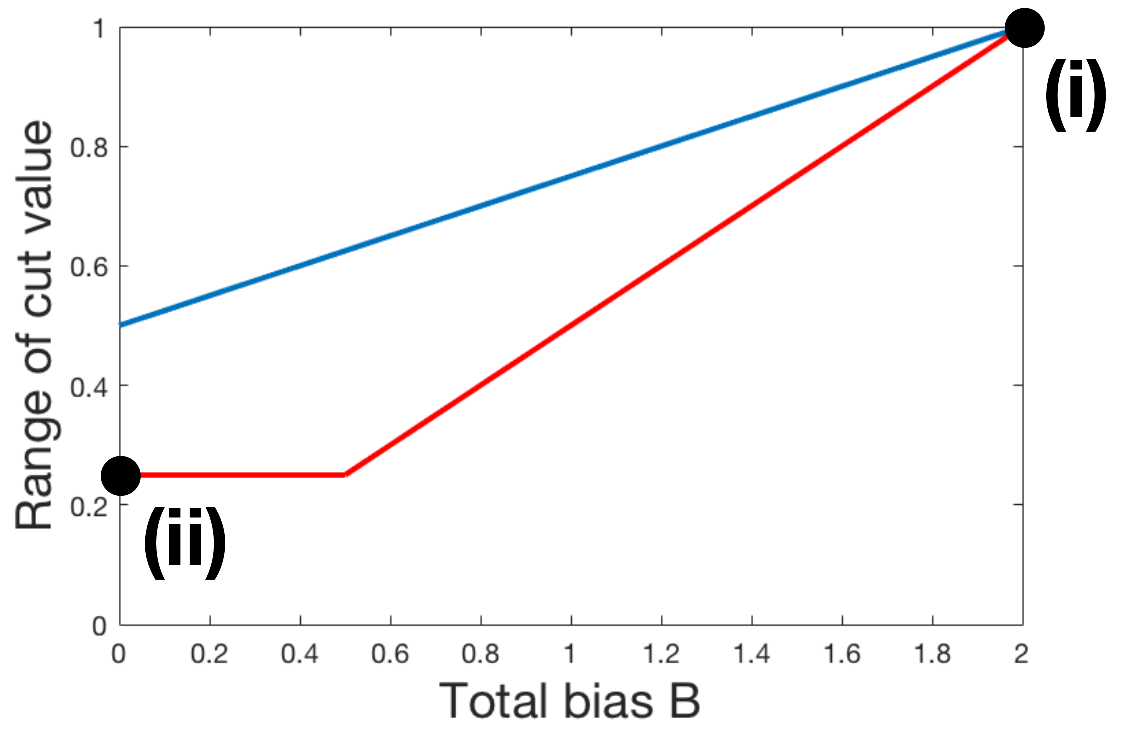
- However, by [GVV17], the total bias *B* of (i) and (ii) are in different ranges!
- **Random sampling with bias**: Let δ be a const. chosen later. For each i,

$\bullet_i$ → In-going set with probability $\frac{1}{2} + \delta \cdot \mathrm{sgn}\left(\mathrm{bias}\left(\bullet_i\right)\right)$

$\bullet_i$ → Out-going set with probability $\frac{1}{2} - \delta \cdot \mathrm{sgn}\left(\mathrm{bias}\left(\bullet_i\right)\right)$

New Idea: Random Sampling with Bias

- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because
  (i) there's a graph with optimal cut value **#edges** and
  (ii) there's a graph with optimal cut value **#edges/4.**



- However, by [GVV17], the total bias B of (i) and (ii) are in different ranges!
- **Random sampling with bias**: Let δ be a const. chosen later. For each i,

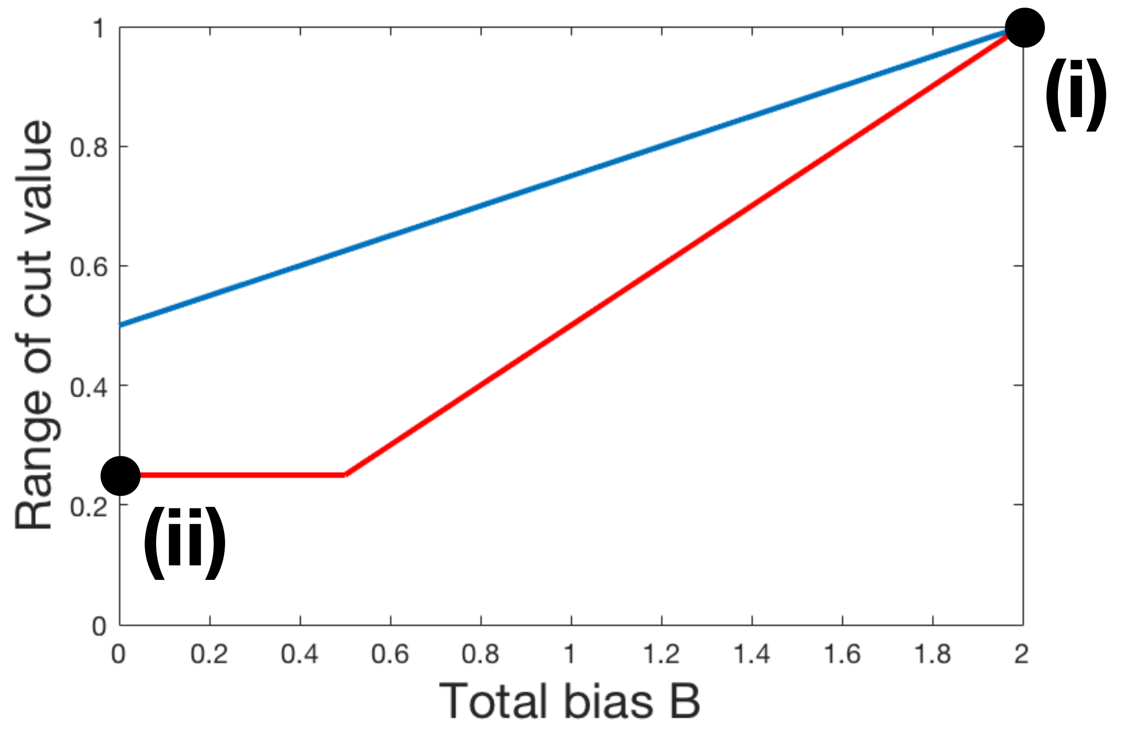
$\bullet_i$ → In-going set with probability $\frac{1}{2} + \delta \cdot \mathrm{sgn}\left(\mathrm{bias}\left(\bullet_i\right)\right)$

$\bullet_i$ → Out-going set with probability $\frac{1}{2} - \delta \cdot \mathrm{sgn}\left(\mathrm{bias}\left(\bullet_i\right)\right)$

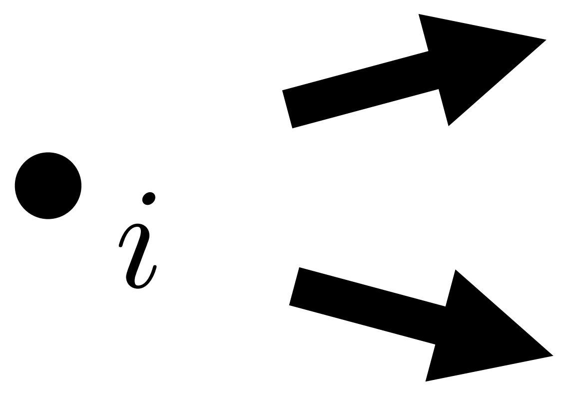
> $\delta = 0$ recovers trivial random sampling.

New Idea: Random Sampling with Bias

- **Recall**: Trivial random sampling gives **1/4**-approx. by outputting #edges/4.
- **Observation**: This is tight for random sampling because
  (i) there's a graph with optimal cut value **#edges** and
  (ii) there's a graph with optimal cut value **#edges/4.**



- However, by [GVV17], the total bias *B* of (i) and (ii) are in different ranges!
- **Random sampling with bias**: Let δ be a const. chosen later. For each i,



In-going set with probability $\frac{1}{2} + \delta \cdot \mathrm{sgn}\Big(\mathrm{bias}\big(\bullet_i\big)\Big)$

Out-going set with probability $\frac{1}{2} - \delta \cdot \mathrm{sgn}\Big(\mathrm{bias}\big(\bullet_i\big)\Big)$

$\delta = 0$ recovers trivial random sampling.
$\delta = 1/2$ gets [GVV17].

New Relation Between Total Bias and Cut Value

New Relation Between Total Bias and Cut Value

- By optimizing the choice of δ and analyzing the expected cut value, we have

New Relation Between Total Bias and Cut Value

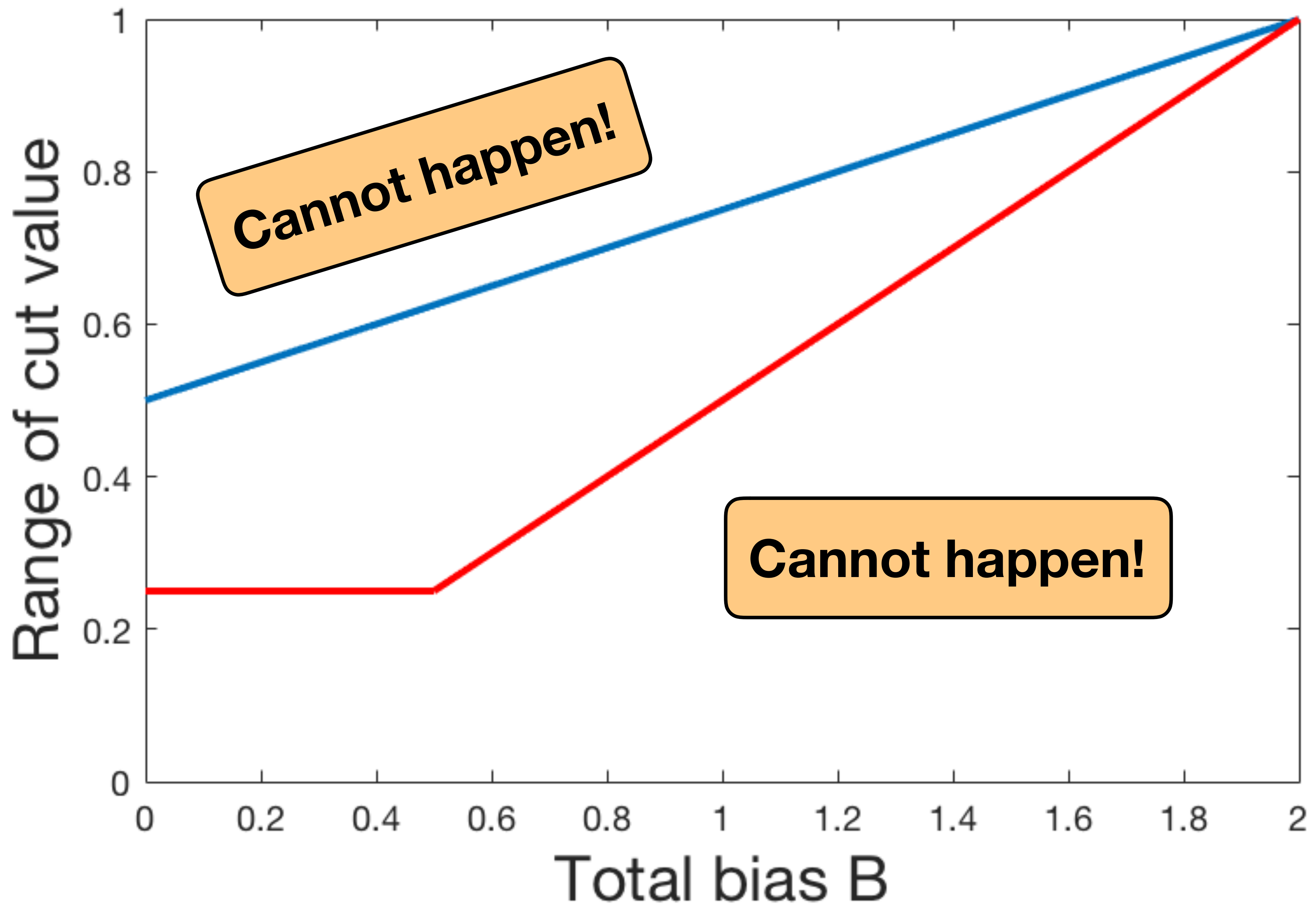
- By optimizing the choice of δ and analyzing the expected cut value, we have

$$\mathsf{val}_{\mathcal{C}} \geq \frac{\#\ \text{edges}}{4} + \frac{B^2}{16(\#\ \text{edges} - B)} \qquad \text{when} \qquad B \in \left[0, \frac{2}{3}\right]$$

New Relation Between Total Bias and Cut Value

- By optimizing the choice of δ and analyzing the expected cut value, we have

$$\text{val}_{\mathcal{C}} \geq \frac{\#\ \text{edges}}{4} + \frac{B^2}{16(\#\ \text{edges} - B)} \qquad \text{when} \qquad B \in \left[0, \frac{2}{3}\right]$$

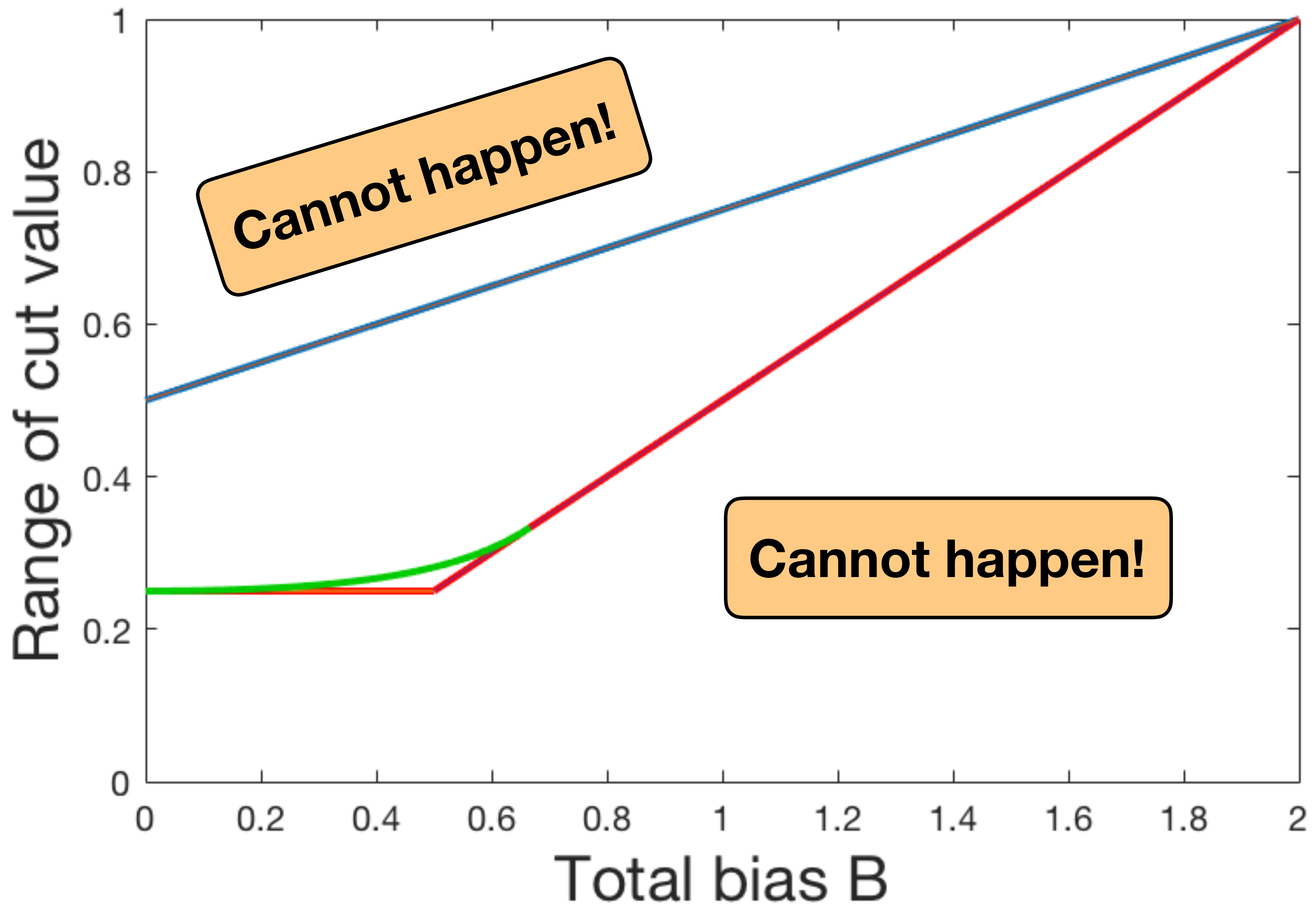


- **Blue line**: cut value upper bound.
- **Red line**: The cut value of greedy cut.

New Relation Between Total Bias and Cut Value

- By optimizing the choice of δ and analyzing the expected cut value, we have

$$\text{val}_{\mathcal{C}} \geq \frac{\#\ \text{edges}}{4} + \frac{B^2}{16(\#\ \text{edges} - B)} \quad \text{when} \quad B \in \left[0, \frac{2}{3}\right]$$

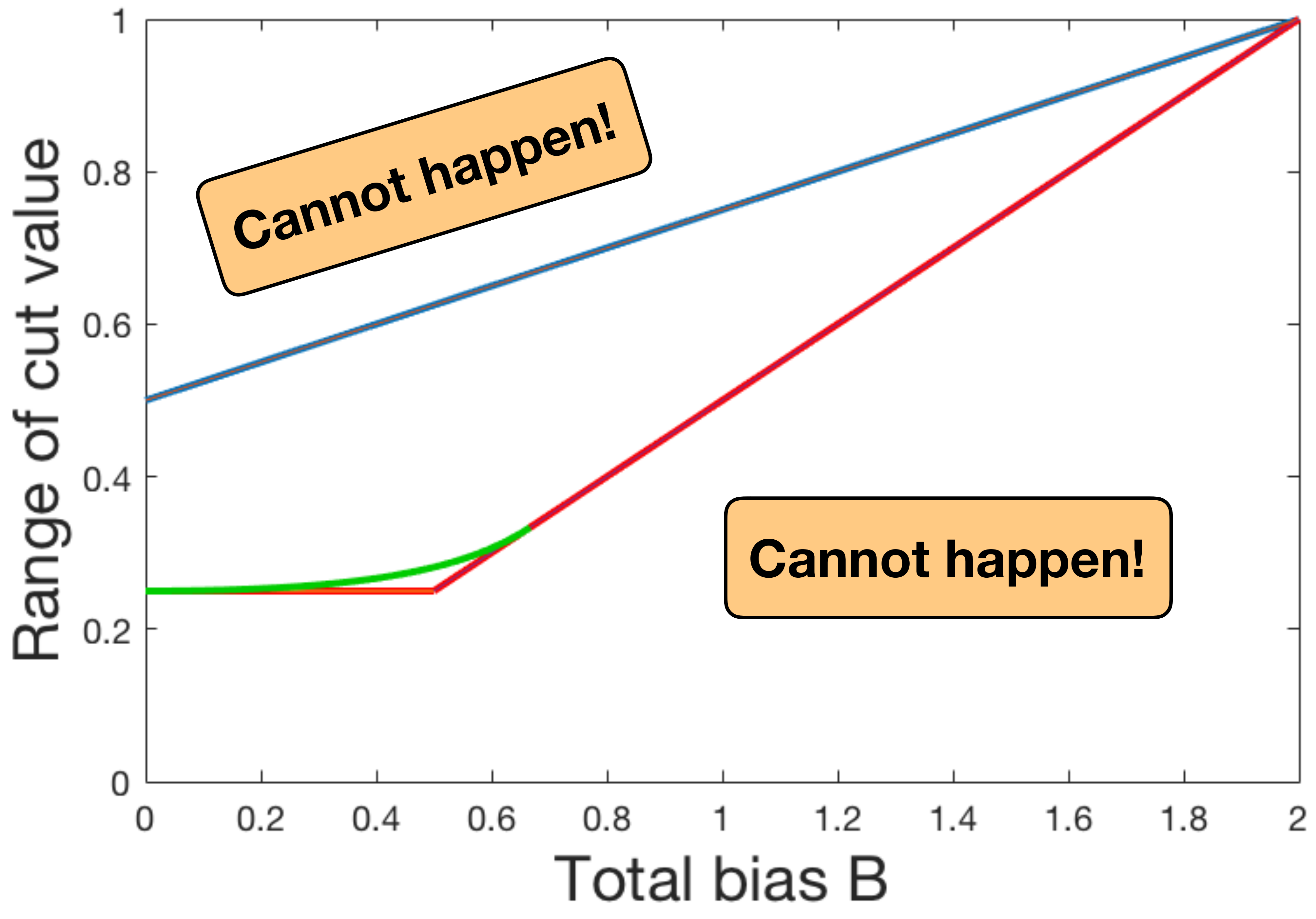


- **Blue line**: cut value upper bound.
- **Red line**: The cut value of greedy cut.
- **Green line**: Cut value achieved by random sampling with bias.

New Relation Between Total Bias and Cut Value

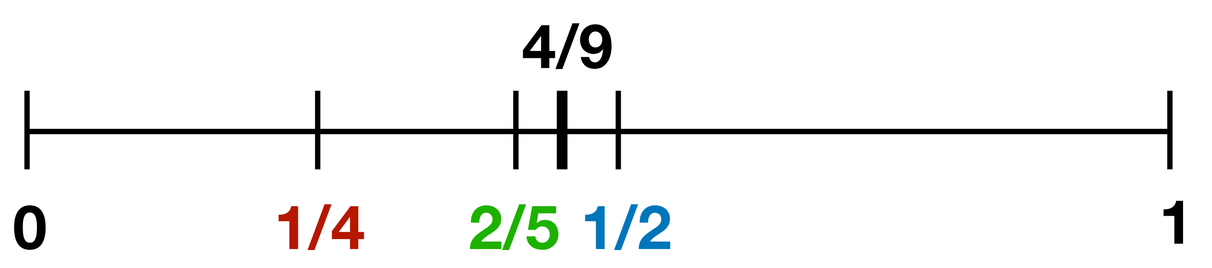
- By optimizing the choice of δ and analyzing the expected cut value, we have

$$\text{val}_{\mathcal{C}} \geq \frac{\#\text{ edges}}{4} + \frac{B^2}{16(\#\text{ edges} - B)} \quad \text{when} \quad B \in \left[0, \frac{2}{3}\right]$$



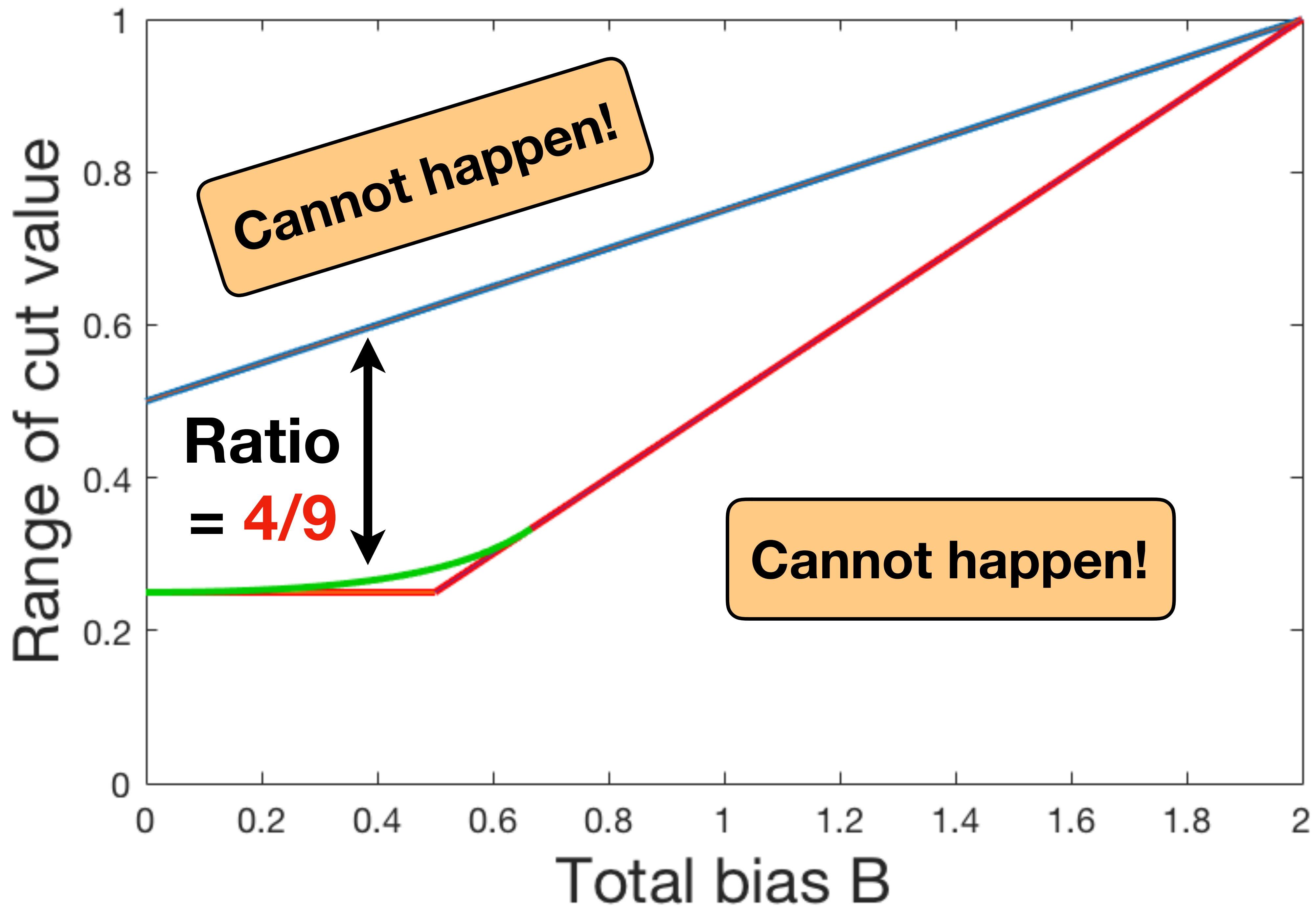
- **Blue line**: cut value upper bound.
- **Red line**: The cut value of greedy cut.
- **Green line**: Cut value achieved by random sampling with bias.
- **Streaming algorithm**: Estimate *B* and output max {**green line**, **red line**}.

New Relation Between Total Bias and Cut Value



- By optimizing the choice of δ and analyzing the expected cut value, we have

$$\text{val}_{\mathcal{C}} \geq \frac{\#\ \text{edges}}{4} + \frac{B^2}{16(\#\ \text{edges} - B)} \quad \text{when} \quad B \in \left[0, \frac{2}{3}\right]$$



- **Blue line**: cut value upper bound.
- **Red line**: The cut value of greedy cut.
- **Green line**: Cut value achieved by random sampling with bias.
- **Streaming algorithm**: Estimate B and output max {**green line**, **red line**}.
- **Ratio**: When B = 2/5, the ratio is **4/9**.

Local Random Sampling is Optimal in Streaming Model

Local Random Sampling is Optimal in Streaming Model

- It turns out that the optimal approx. ratio of all the boolean 2CSP can be achieved by local random sampling analysis.

Local Random Sampling is Optimal in Streaming Model

- It turns out that the optimal approx. ratio of all the boolean 2CSP can be achieved by local random sampling analysis.

Λ	α_{Λ}	Previous	Reference

Local Random Sampling is Optimal in Streaming Model

- It turns out that the optimal approx. ratio of all the boolean 2CSP can be achieved by local random sampling analysis.

Λ	α_Λ	Previous	Reference
2XOR	$\frac{1}{2}$	$\frac{1}{2}$	Trivial

Local Random Sampling is Optimal in Streaming Model

- It turns out that the optimal approx. ratio of all the boolean 2CSP can be achieved by local random sampling analysis.

Λ	α_Λ	Previous	Reference
2XOR	$\frac{1}{2}$	$\frac{1}{2}$	Trivial
2EOR	$\frac{3}{4}$	$\left[\frac{3}{4}, 1\right]$	Trivial

Local Random Sampling is Optimal in Streaming Model

- It turns out that the optimal approx. ratio of all the boolean 2CSP can be achieved by local random sampling analysis.

Λ	α_Λ	Previous	Reference
2XOR	$\frac{1}{2}$	$\frac{1}{2}$	Trivial
2EOR	$\frac{3}{4}$	$\left[\frac{3}{4}, 1\right]$	Trivial
2AND	$\frac{4}{9}$	$\left[\frac{2}{5}, \frac{1}{2}\right]$	Biased sampling

Local Random Sampling is Optimal in Streaming Model

- It turns out that the optimal approx. ratio of all the boolean 2CSP can be achieved by local random sampling analysis.

Λ	α_Λ	Previous	Reference
2XOR	$\frac{1}{2}$	$\frac{1}{2}$	Trivial
2EOR	$\frac{3}{4}$	$\left[\frac{3}{4}, 1\right]$	Trivial
2AND	$\frac{4}{9}$	$\left[\frac{2}{5}, \frac{1}{2}\right]$	Biased sampling
2OR	$\frac{\sqrt{2}}{2}$	$\left[\frac{1}{2}, 1\right]$	Biased sampling

Local Random Sampling is Optimal in Streaming Model

- It turns out that the optimal approx. ratio of all the boolean 2CSP can be achieved by local random sampling analysis.

Λ	α_Λ	Previous	Reference
2XOR	$\frac{1}{2}$	$\frac{1}{2}$	Trivial
2EOR	$\frac{3}{4}$	$\left[\frac{3}{4}, 1\right]$	Trivial
2AND	$\frac{4}{9}$	$\left[\frac{2}{5}, \frac{1}{2}\right]$	Biased sampling
2OR	$\frac{\sqrt{2}}{2}$	$\left[\frac{1}{2}, 1\right]$	Biased sampling

- See our paper for more details!

Hardness

Hardness

Find Instances Matching
Random Sampling's Bounds

Streaming Lower Bounds via Communication Complexity

Streaming Lower Bounds via Communication Complexity

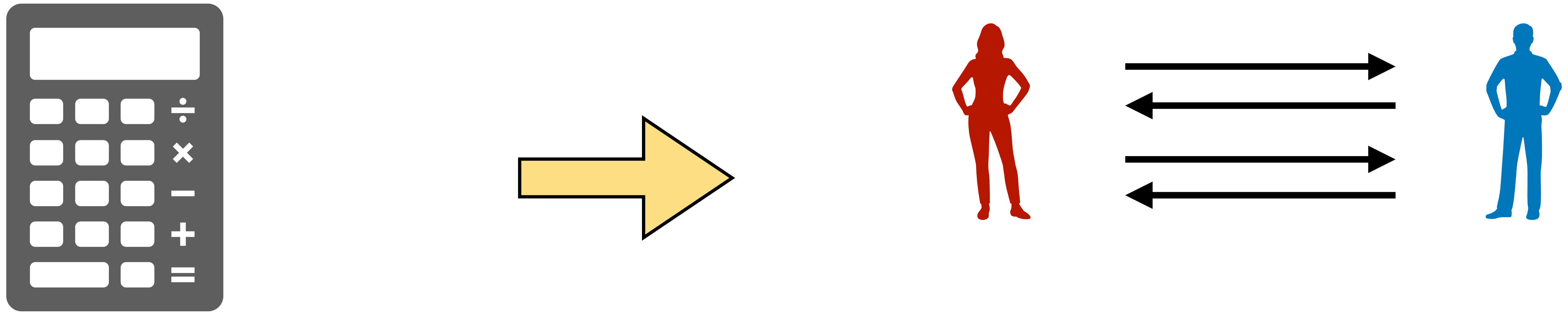
- *Unconditional lower bounds* from **communication games**.

Streaming Lower Bounds via Communication Complexity

- *Unconditional lower bounds* from **communication games**.
- **High-level idea**:

Streaming Lower Bounds via Communication Complexity

- *Unconditional lower bounds* from **communication games**.
- **High-level idea**:

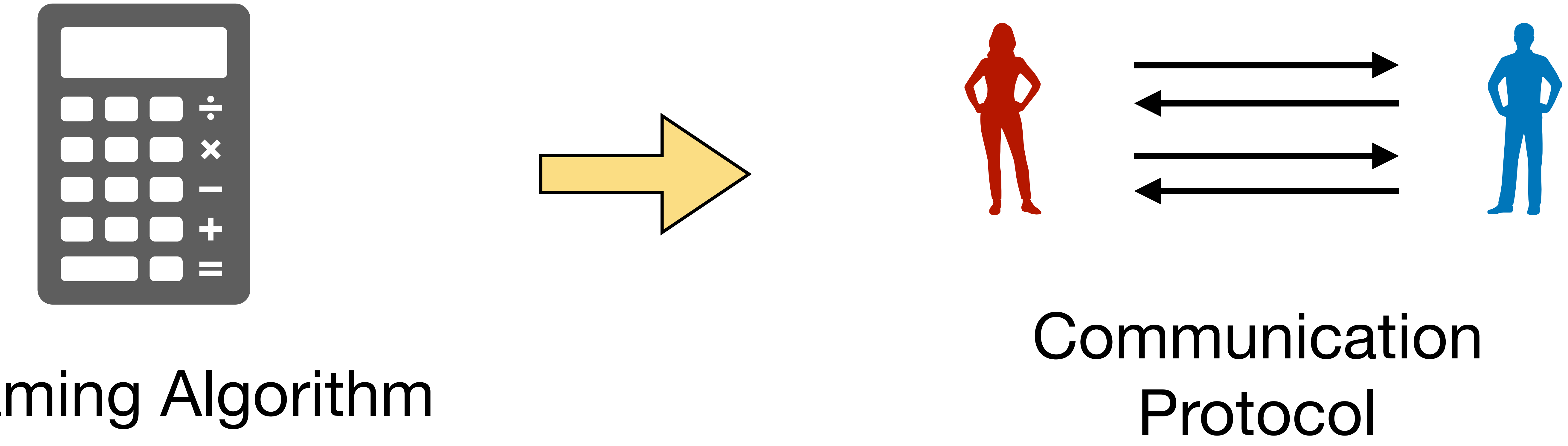


Streaming Algorithm

Communication Protocol

Streaming Lower Bounds via Communication Complexity

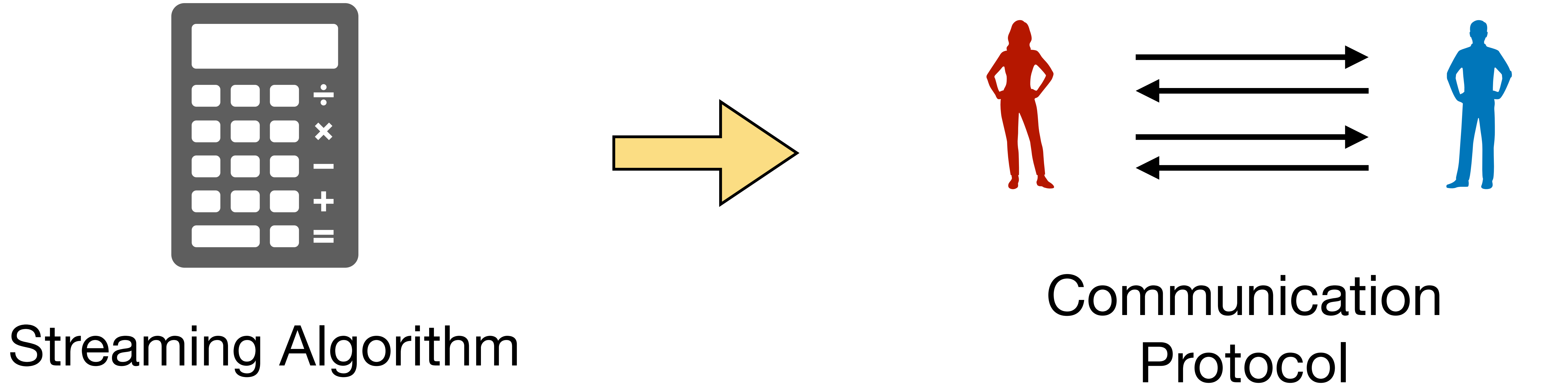
- *Unconditional lower bounds* from **communication games**.
- **High-level idea**:



- **Usage**: Alice and Bob insert some inputs to the streaming algorithm and send the "***configuration***" as the message.

Streaming Lower Bounds via Communication Complexity

- *Unconditional lower bounds* from **communication games**.
- **High-level idea**:



- **Usage**: Alice and Bob insert some inputs to the streaming algorithm and send the “***configuration***” as the message.
- Space complexity of streaming algorithm **>=** communication complexity.

Distributional Boolean Hidden Partition (DBHP) Problem

Distributional Boolean Hidden Partition (DBHP) Problem

- Used by [Kapralov-Khanna-Sudan-15] in proving hardness of Max-Cut.

Distributional Boolean Hidden Partition (DBHP) Problem

- Used by [Kapralov-Khanna-Sudan-15] in proving hardness of Max-Cut.

Alice

$$X^* \in \{0,1\}^n$$

Distributional Boolean Hidden Partition (DBHP) Problem

* Each row of M contains exactly two 1s.

- Used by [Kapralov-Khanna-Sudan-15] in proving hardness of Max-Cut.

Alice

$X^* \in \{0,1\}^n$

c bits →

Bob

$w \in \{0,1\}^{0.01n}$

$M \in \{0,1\}^{0.01n \times n}$

Distributional Boolean Hidden Partition (DBHP) Problem

* Each row of M contains exactly two 1s.

- Used by [Kapralov-Khanna-Sudan-15] in proving hardness of Max-Cut.

Alice

$$X^* \in \{0,1\}^n$$

c bits →

Bob

$$w \in \{0,1\}^{0.01n}$$

$$M \in \{0,1\}^{0.01n \times n}$$

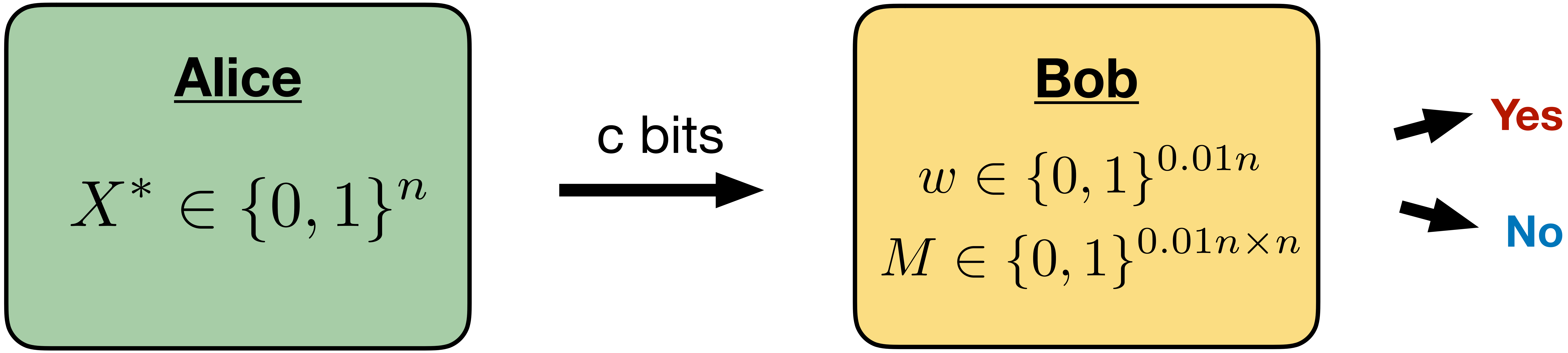
→ Yes

→ No

Distributional Boolean Hidden Partition (DBHP) Problem

* Each row of M contains exactly two 1s.

- Used by [Kapralov-Khanna-Sudan-15] in proving hardness of Max-Cut.

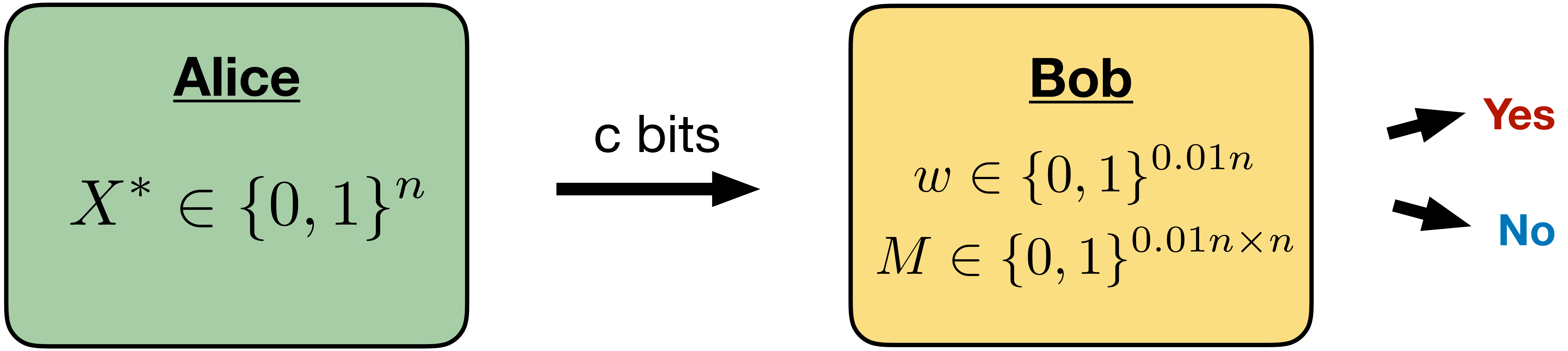


- **Yes distribution**: Exists $X^* \in \{0,1\}^n$ such that $w_t = M_t X^*, \ \forall t \in [T]$.

Distributional Boolean Hidden Partition (DBHP) Problem

* Each row of M contains exactly two 1s.

- Used by [Kapralov-Khanna-Sudan-15] in proving hardness of Max-Cut.

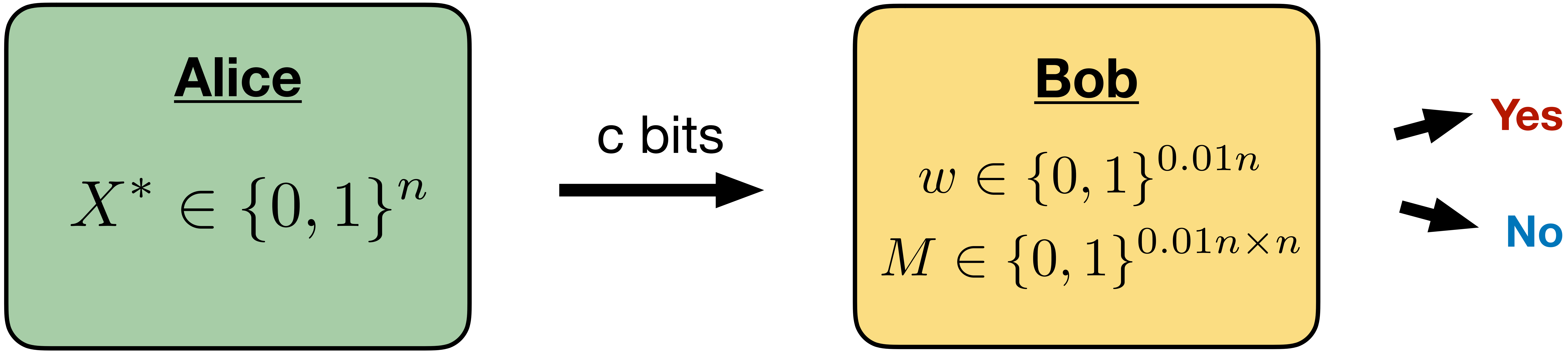


- **Yes distribution**: Exists $X^* \in \{0,1\}^n$ such that $w_t = M_t X^*, \ \forall t \in [T]$.
- **No distribution**: w_t is uniformly random $\forall t \in [T]$.

Distributional Boolean Hidden Partition (DBHP) Problem

* Each row of M contains exactly two 1s.

- Used by [Kapralov-Khanna-Sudan-15] in proving hardness of Max-Cut.

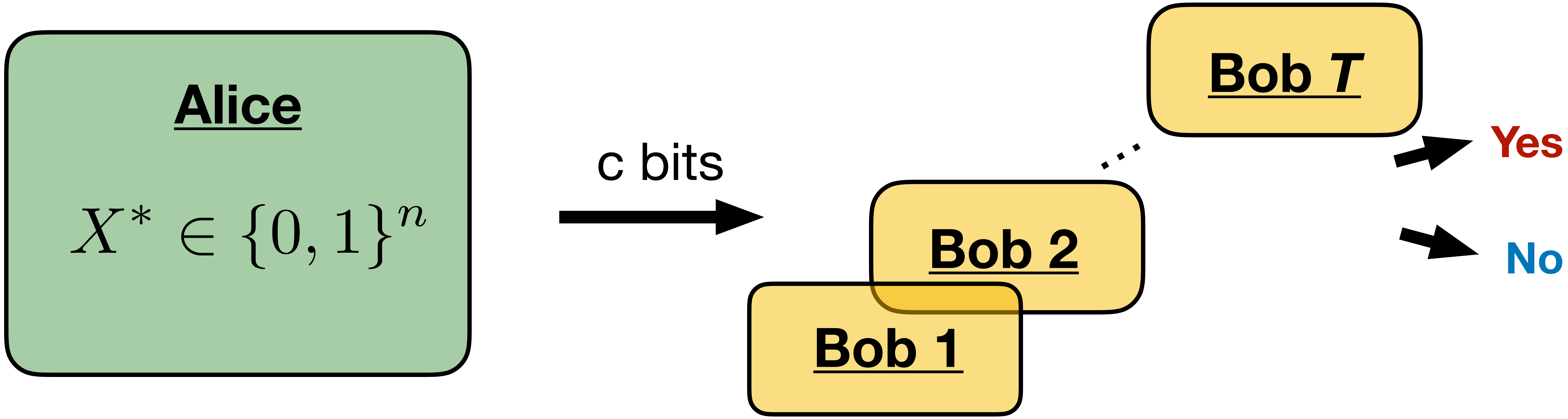


- **Yes distribution**: Exists $X^* \in \{0,1\}^n$ such that $w_t = M_t X^*, \ \forall t \in [T]$.
- **No distribution**: w_t is uniformly random $\forall t \in [T]$.
- [Gavinsky et al. 07] showed that DBHP needs $\Omega(\sqrt{n})$ communication.

Distributional Boolean Hidden Partition (DBHP) Problem

* Each row of M contains exactly two 1s.

- Used by [Kapralov-Khanna-Sudan-15] in proving hardness of Max-Cut.



- **Yes distribution**: Exists $X^* \in \{0,1\}^n$ such that $w_t = M_t X^*, \ \forall t \in [T]$.
- **No distribution**: w_t is uniformly random $\forall t \in [T]$.
- [Gavinsky et al. 07] showed that DBHP needs $\Omega(\sqrt{n})$ communication.
- **Parallel repetition:** constant many copies to increase the number of edges.

Example of DBHP (with Parallel Repetition)

Example of DBHP (with Parallel Repetition)

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

$$w_2 = [1\ 0\ 1]^\top$$

$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Example of DBHP (with Parallel Repetition)

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

$$w_2 = [1\ 0\ 1]^\top$$

$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

- Can you see this is a **Yes** case or **No** case?

- **Yes distribution**: Exists $X^* \in \{0,1\}^n$ such that $w_t = M_t X^*,\ \forall t \in [T]$.
- **No distribution**: w_t is uniformly random $\forall t \in [T]$.

Example of DBHP (with Parallel Repetition)

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

$$w_2 = [1\ 0\ 1]^\top$$

$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

- Can you see this is a **Yes** case or **No** case?
- The answer is **Yes**. The hidden partition is $X^* = [0\ 0\ 1\ 0\ 1]^\top$.

Example of DBHP (with Parallel Repetition)

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

$$w_2 = [1\ 0\ 1]^\top$$

$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

- Can you see this is a **Yes** case or **No** case?
- The answer is **Yes**. The hidden partition is $X^* = [0\ 0\ 1\ 0\ 1]^\top$.
- Can you see the connection to Max-CUT?

Reducing DBHP to Max-CUT

Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

$$w_2 = [1\ 0\ 1]^\top$$

$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

$$w_2 = [1\ 0\ 1]^\top$$

$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

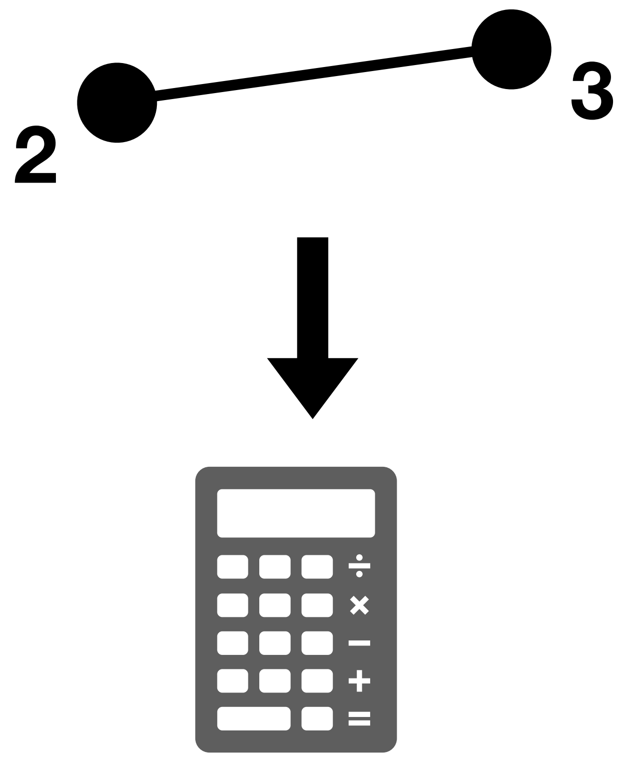
$$w_2 = [1\ 0\ 1]^\top$$

$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$



Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

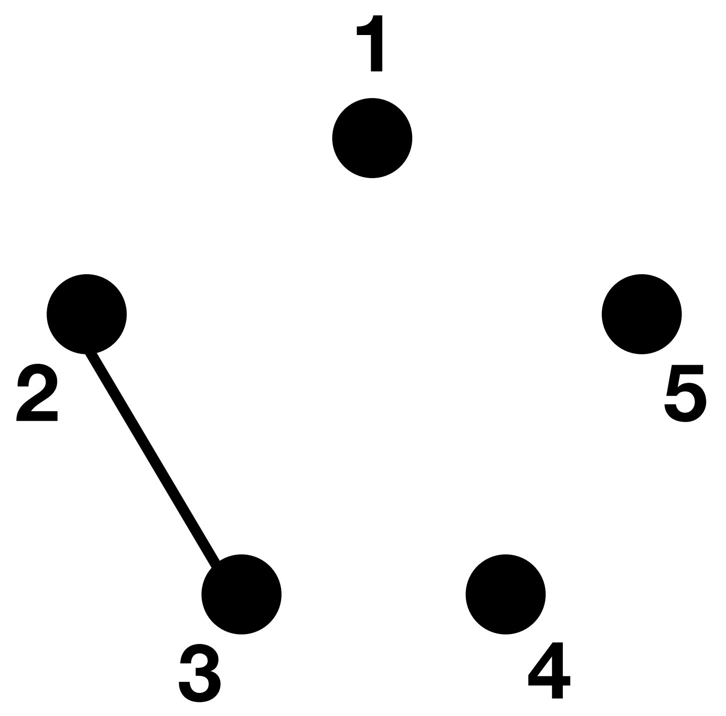
$$w_2 = [1\ 0\ 1]^\top$$

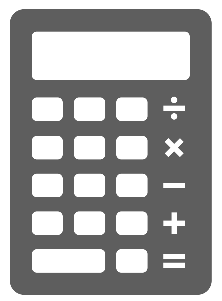
$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$





Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

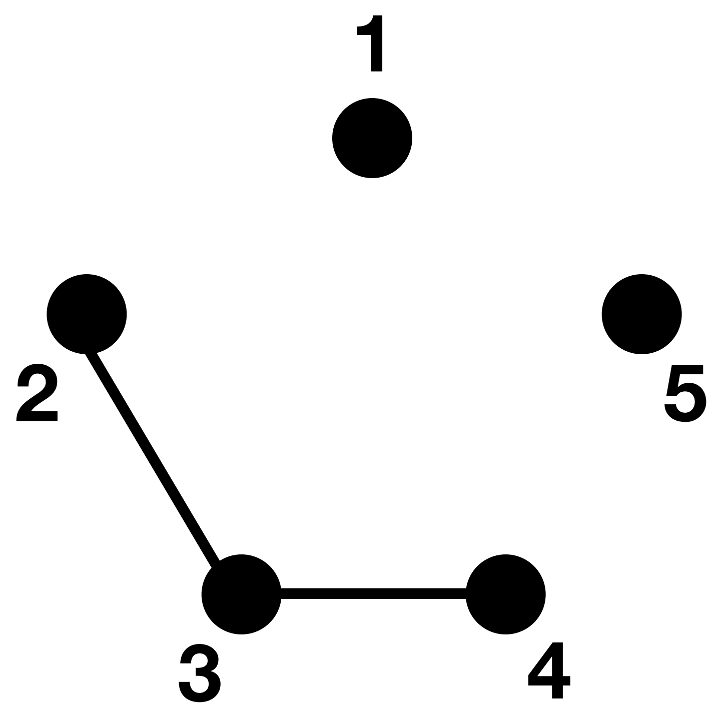
$$w_2 = [1\ 0\ 1]^\top$$

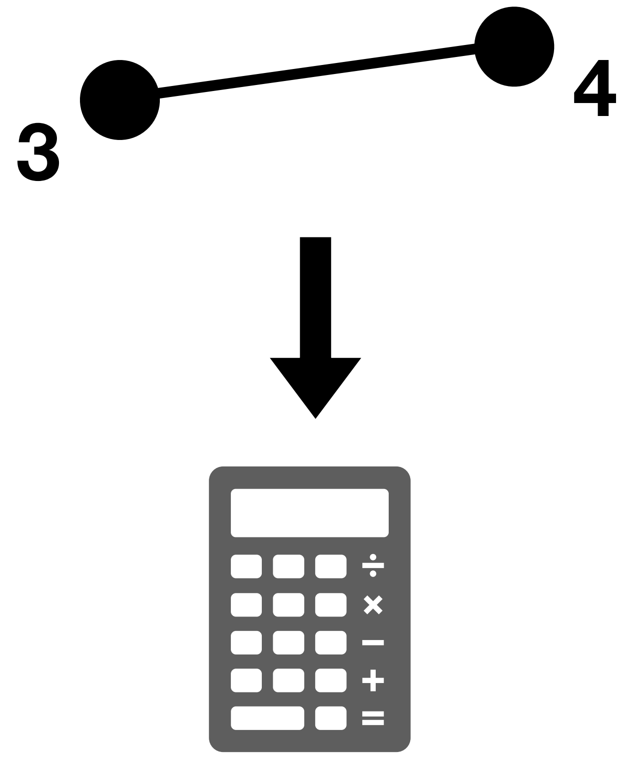
$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$





Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

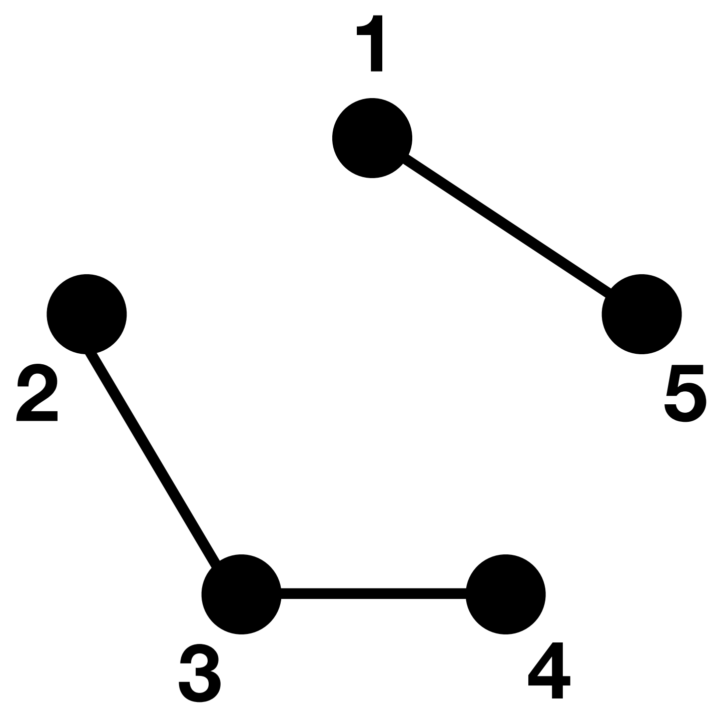
$$w_2 = [1\ 0\ 1]^\top$$

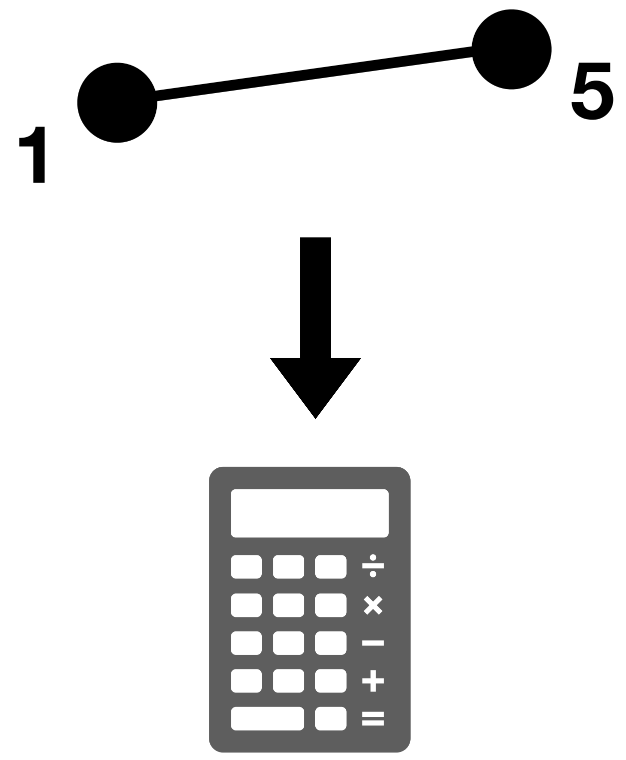
$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$





Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

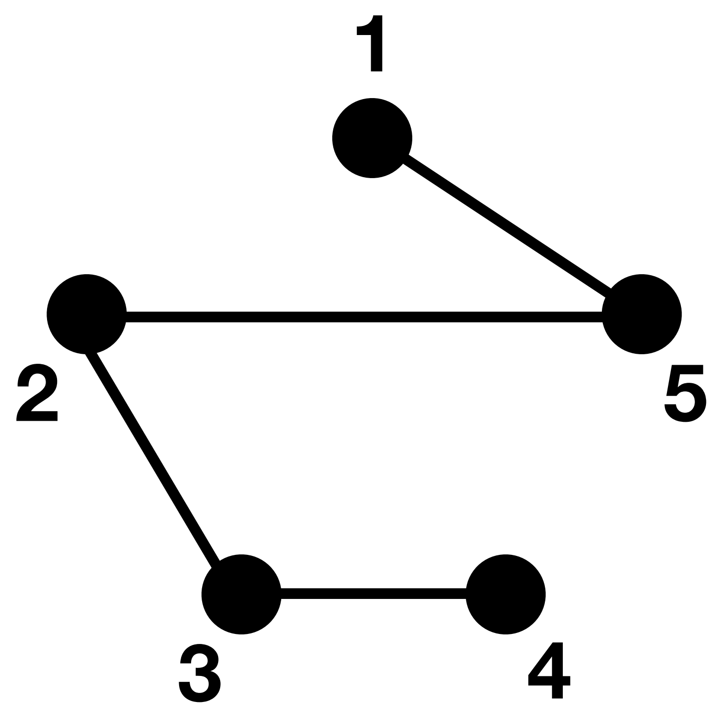
$$w_2 = [1\ 0\ 1]^\top$$

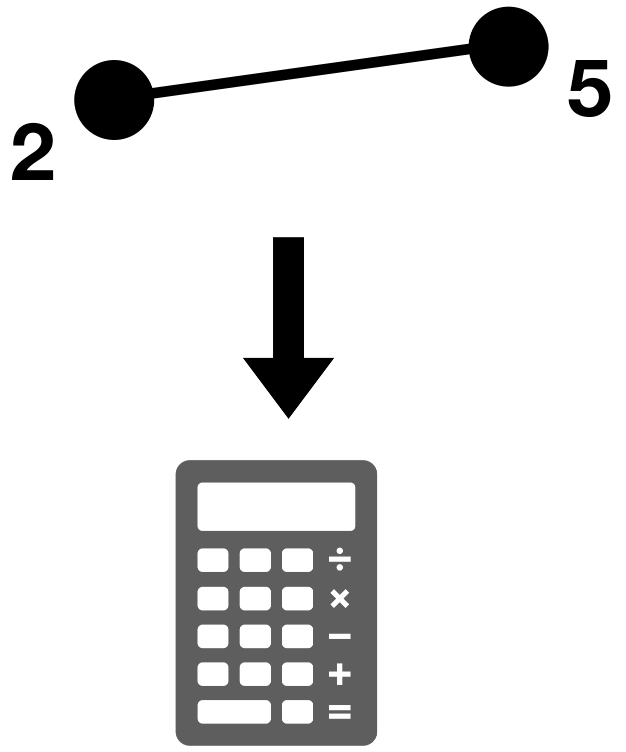
$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$





Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

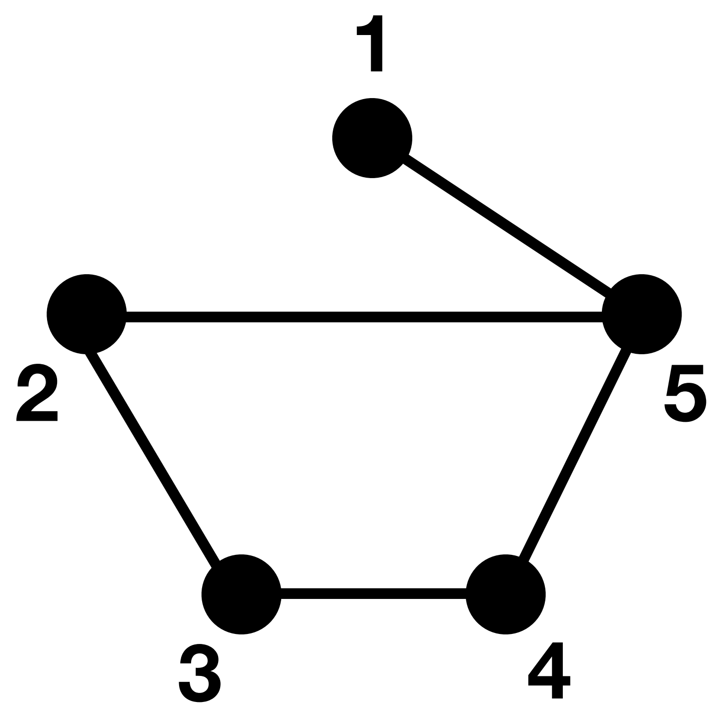
$$w_2 = [1\ 0\ 1]^\top$$

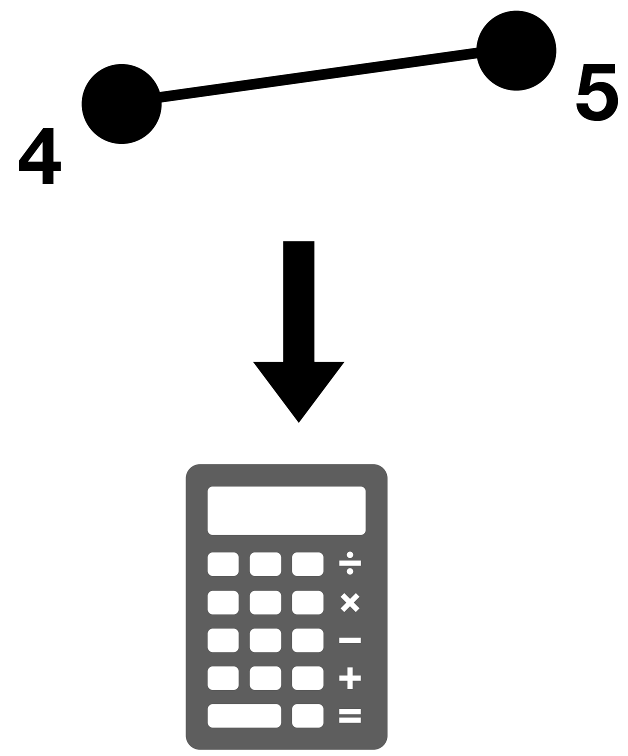
$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$





Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

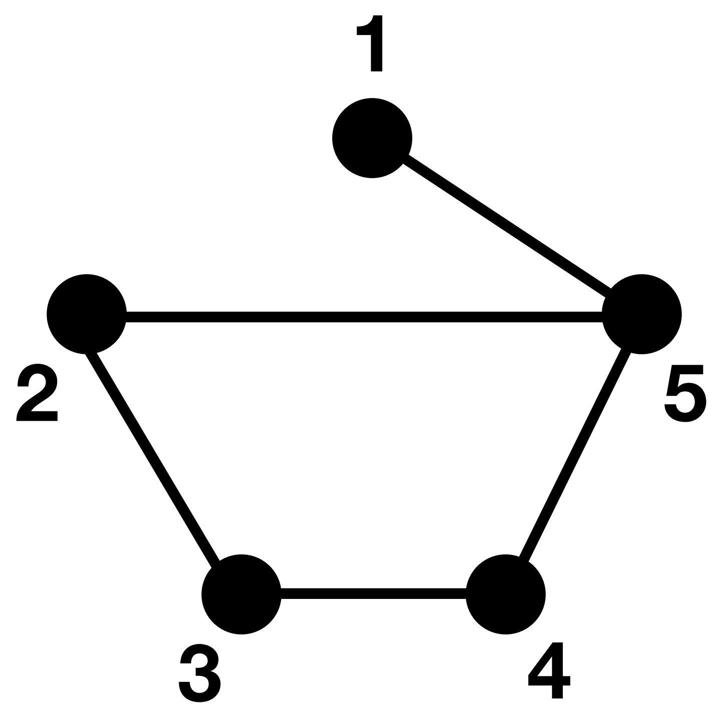
$$w_2 = [1\ 0\ 1]^\top$$

$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$



$\rightarrow v$

Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

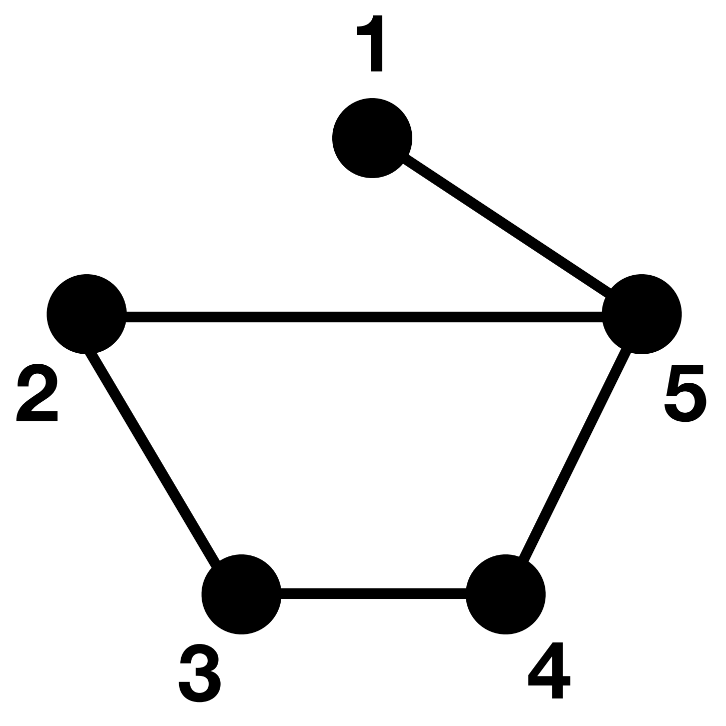
$$w_2 = [1\ 0\ 1]^\top$$

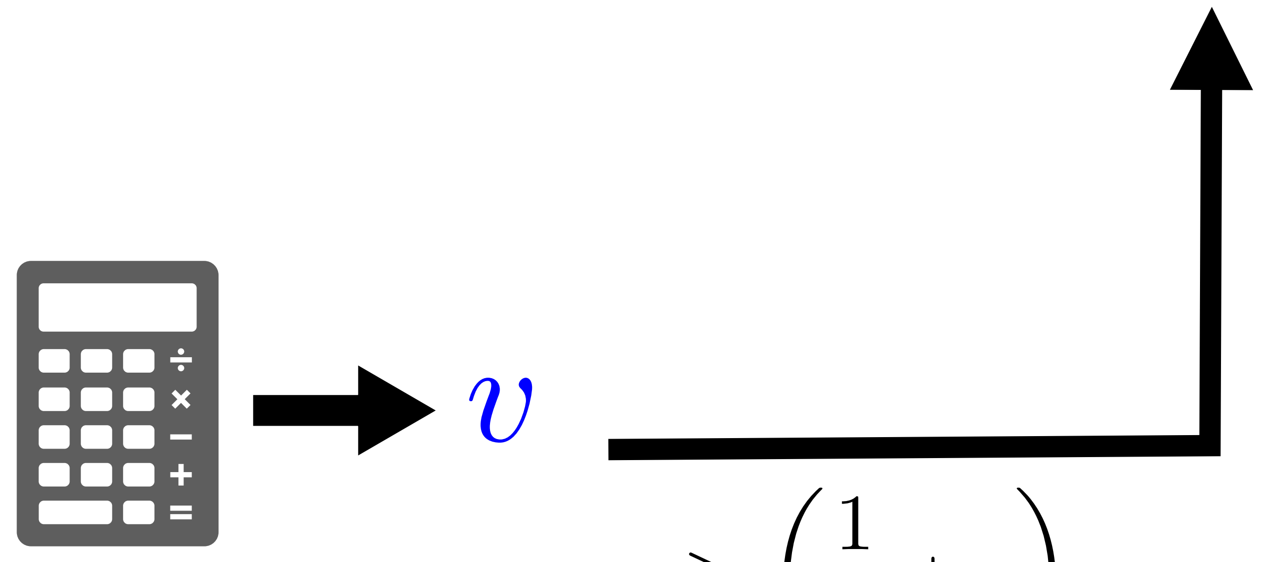
$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$





v

$v \geq \left(\frac{1}{2} + \epsilon\right) \cdot m$

Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

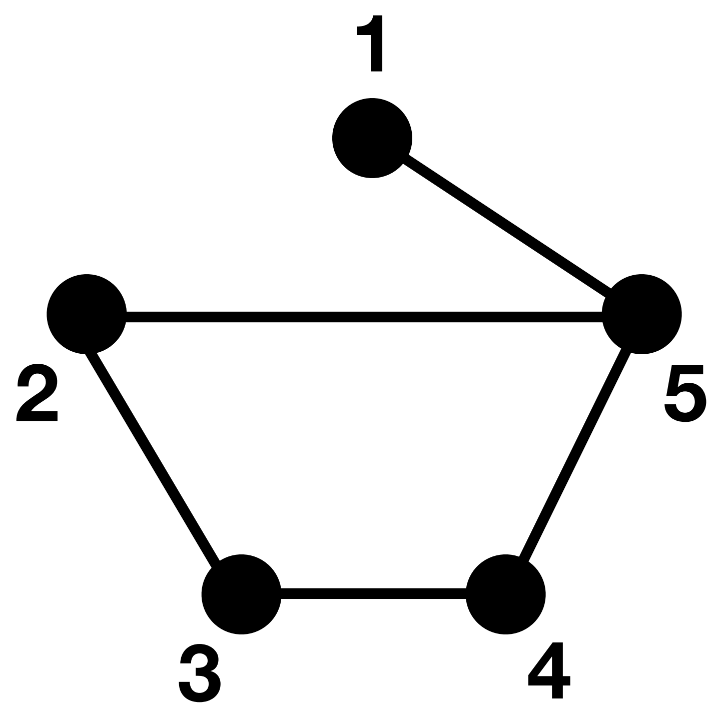
$$w_2 = [1\ 0\ 1]^\top$$

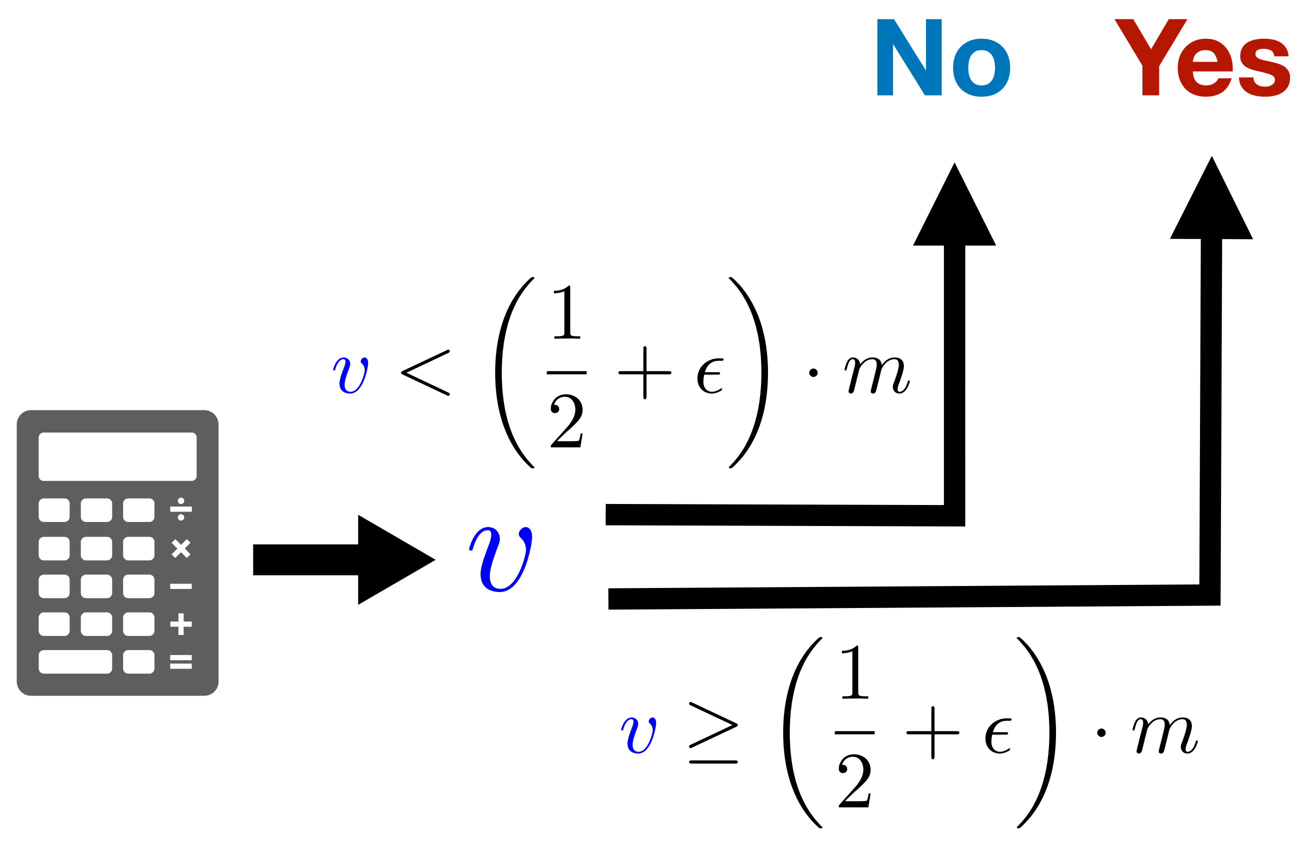
$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$





Reducing DBHP to Max-CUT

Bob 1

$$w_1 = [1\ 0\ 0]^\top$$

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Bob 2

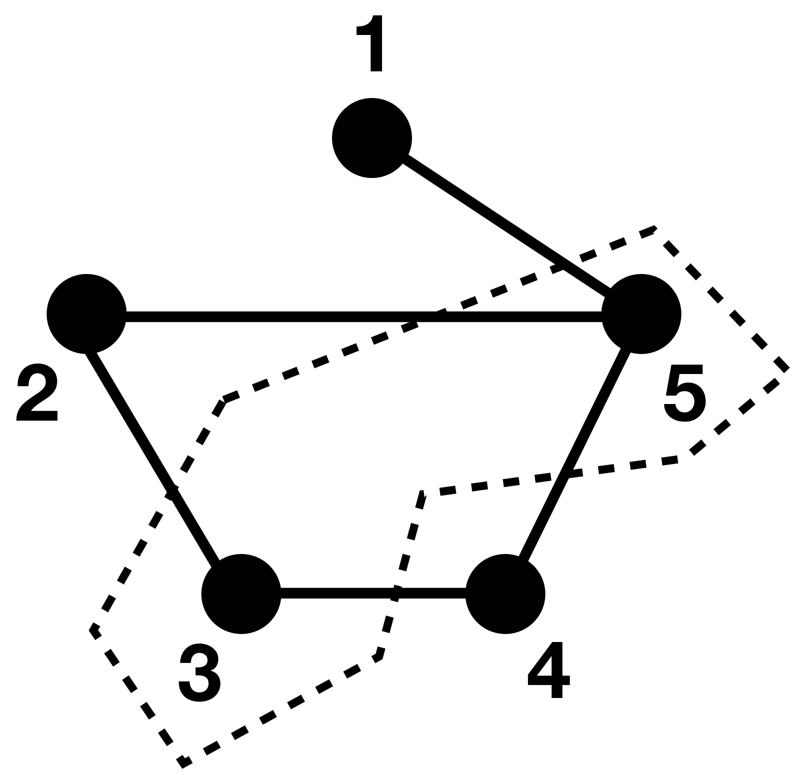
$$w_2 = [1\ 0\ 1]^\top$$

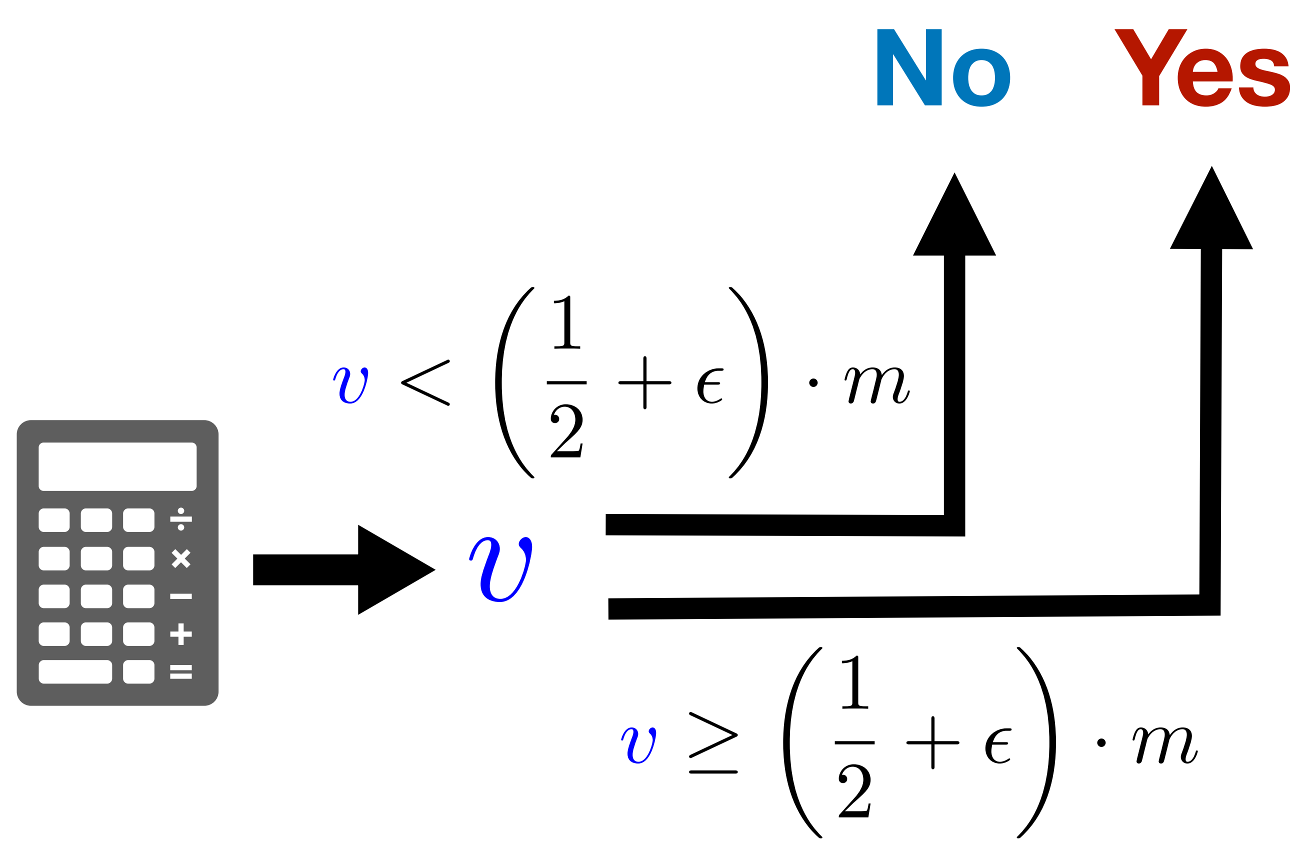
$$M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Bob 3

$$w_3 = [1\ 1\ 0]^\top$$

$$M_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$





How to Use DBHP? A Graph View

How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.

How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.

Yes Distribution

No Distribution

How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.

Yes Distribution

$\exists X^* \text{ s.t. } w_t = M_t X^*$

No Distribution

How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.

Yes Distribution

$\exists X^*$ s.t. $w_t = M_t X^*$

No Distribution

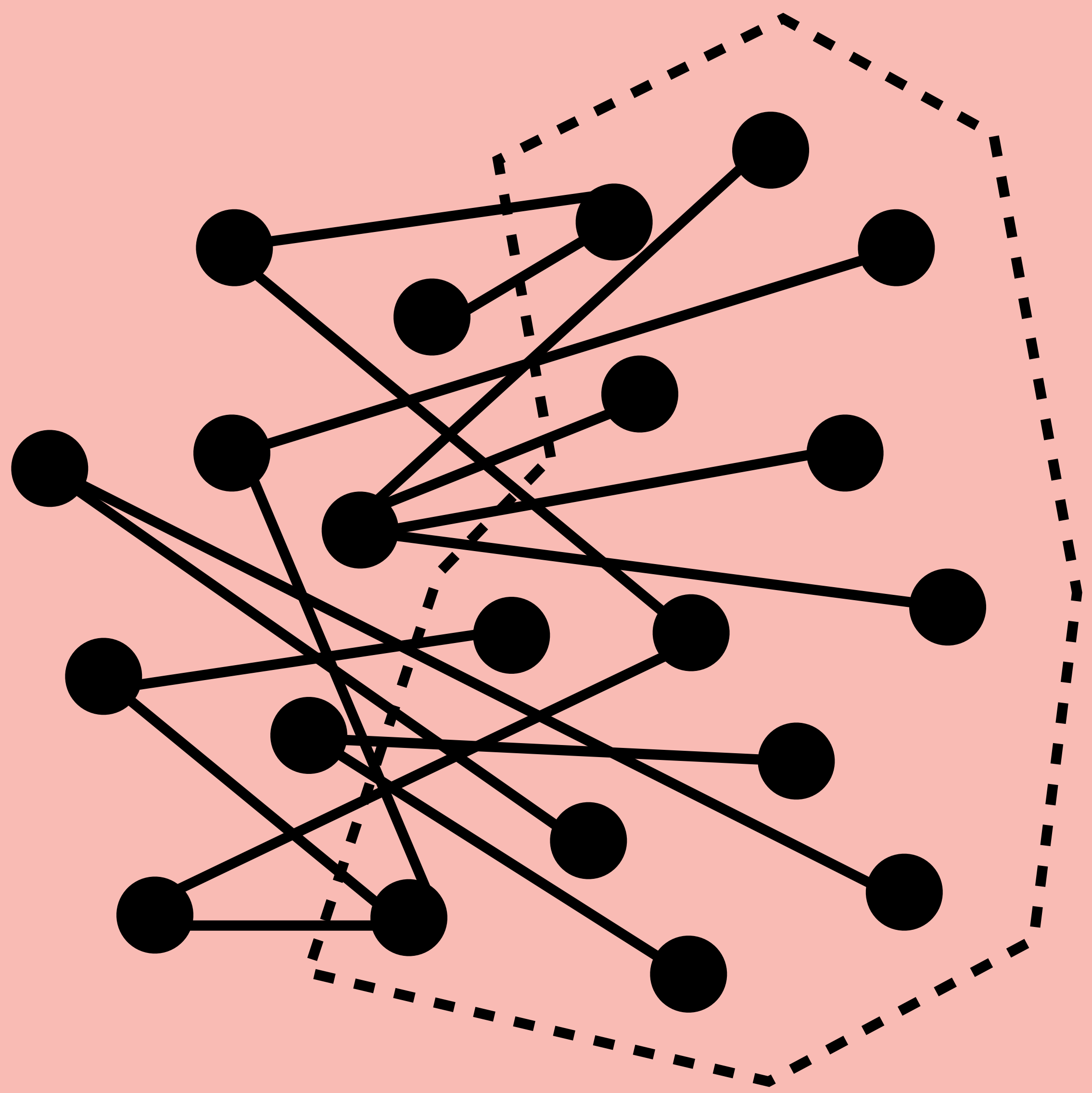
w_t is uniformly random

How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.

Yes Distribution

$\exists X^*$ s.t. $w_t = M_t X^*$



No Distribution

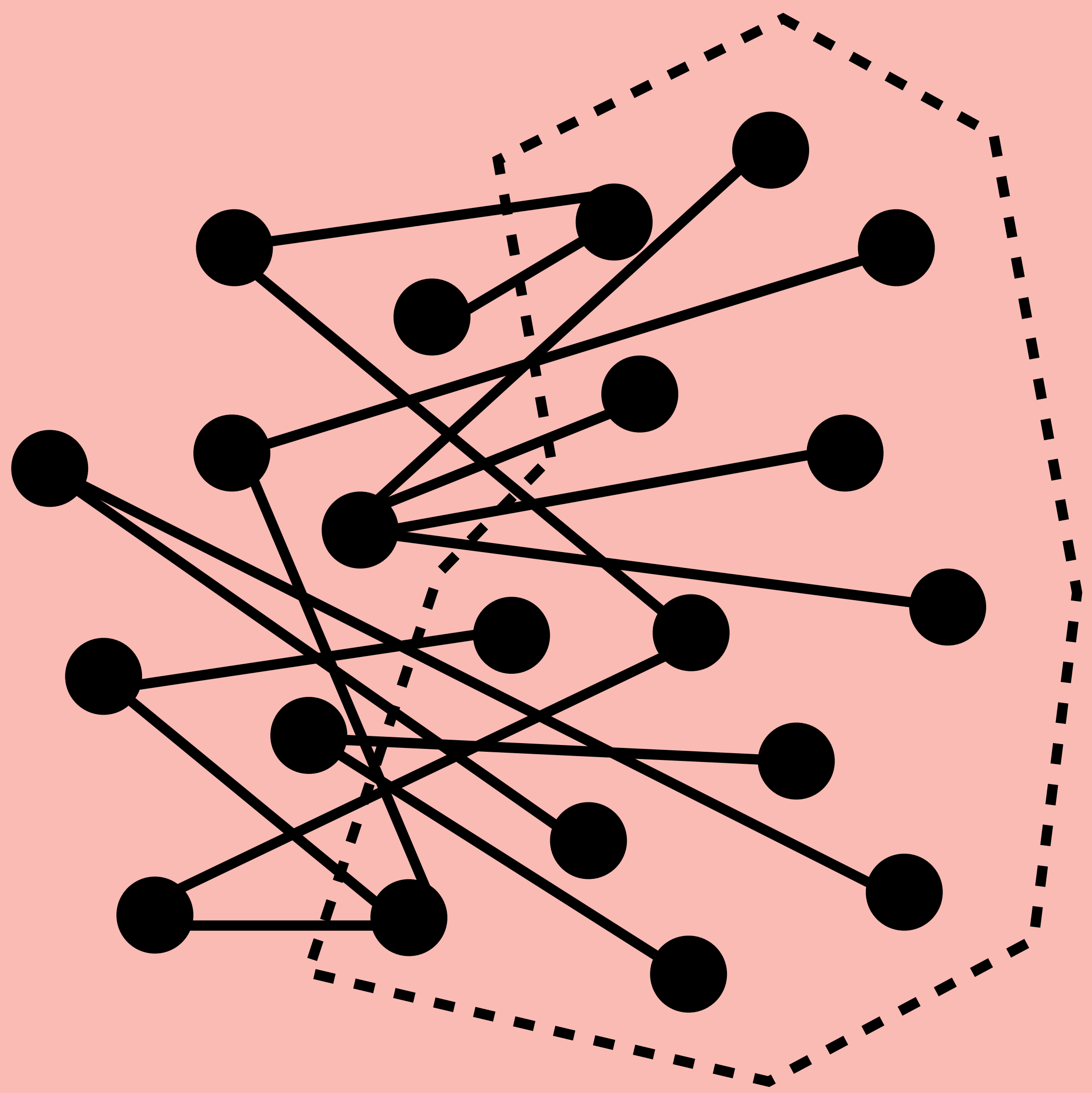
w_t is uniformly random

How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.

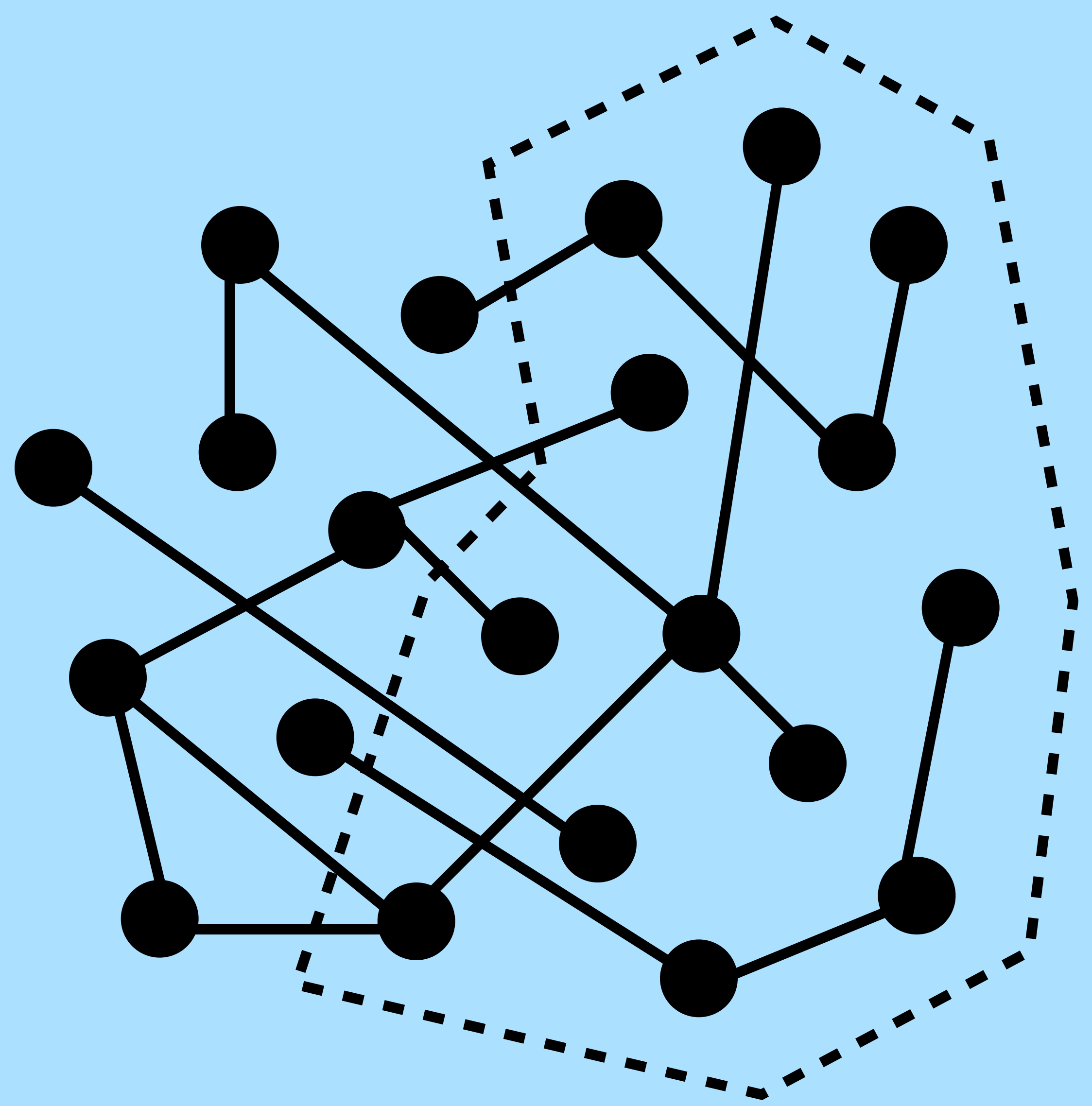
Yes Distribution

$\exists X^* \text{ s.t. } w_t = M_t X^*$



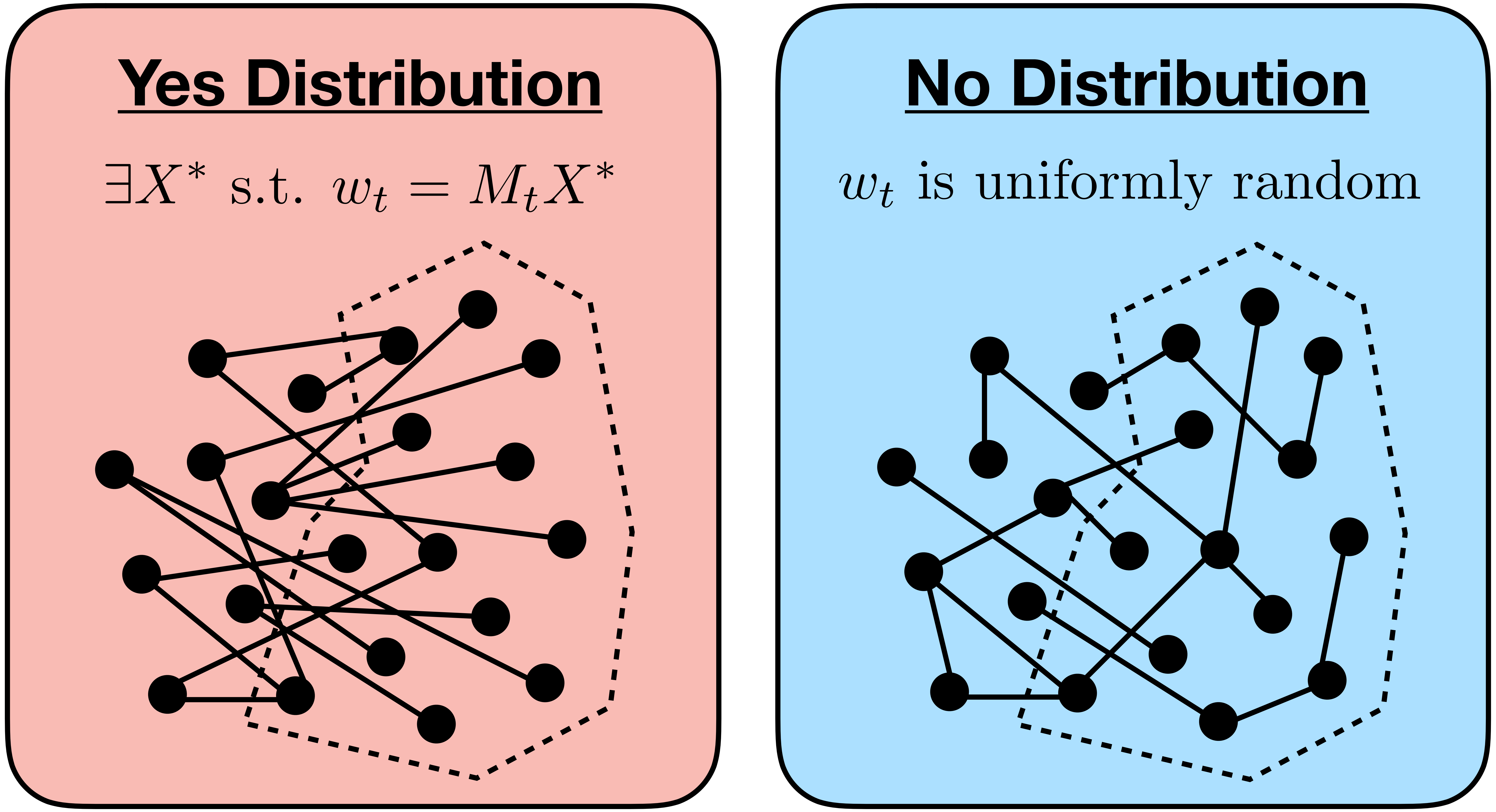
No Distribution

w_t is uniformly random



How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.



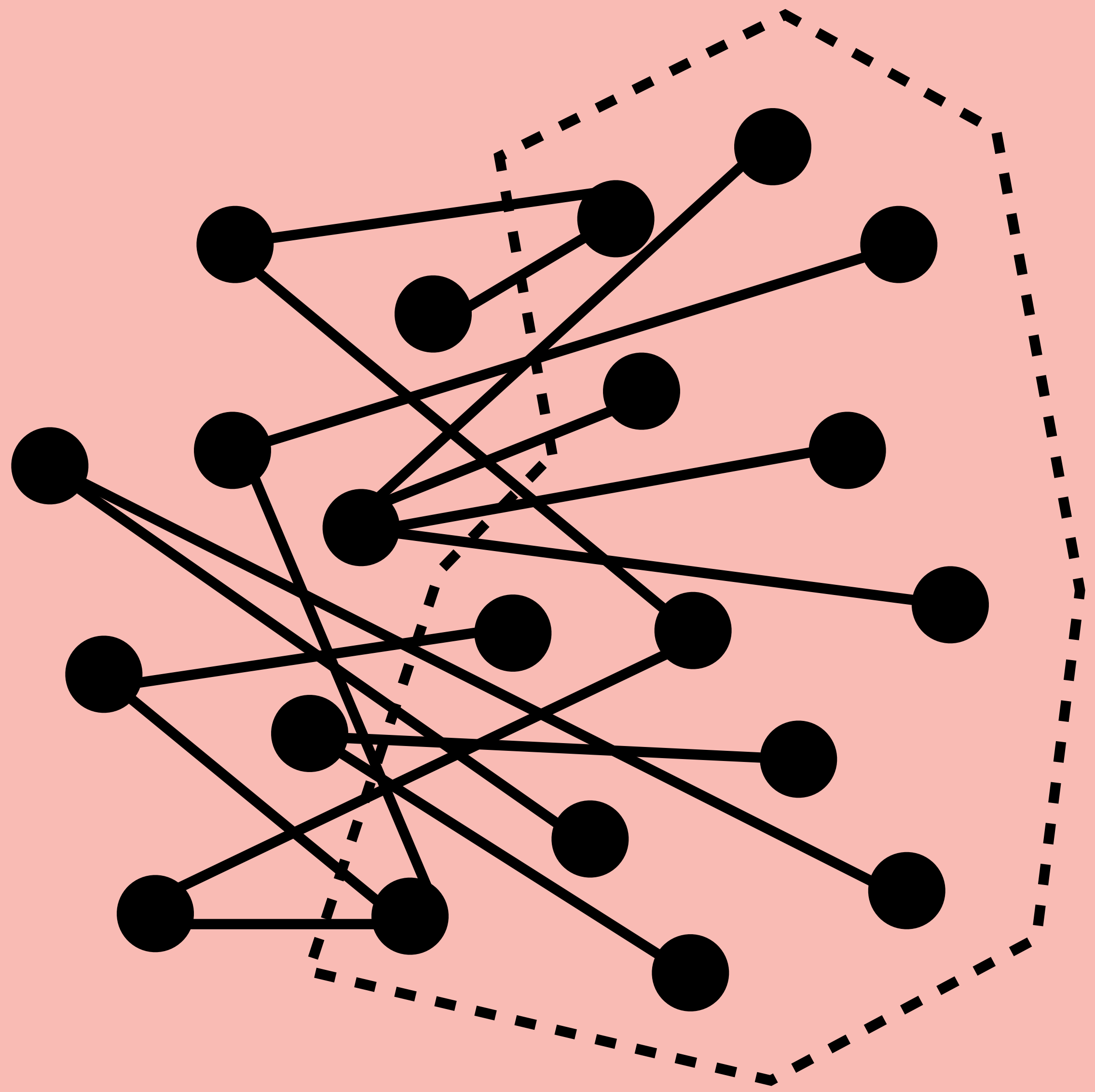
- Each player possesses a subset of the edges.

How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.

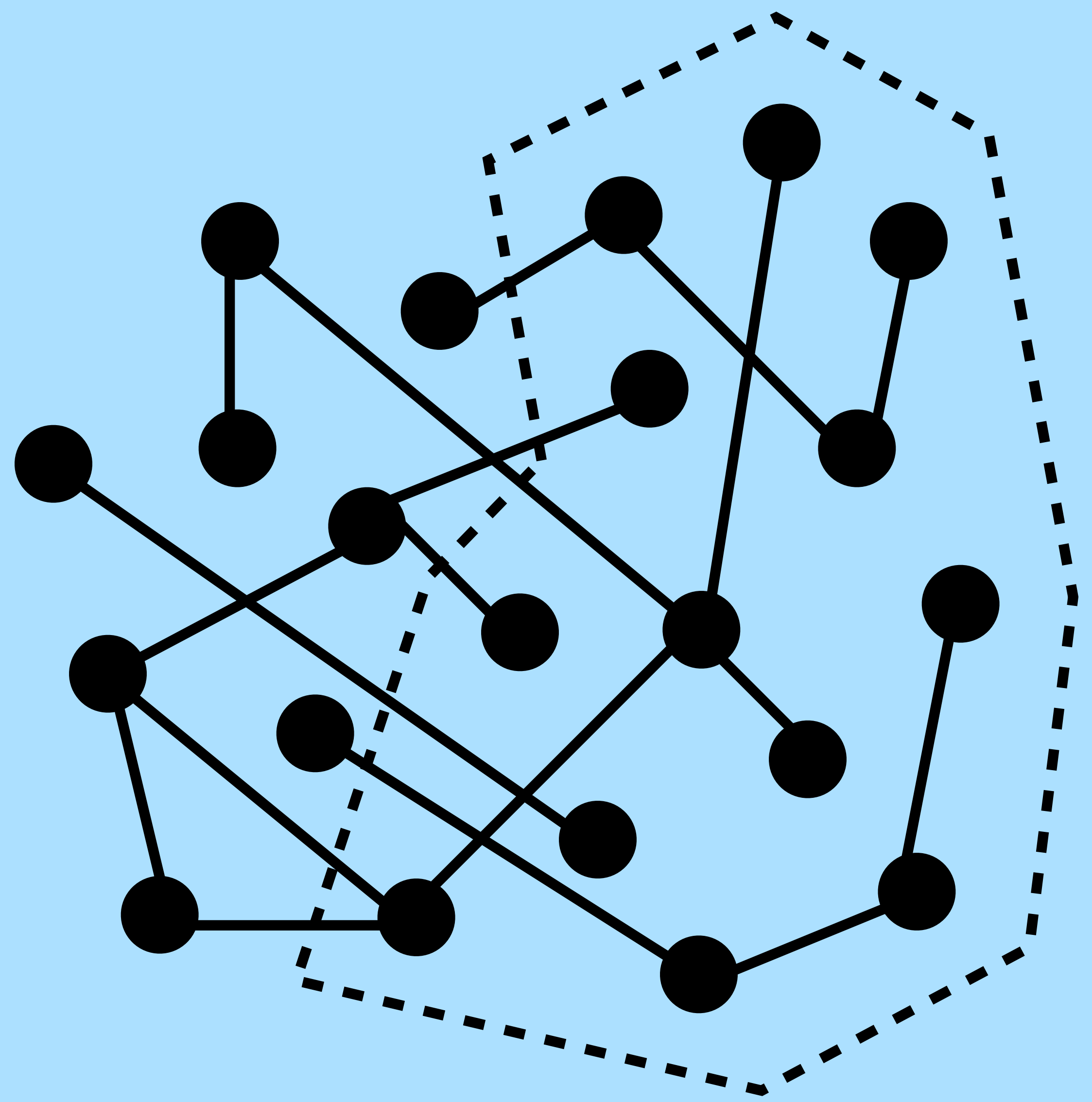
Yes Distribution

$\exists X^* \text{ s.t. } w_t = M_t X^*$



No Distribution

w_t is uniformly random



Yes' Distribution

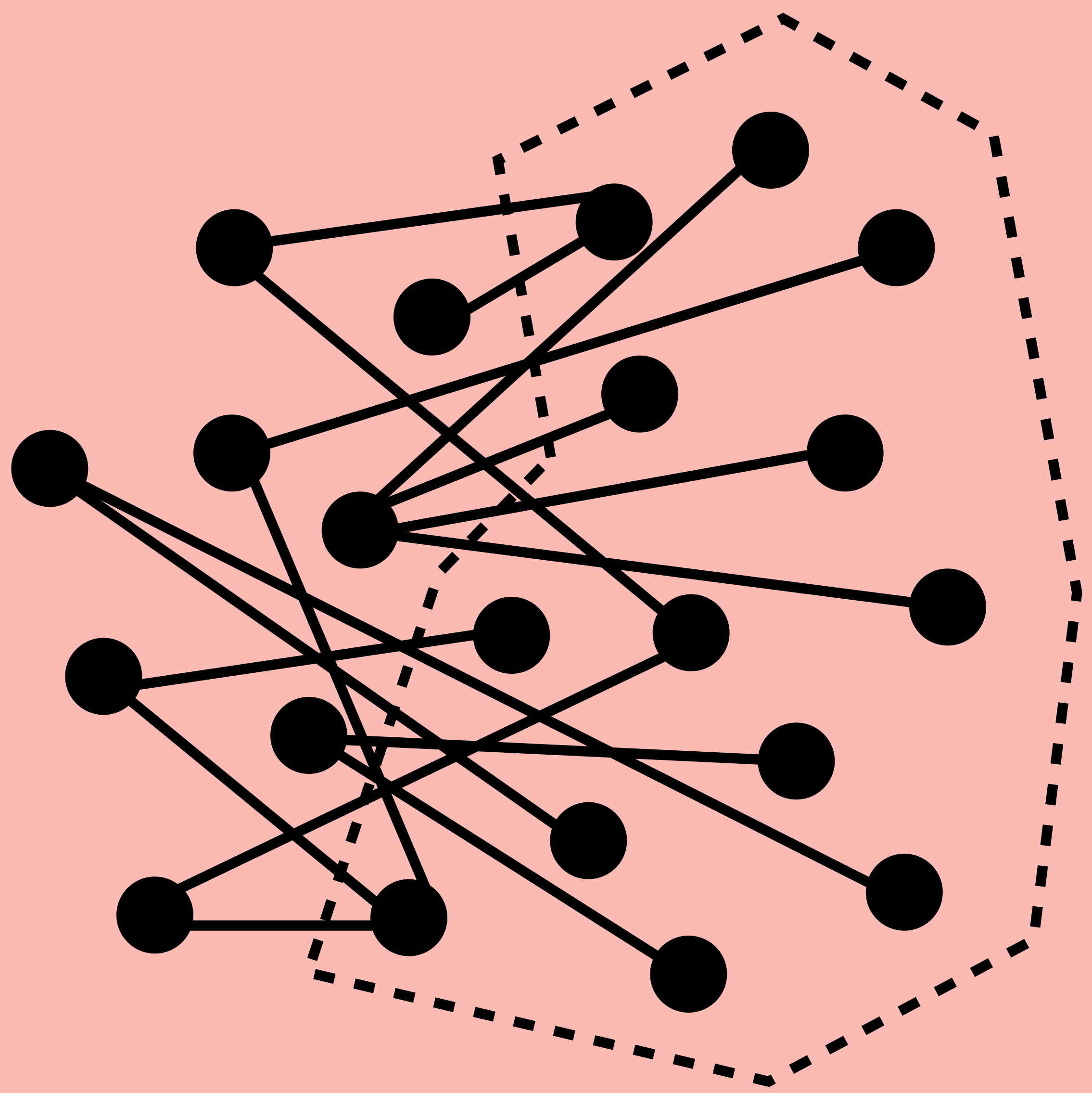
- Each player possesses a subset of the edges.

How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.

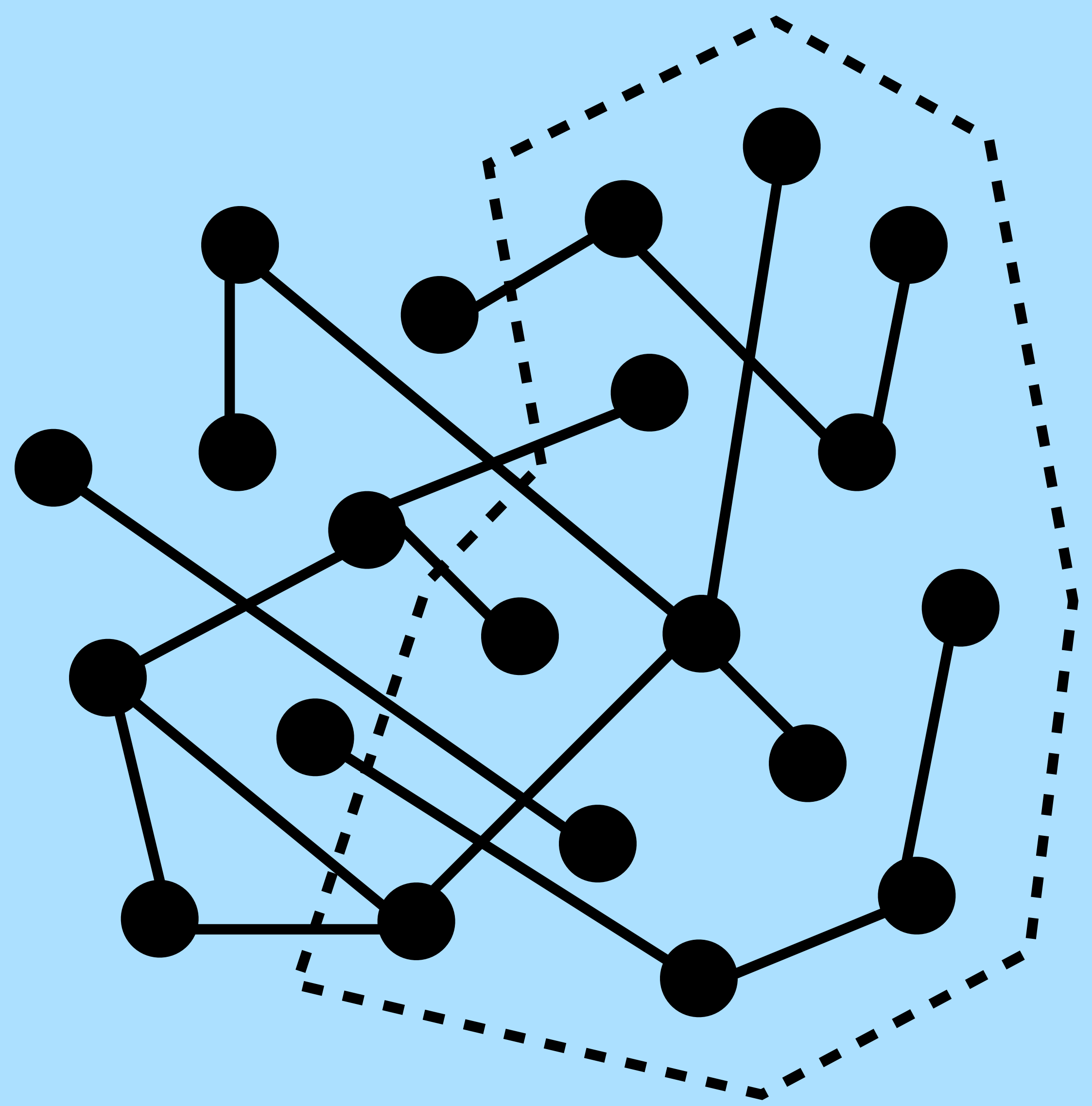
Yes Distribution

$\exists X^* \text{ s.t. } w_t = M_t X^*$



No Distribution

w_t is uniformly random



Yes' Distribution

$\exists X^* \text{ s.t. } w_t = \mathbf{1} - M_t X^*$

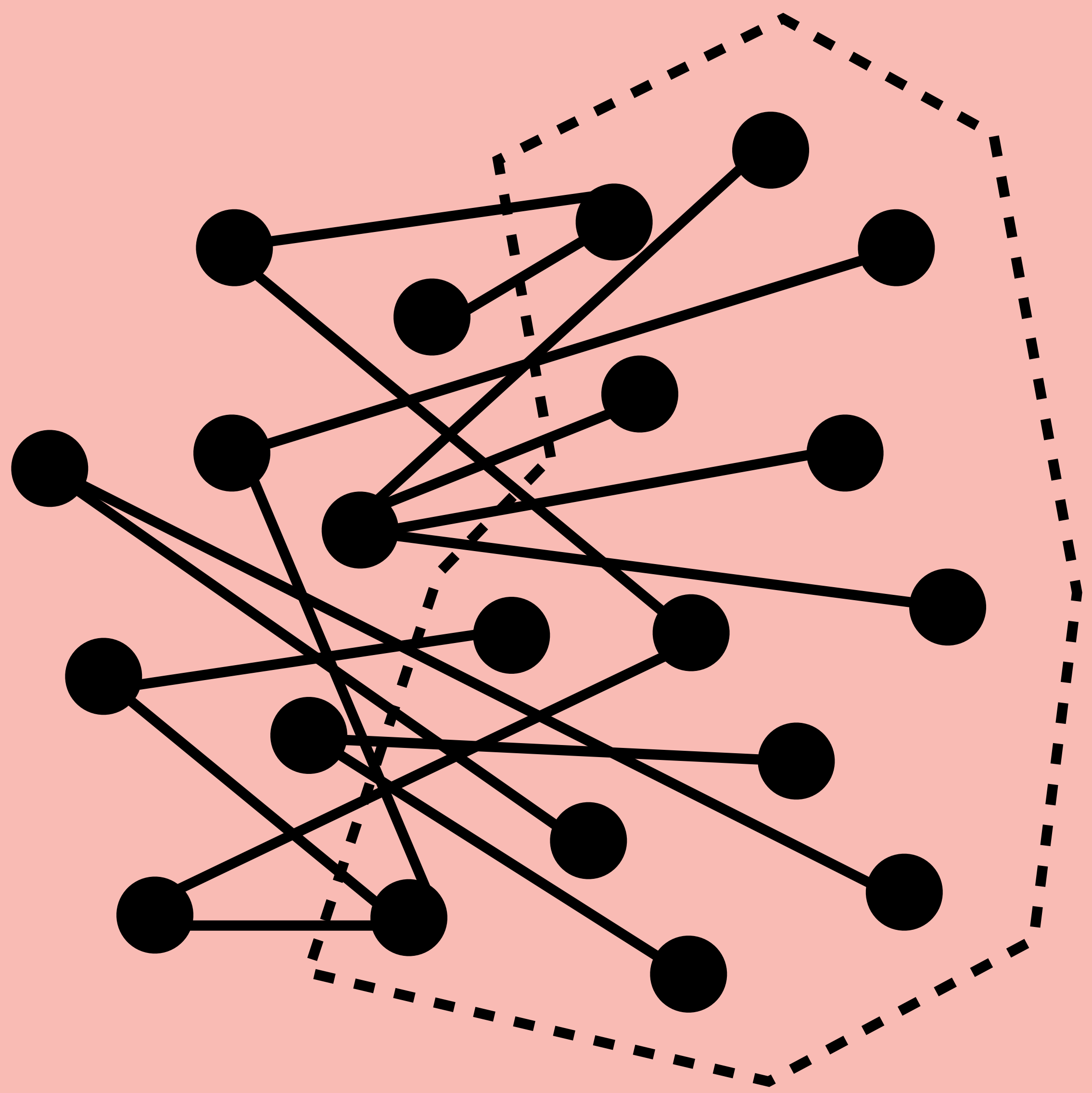
- Each player possesses a subset of the edges.

How to Use DBHP? A Graph View

- Think of each row of M_t as a random edge and w_t picks the edges.

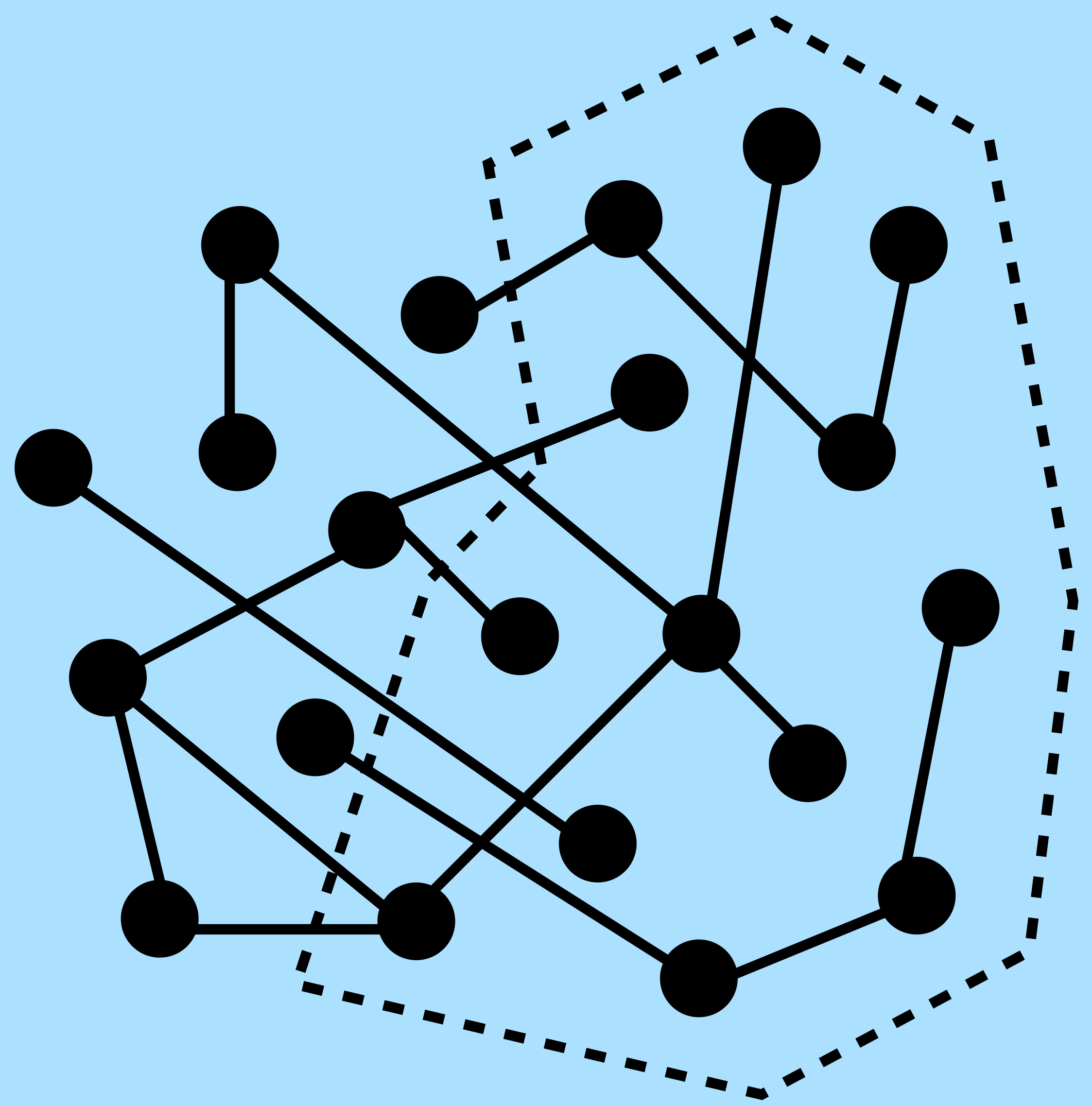
Yes Distribution

$\exists X^* \text{ s.t. } w_t = M_t X^*$



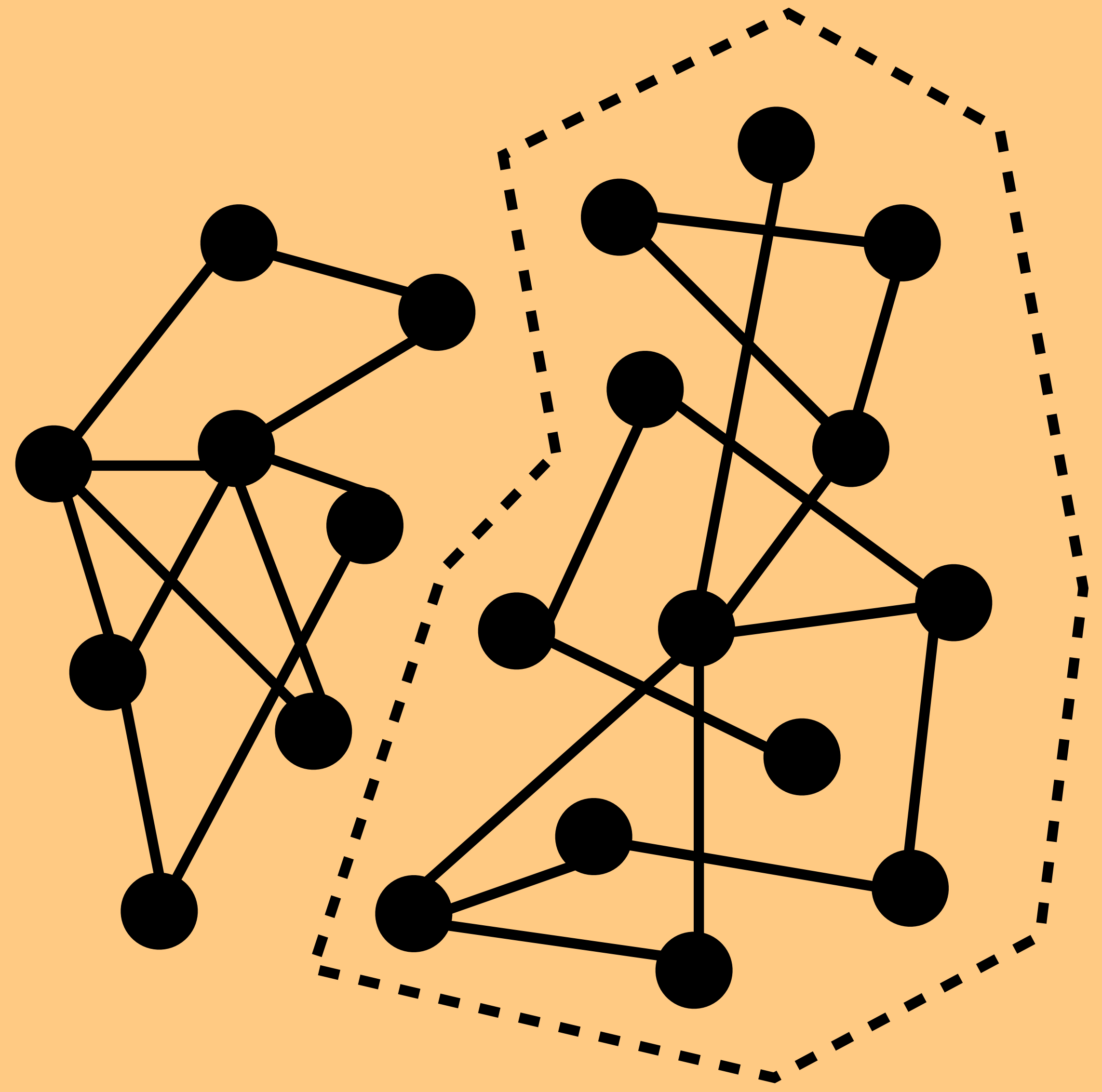
No Distribution

w_t is uniformly random



Yes' Distribution

$\exists X^* \text{ s.t. } w_t = \mathbf{1} - M_t X^*$



- Each player possesses a subset of the edges.

Reducing DBHP to Max-CUT

Bob 1

$w_1 \in \{0,1\}^{0.01n}$

Bob 2

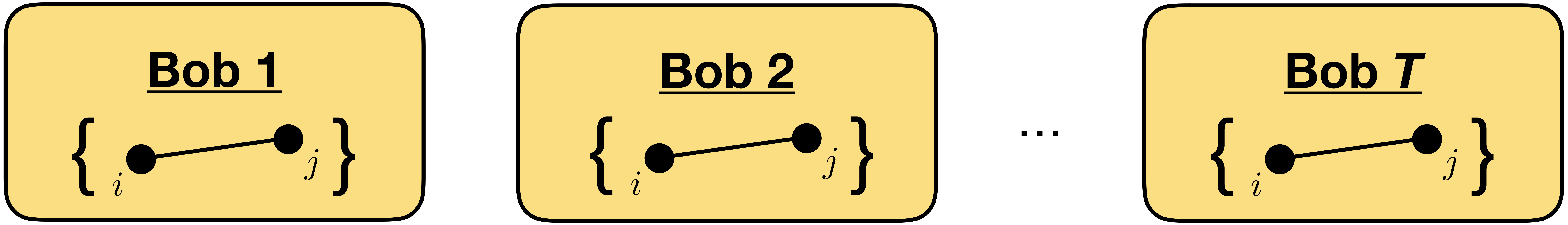
$w_2 \in \{0,1\}^{0.01n}$

…

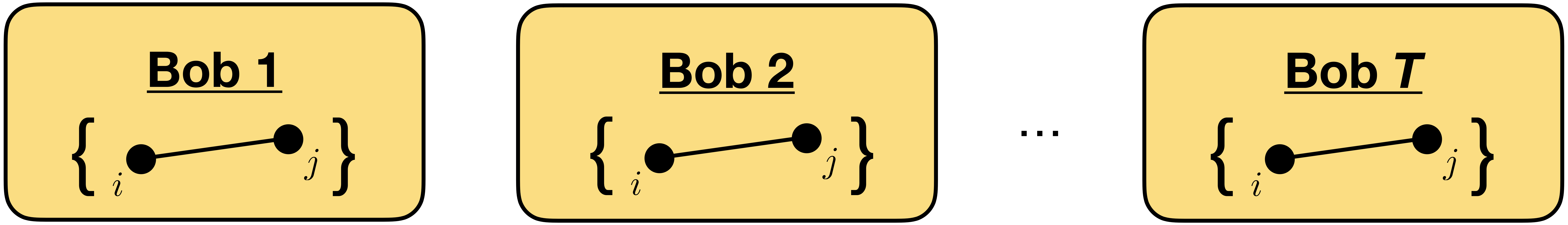
Bob *T*

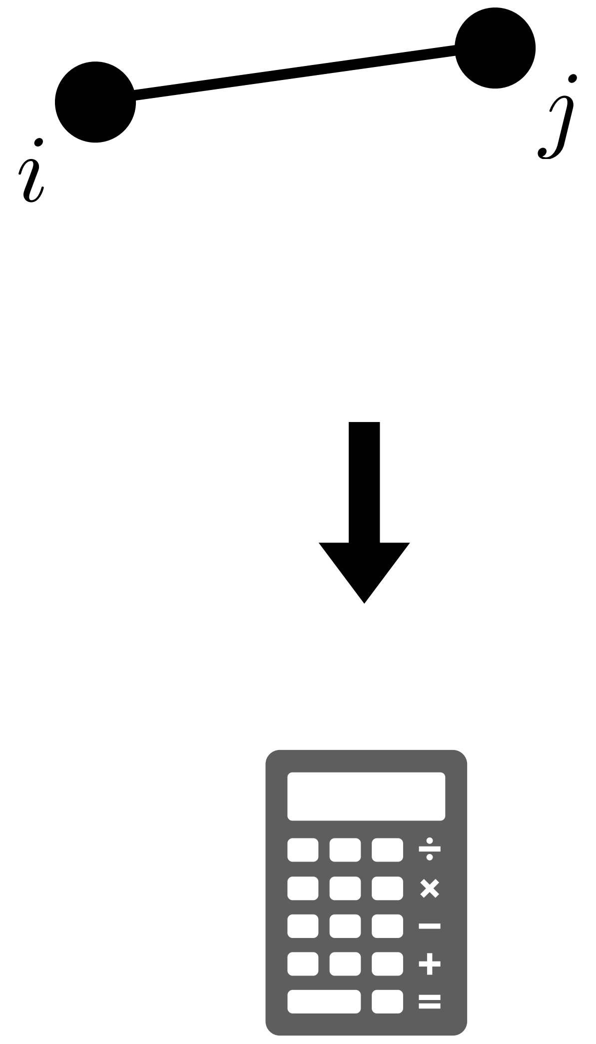
$w_T \in \{0,1\}^{0.01n}$

Reducing DBHP to Max-CUT

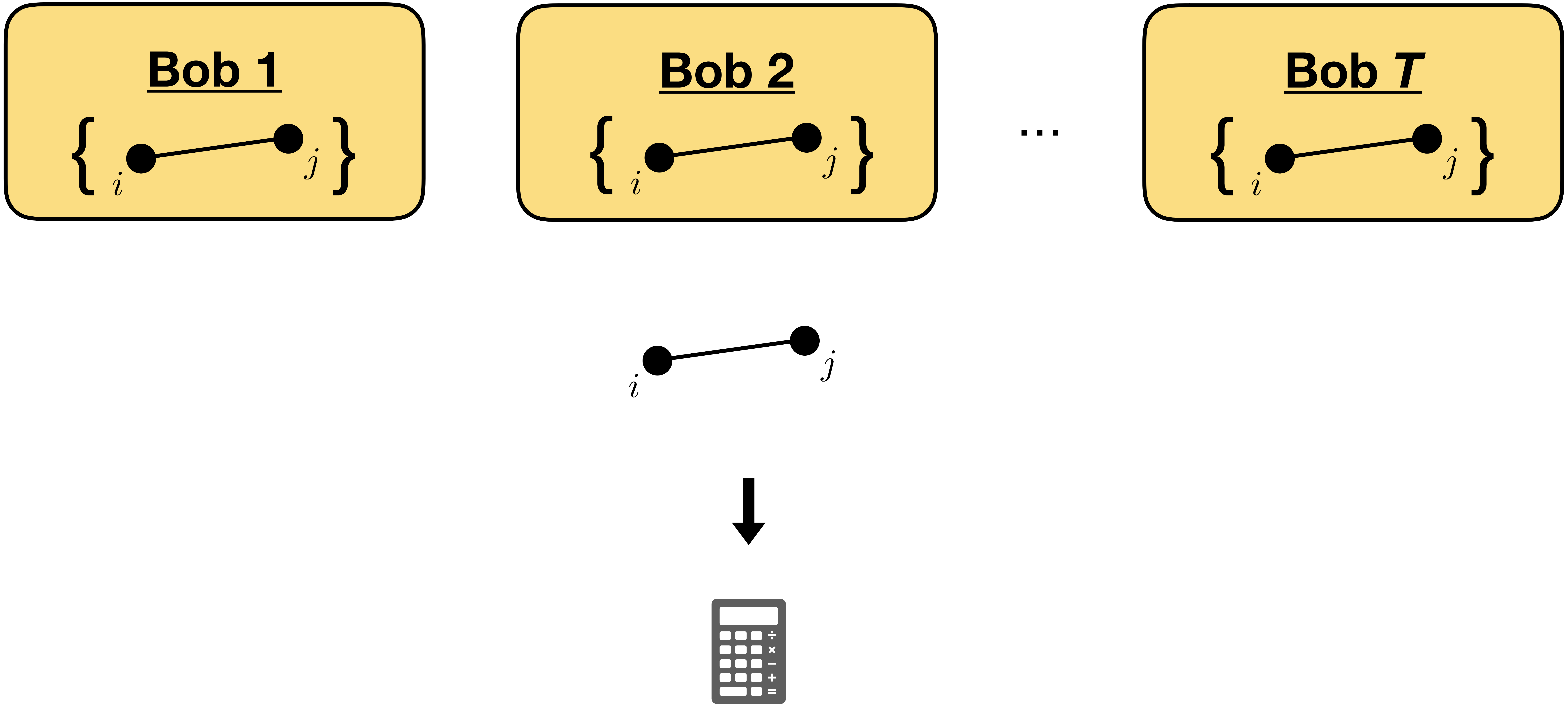


Reducing DBHP to Max-CUT

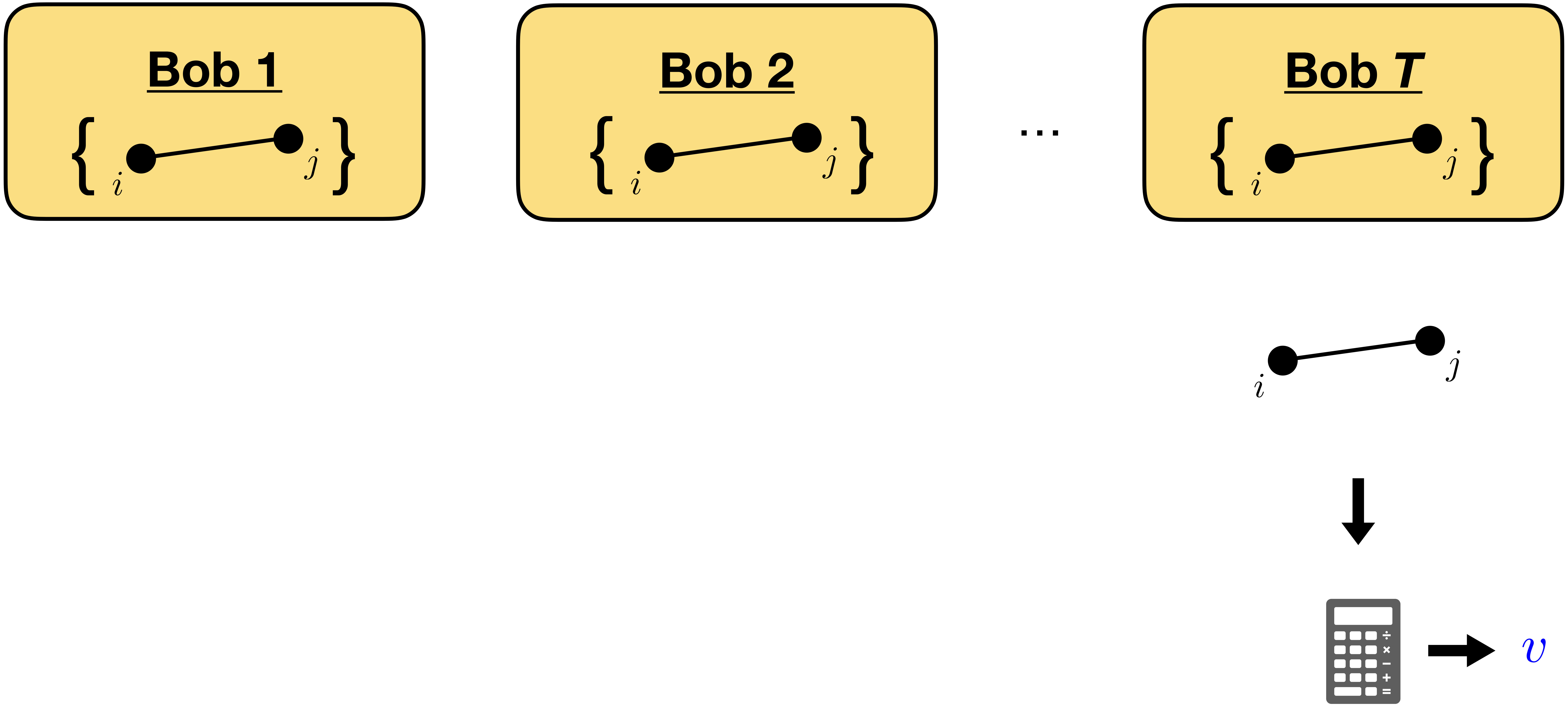




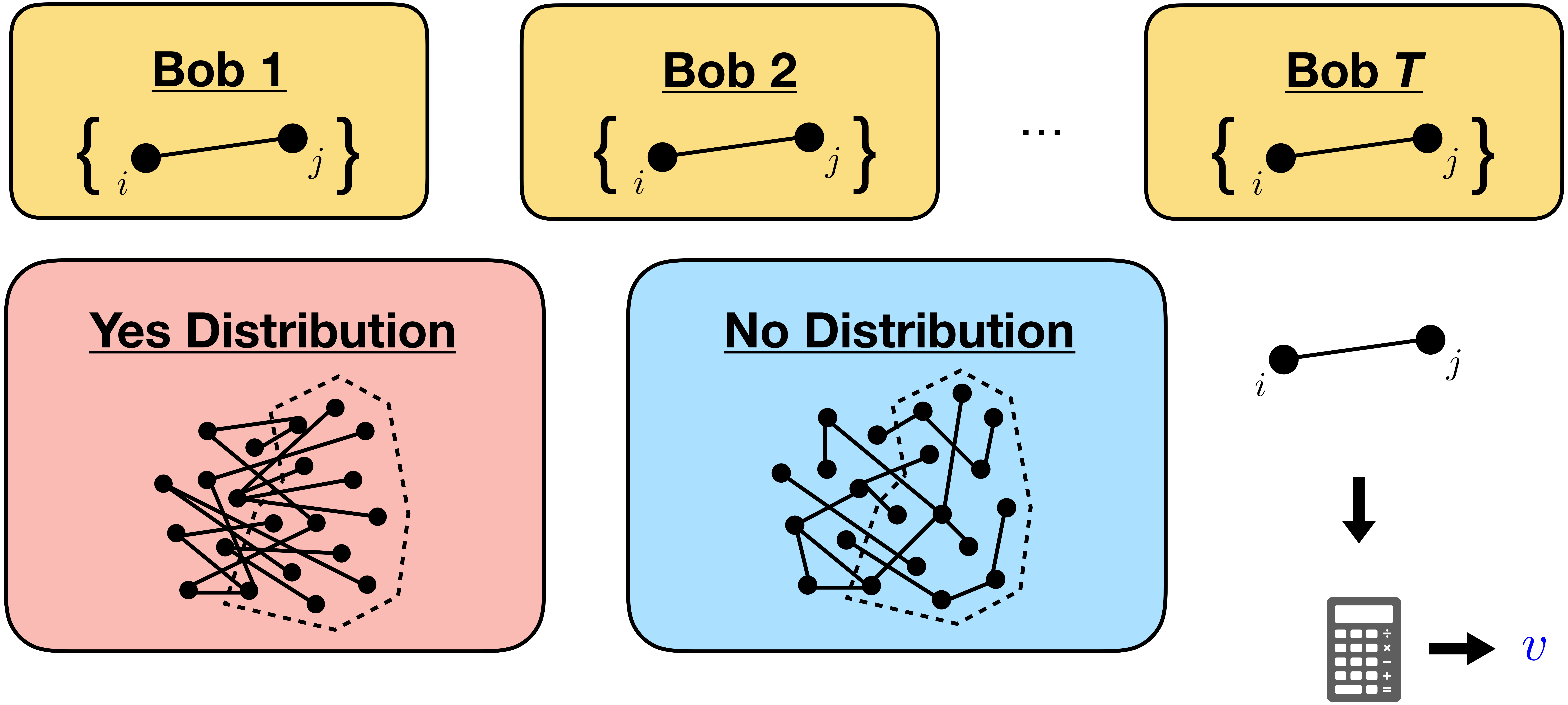
Reducing DBHP to Max-CUT



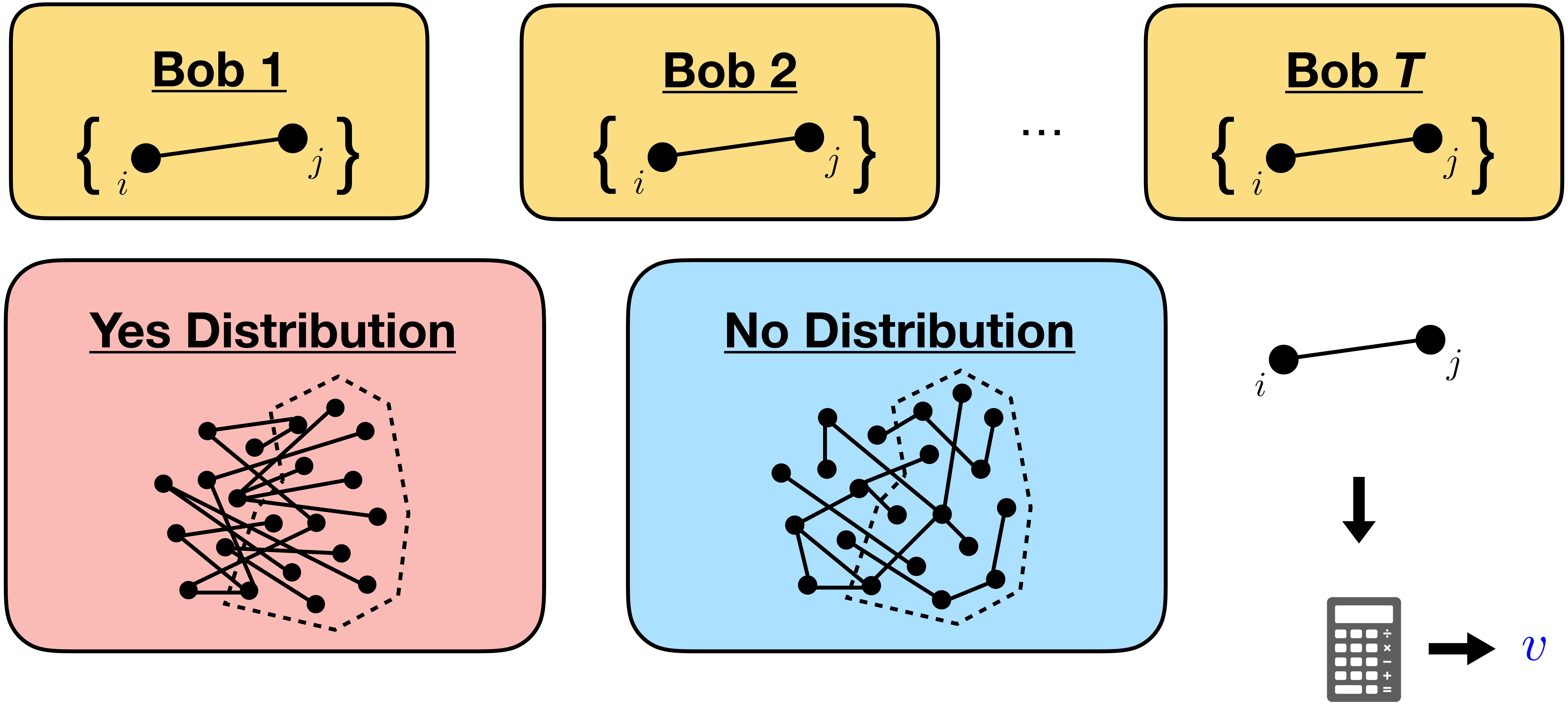
Reducing DBHP to Max-CUT



Reducing DBHP to Max-CUT

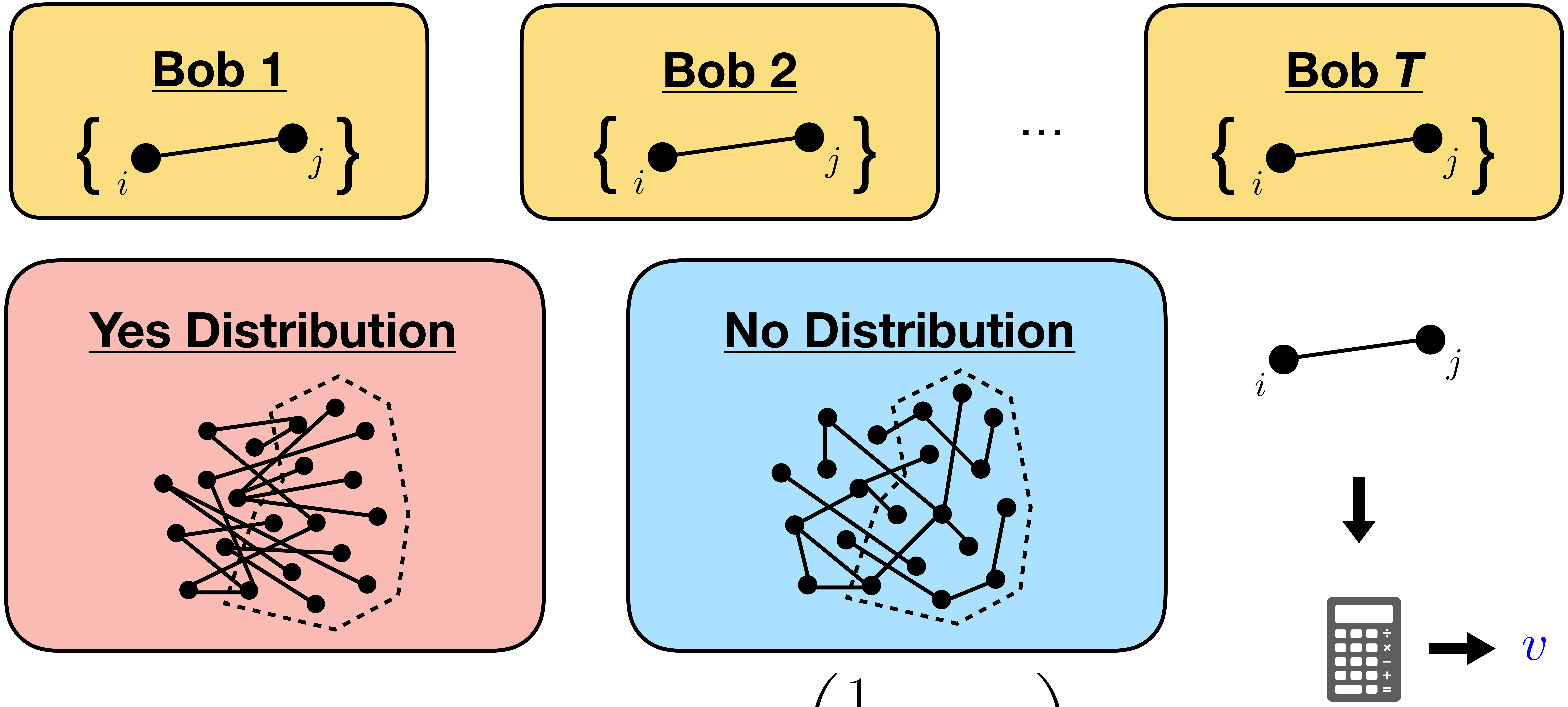


Reducing DBHP to Max-CUT



$\mathrm{val}_{\mathcal{C}} = m$

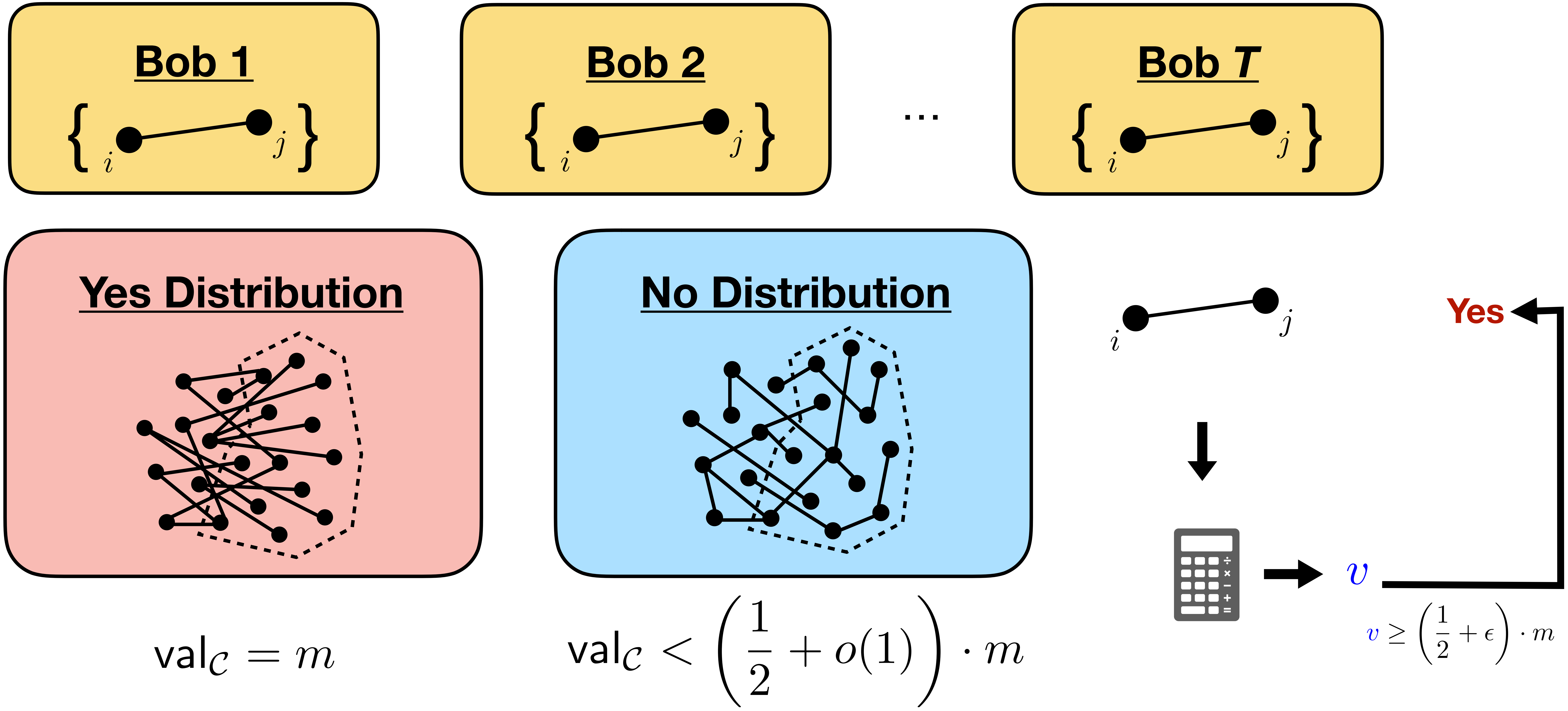
Reducing DBHP to Max-CUT



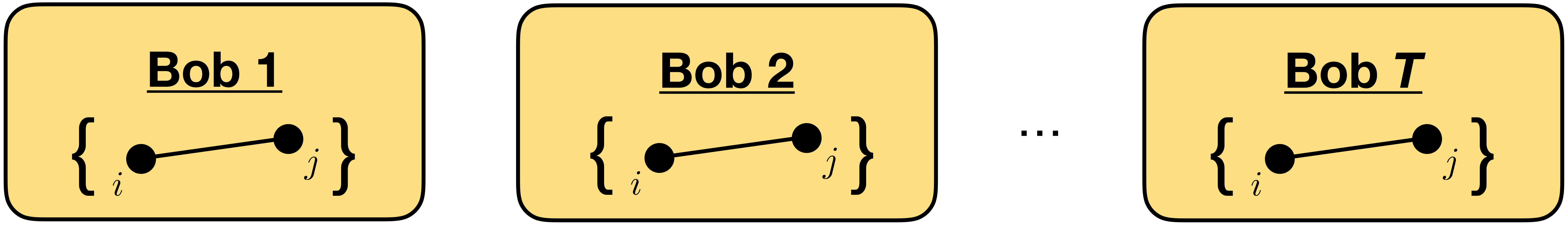
$$\mathsf{val}_{\mathcal{C}} = m$$

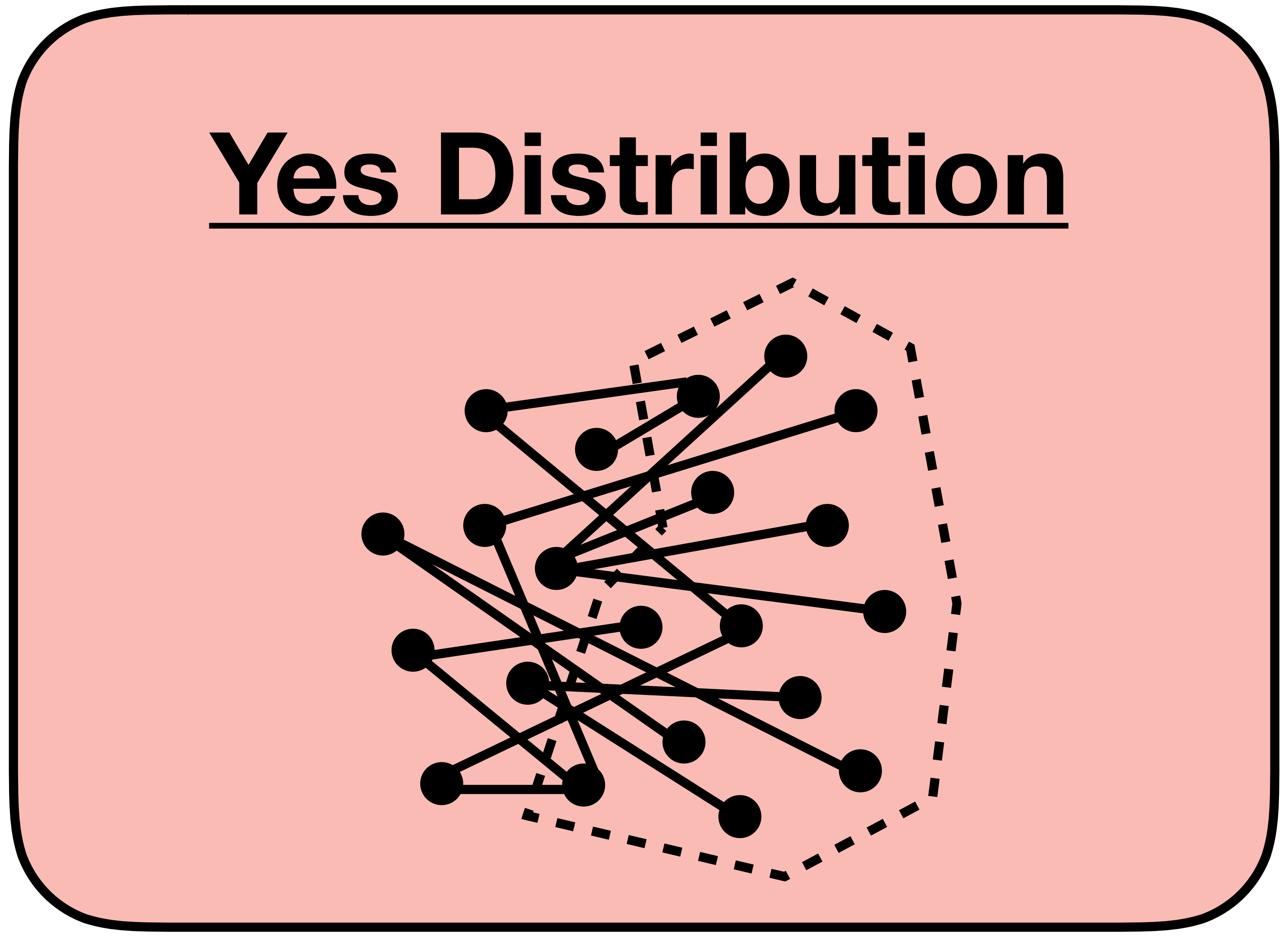
$$\mathsf{val}_{\mathcal{C}} < \left(\frac{1}{2} + o(1)\right) \cdot m$$

Reducing DBHP to Max-CUT

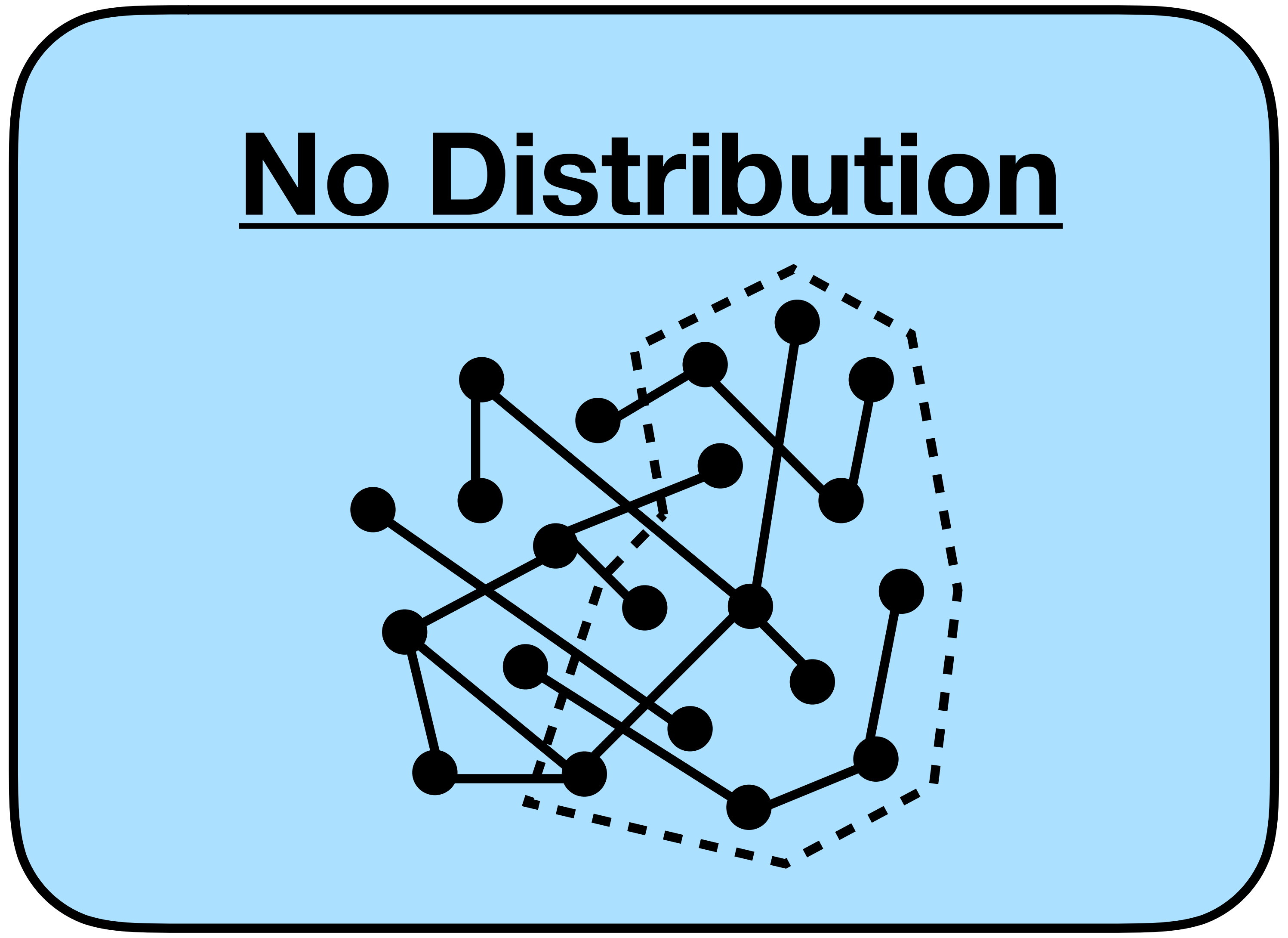


Reducing DBHP to Max-CUT

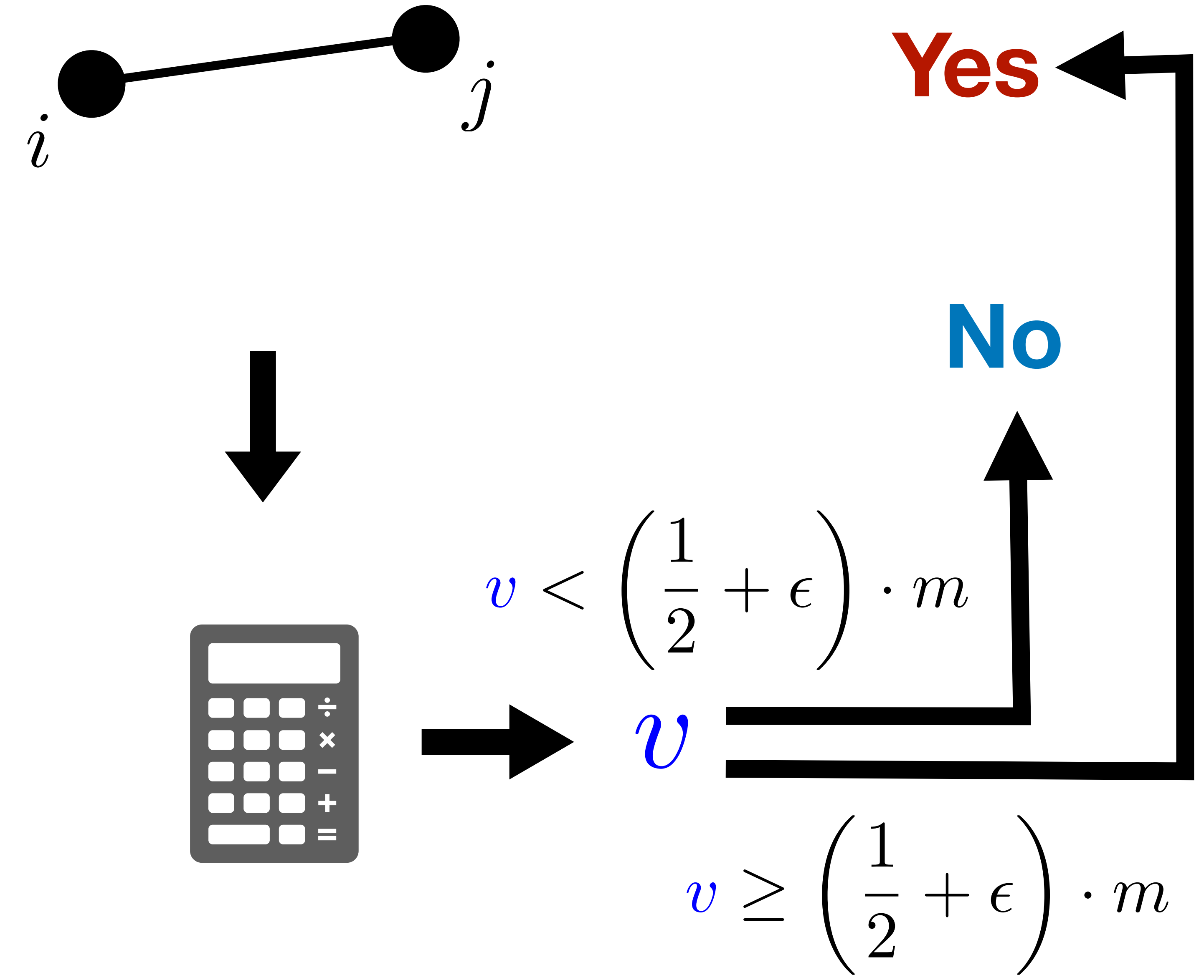




$$\text{val}_{\mathcal{C}} = m$$



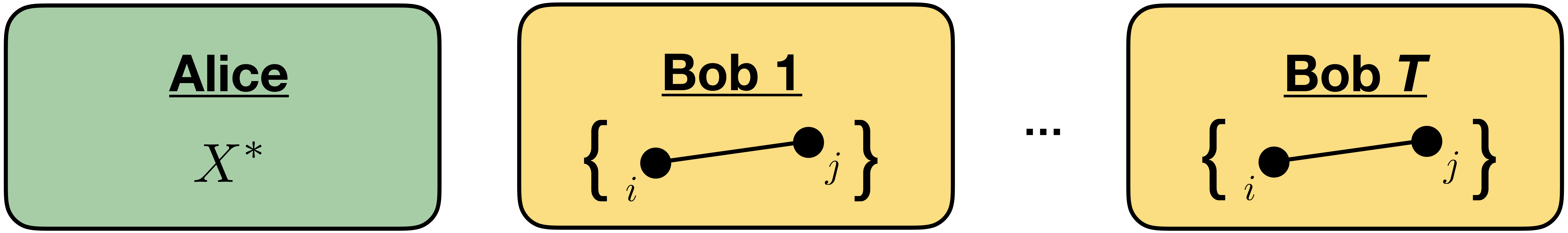
$$\text{val}_{\mathcal{C}} < \left(\frac{1}{2} + o(1)\right) \cdot m$$



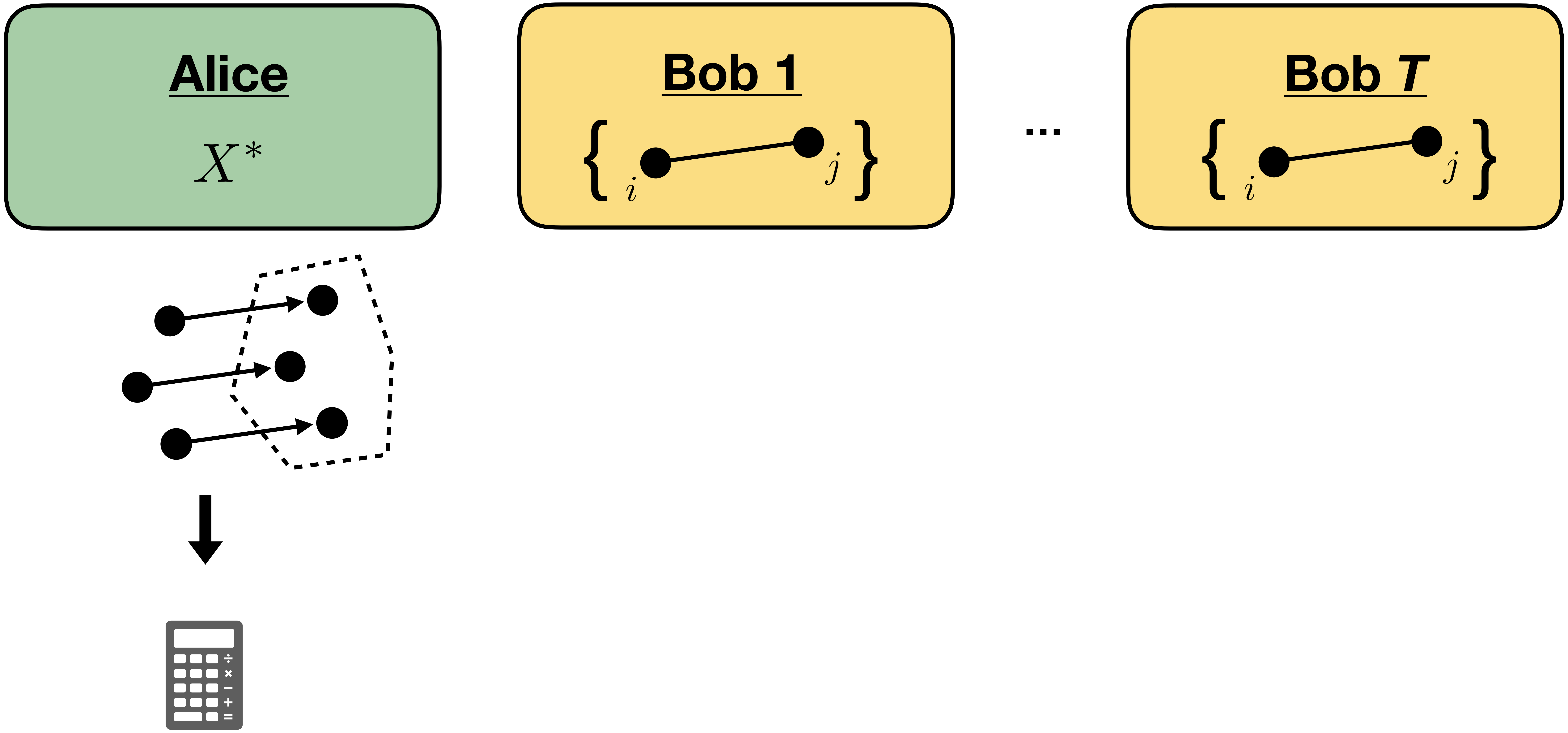
Boolean 2CSP

Λ	α_Λ	Previous	Reference
2XOR	$\frac{1}{2}$ ✓	$\frac{1}{2}$	Trivial
2EOR	$\frac{3}{4}$	$\left[\frac{3}{4}, 1\right]$	Trivial
2AND	$\frac{4}{9}$	$\left[\frac{2}{5}, \frac{1}{2}\right]$	Biased sampling
2OR	$\frac{\sqrt{2}}{2}$	$\left[\frac{1}{2}, 1\right]$	Biased sampling

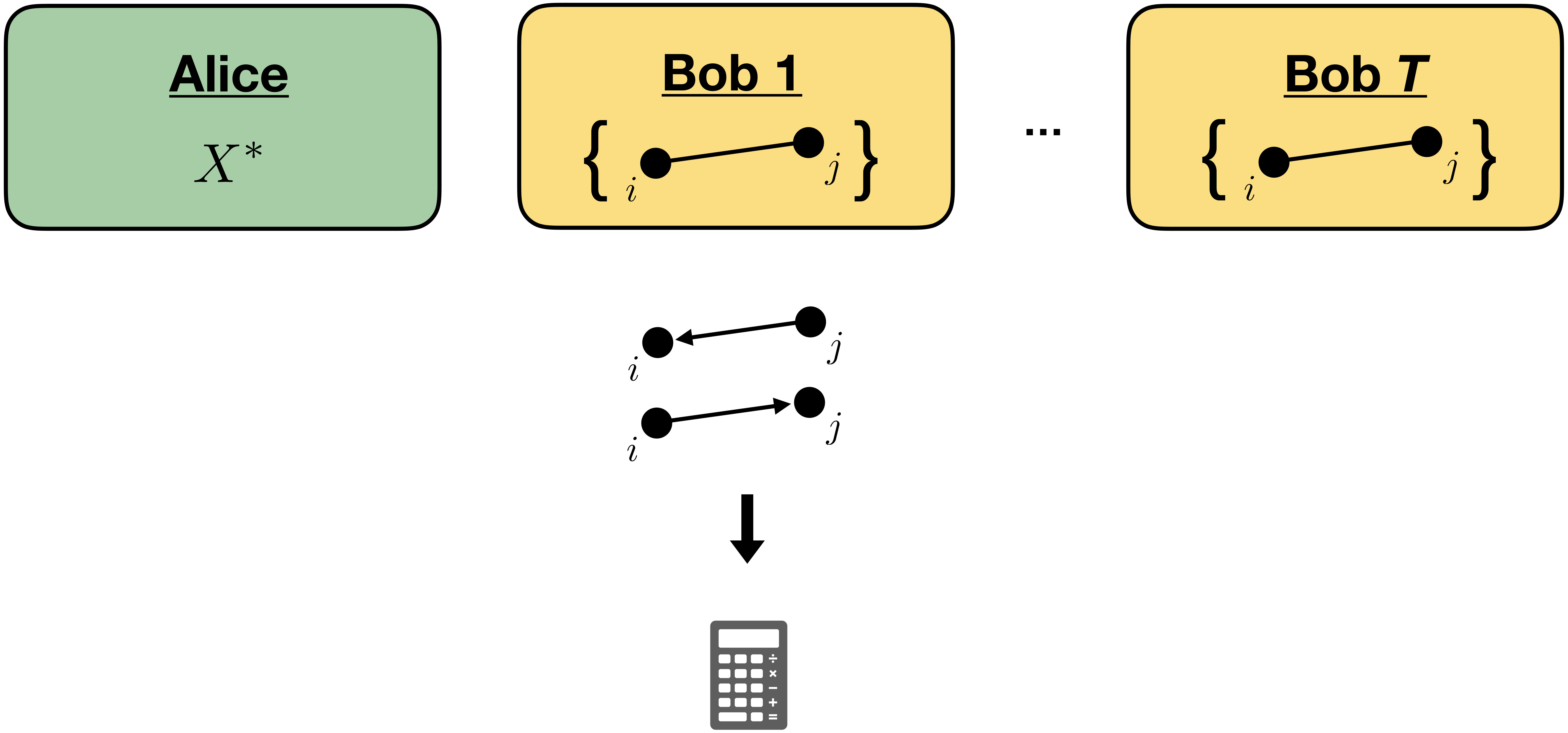
Reducing DBHP to Max-DICUT (Max-2AND)



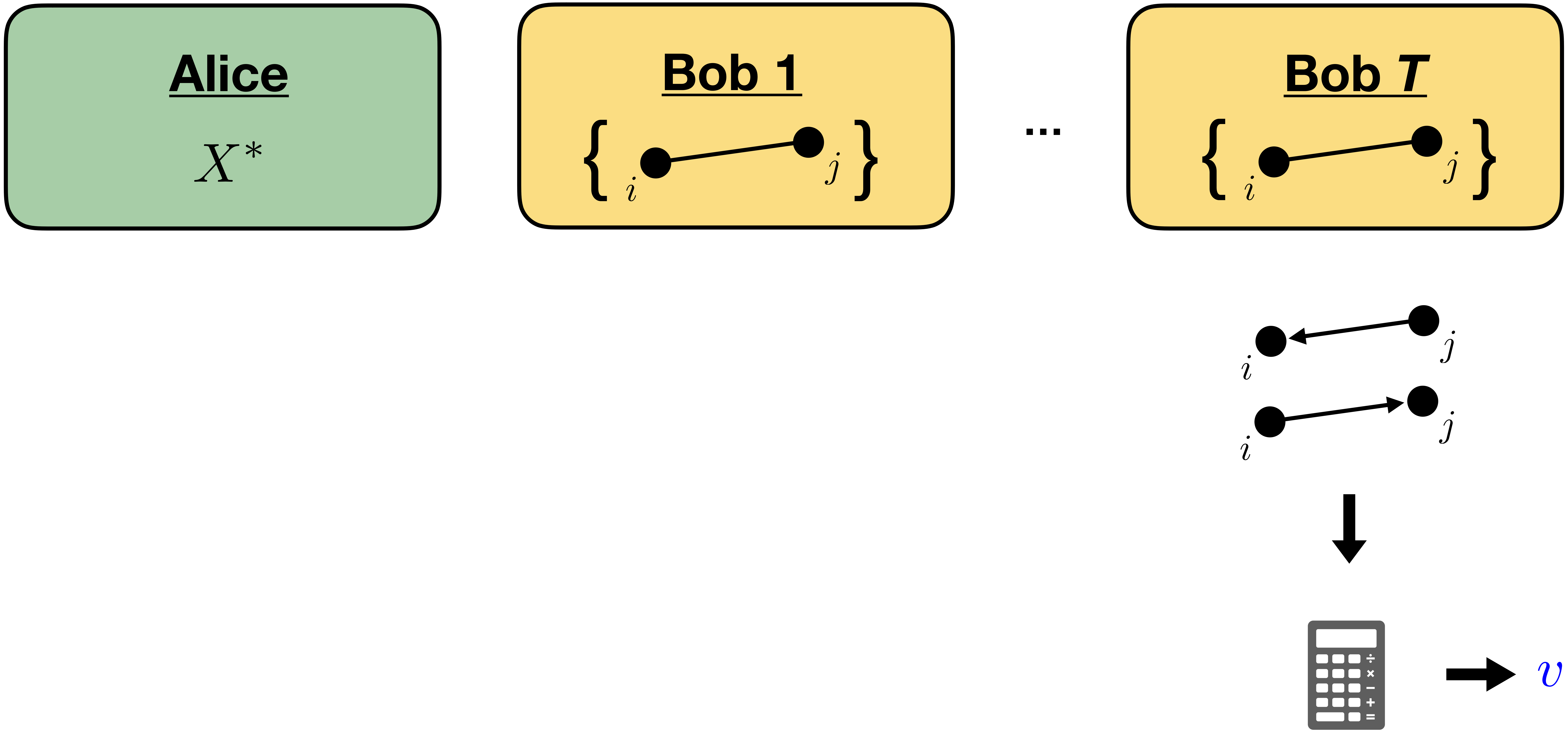
Reducing DBHP to Max-DICUT (Max-2AND)



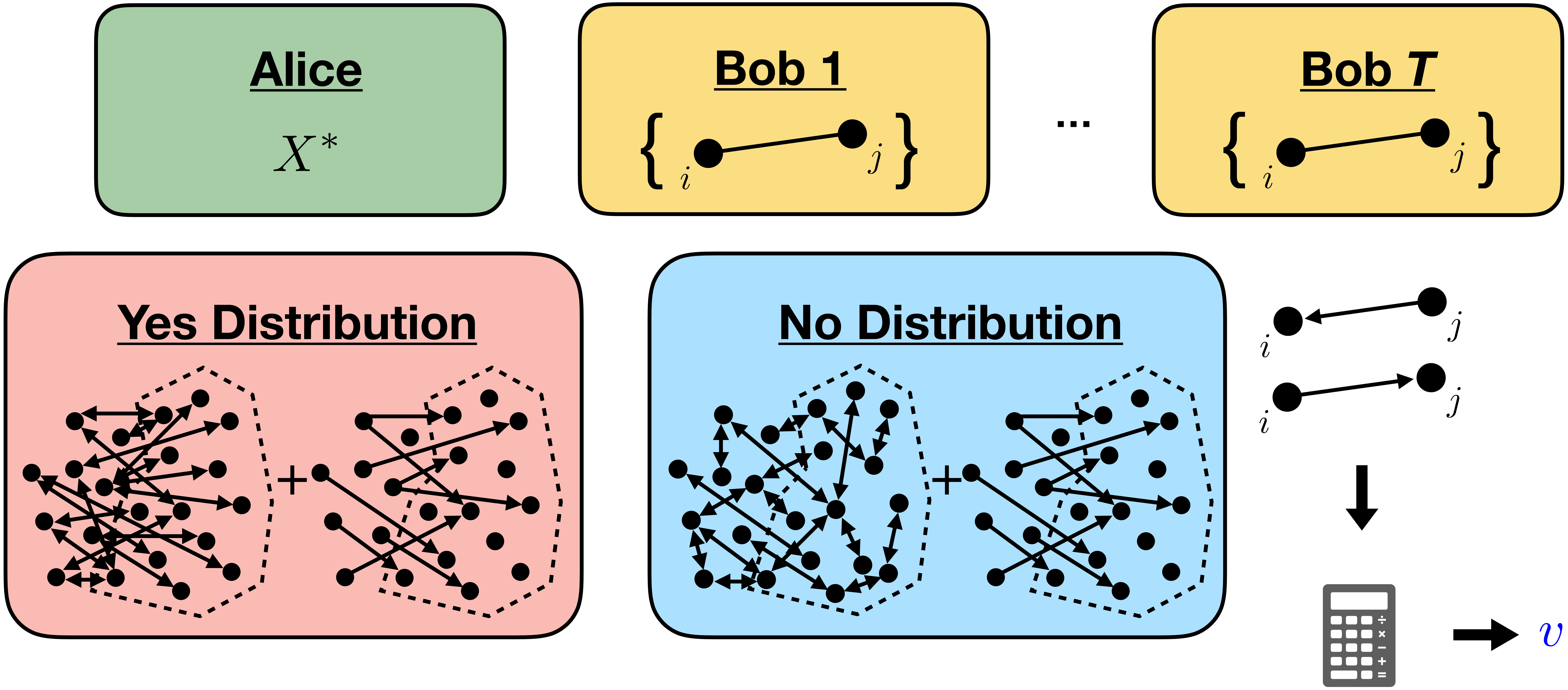
Reducing DBHP to Max-DICUT (Max-2AND)



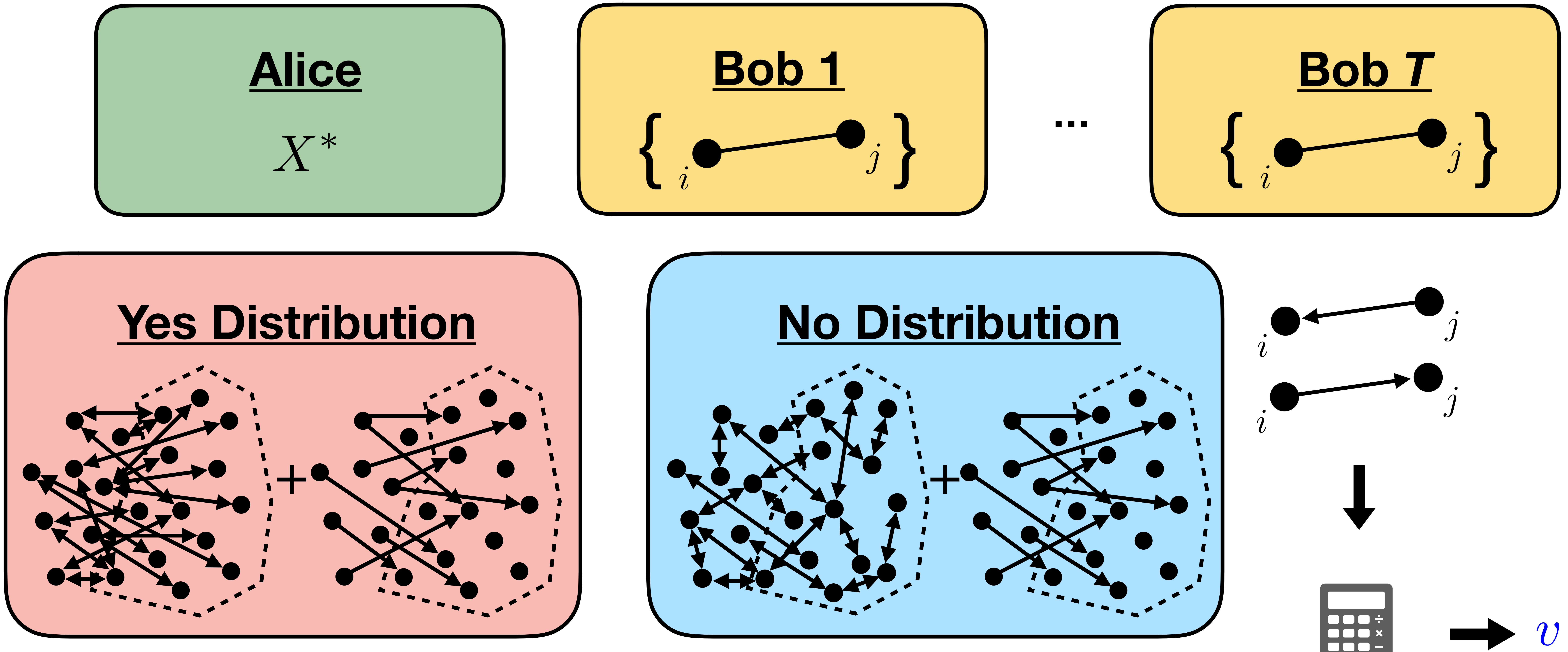
Reducing DBHP to Max-DICUT (Max-2AND)



Reducing DBHP to Max-DICUT (Max-2AND)

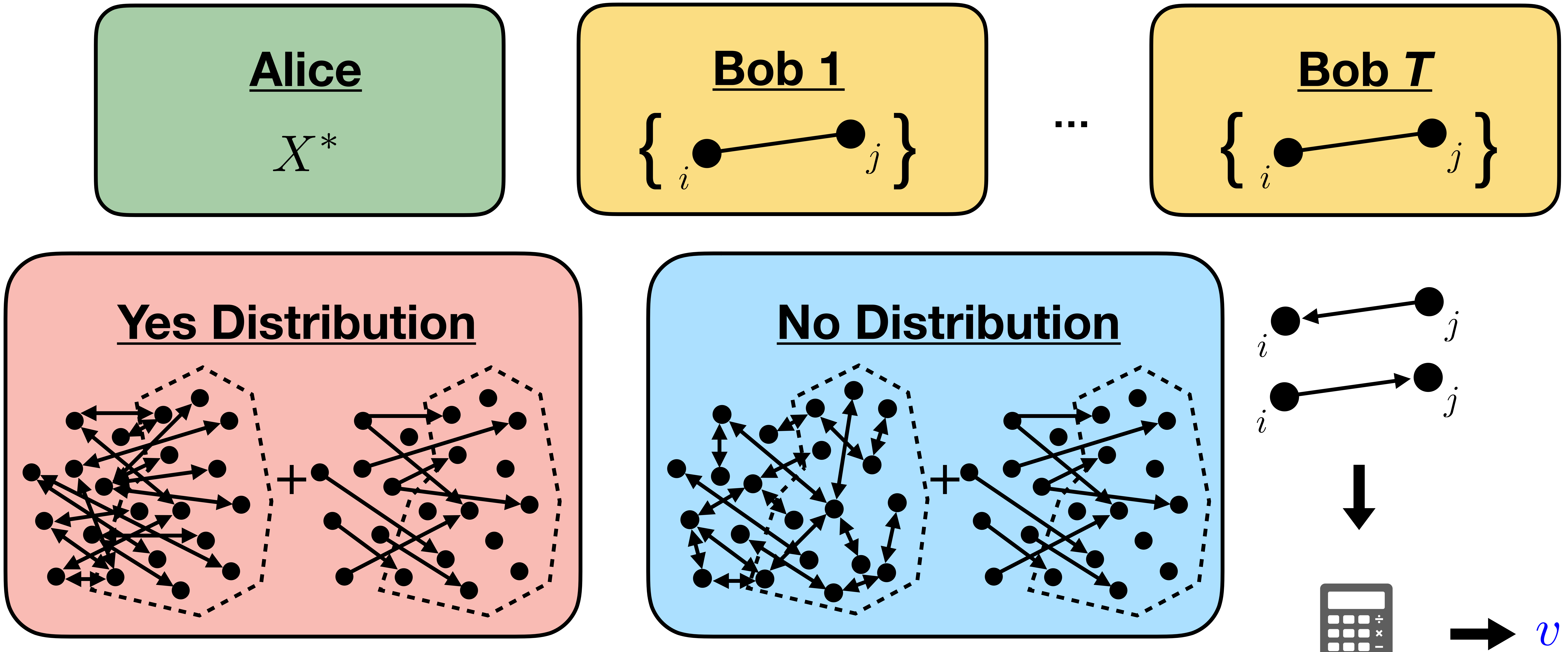


Reducing DBHP to Max-DICUT (Max-2AND)



$$\text{val}_{\mathcal{C}} \geq (1 - o(1)) \left(\frac{m}{2} + \frac{B}{4} \right)$$

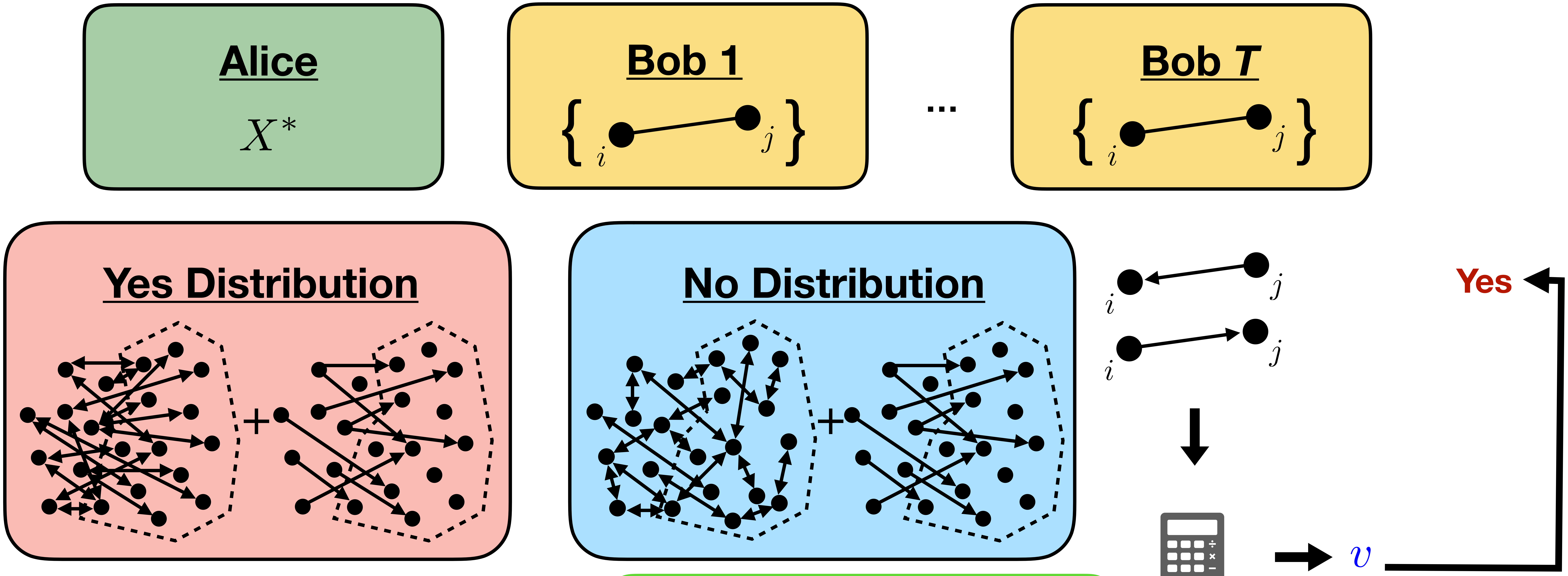
Reducing DBHP to Max-DICUT (Max-2AND)



$$\mathrm{val}_{\mathcal{C}} \geq (1-o(1))\left(\frac{m}{2}+\frac{B}{4}\right)$$

$$\mathrm{val}_{\mathcal{C}} < (1+o(1))\left(\frac{m}{4}+\frac{B^2}{16(m-B)}\right)$$

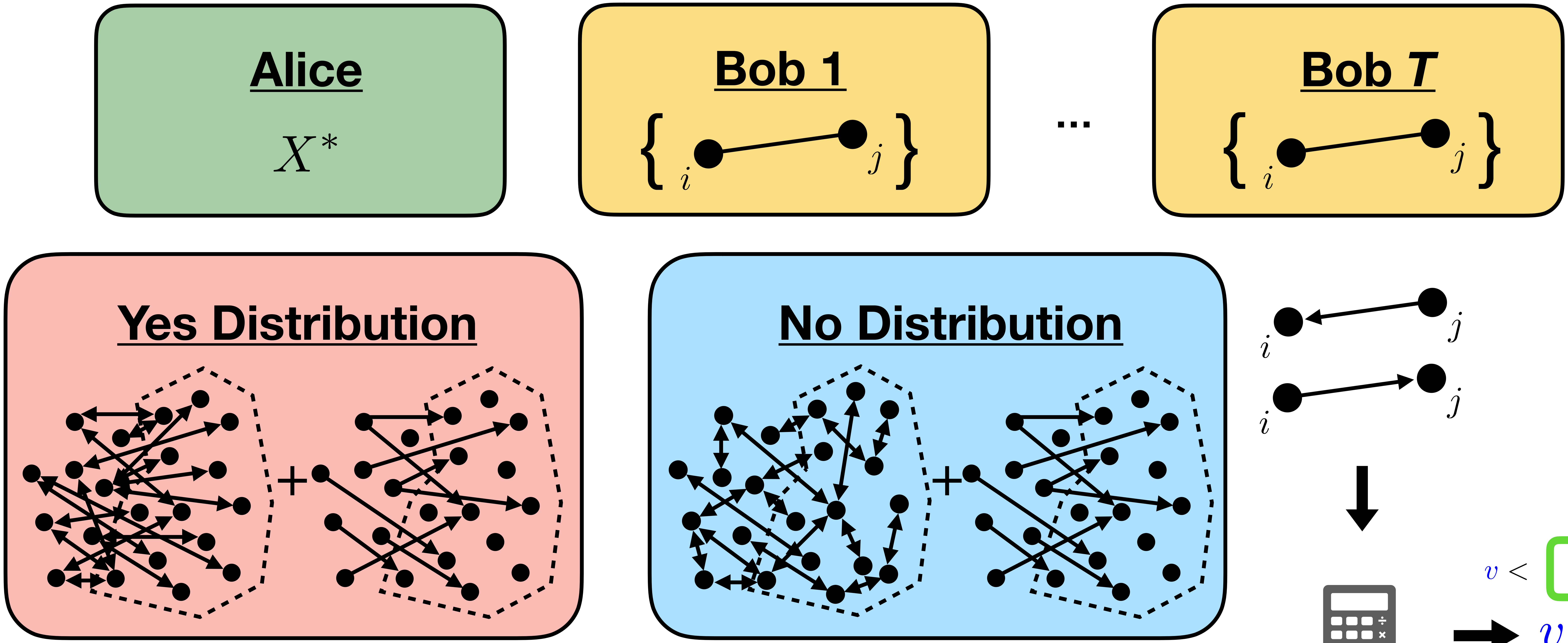
Reducing DBHP to Max-DICUT (Max-2AND)

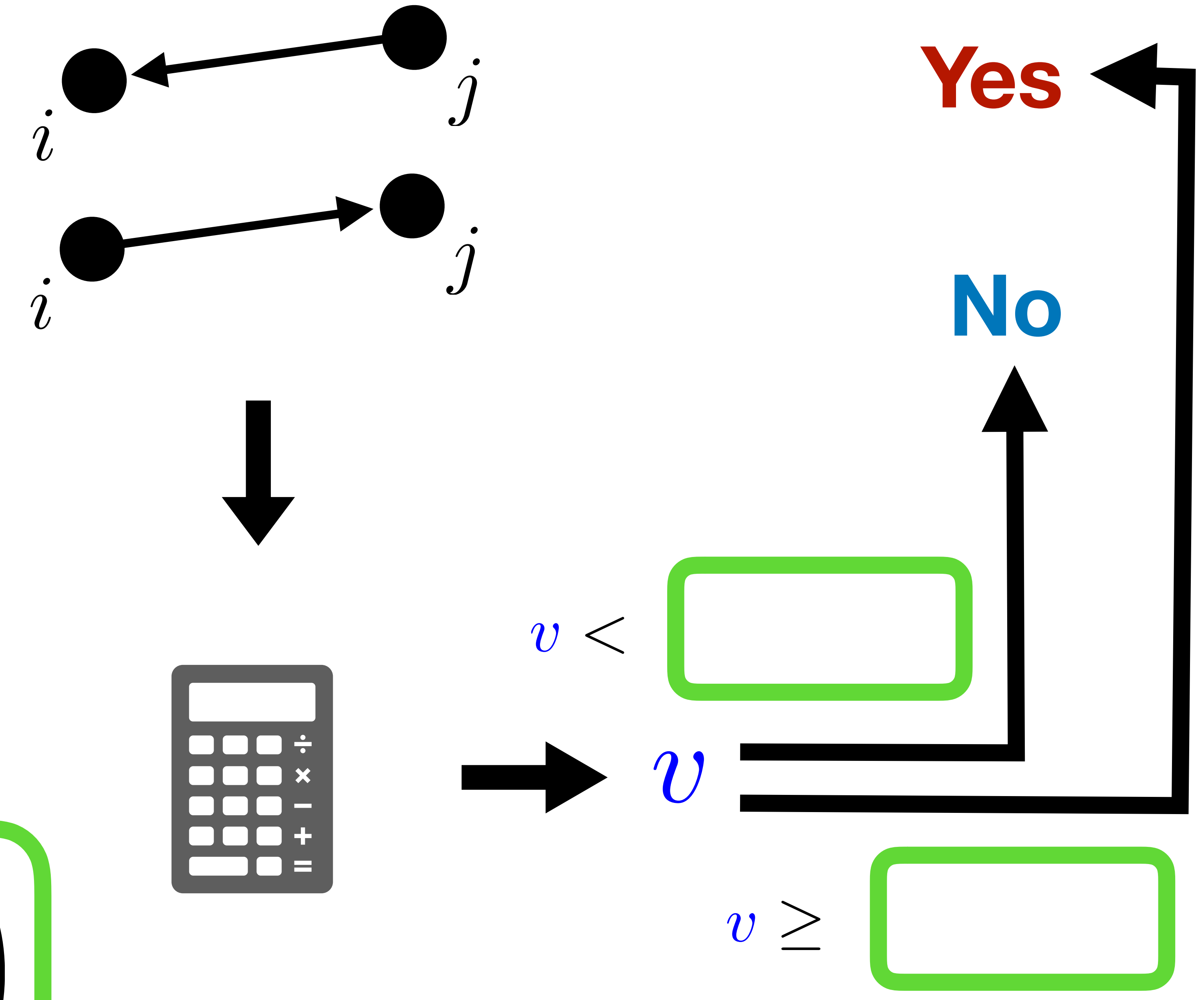


$\mathsf{val}_{\mathcal{C}} \geq (1 - o(1))\left(\frac{m}{2} + \frac{B}{4}\right)$ $\mathsf{val}_{\mathcal{C}} < (1 + o(1))\left(\frac{m}{4} + \frac{B^2}{16(m-B)}\right)$

$v \geq$

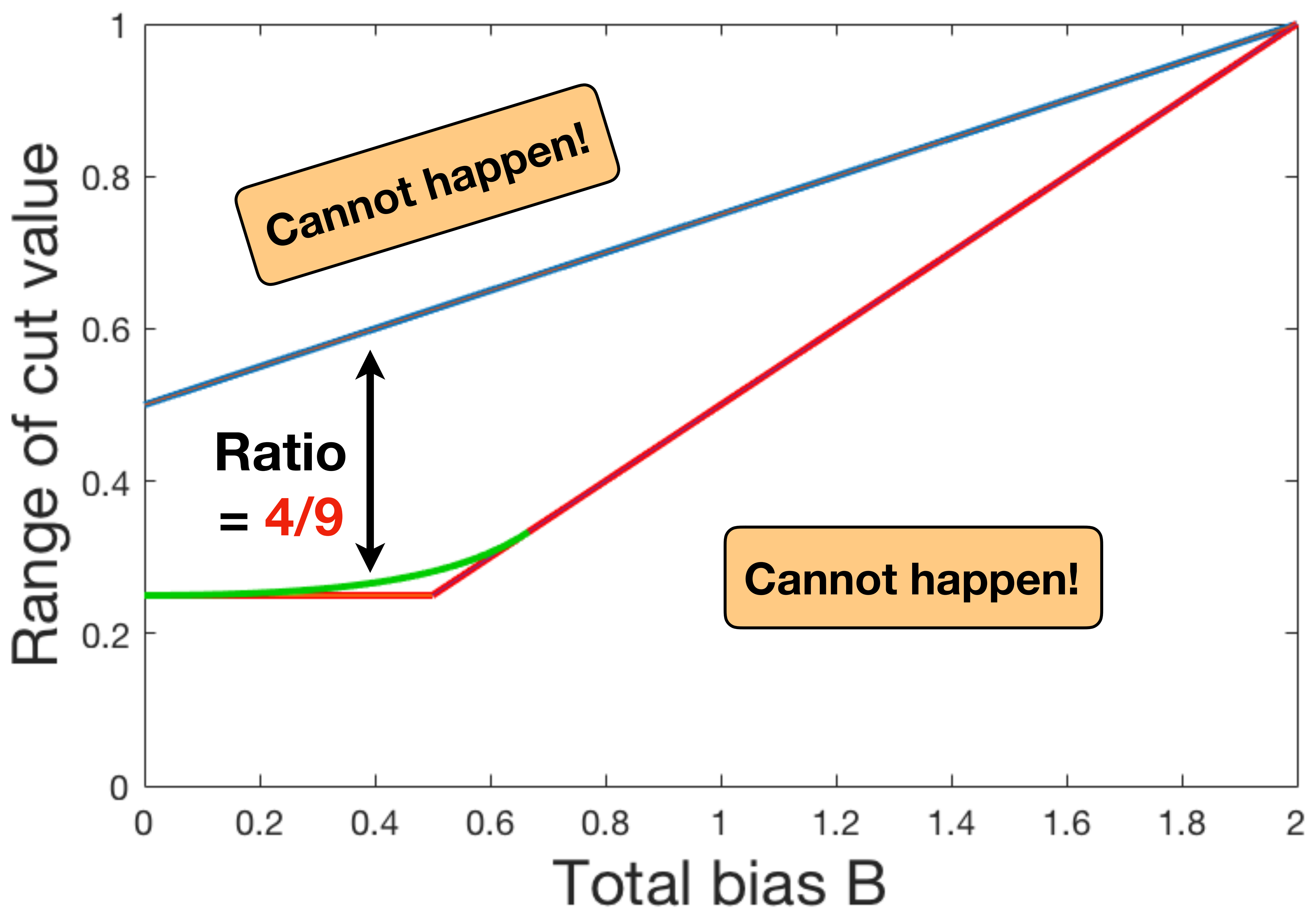
Reducing DBHP to Max-DICUT (Max-2AND)

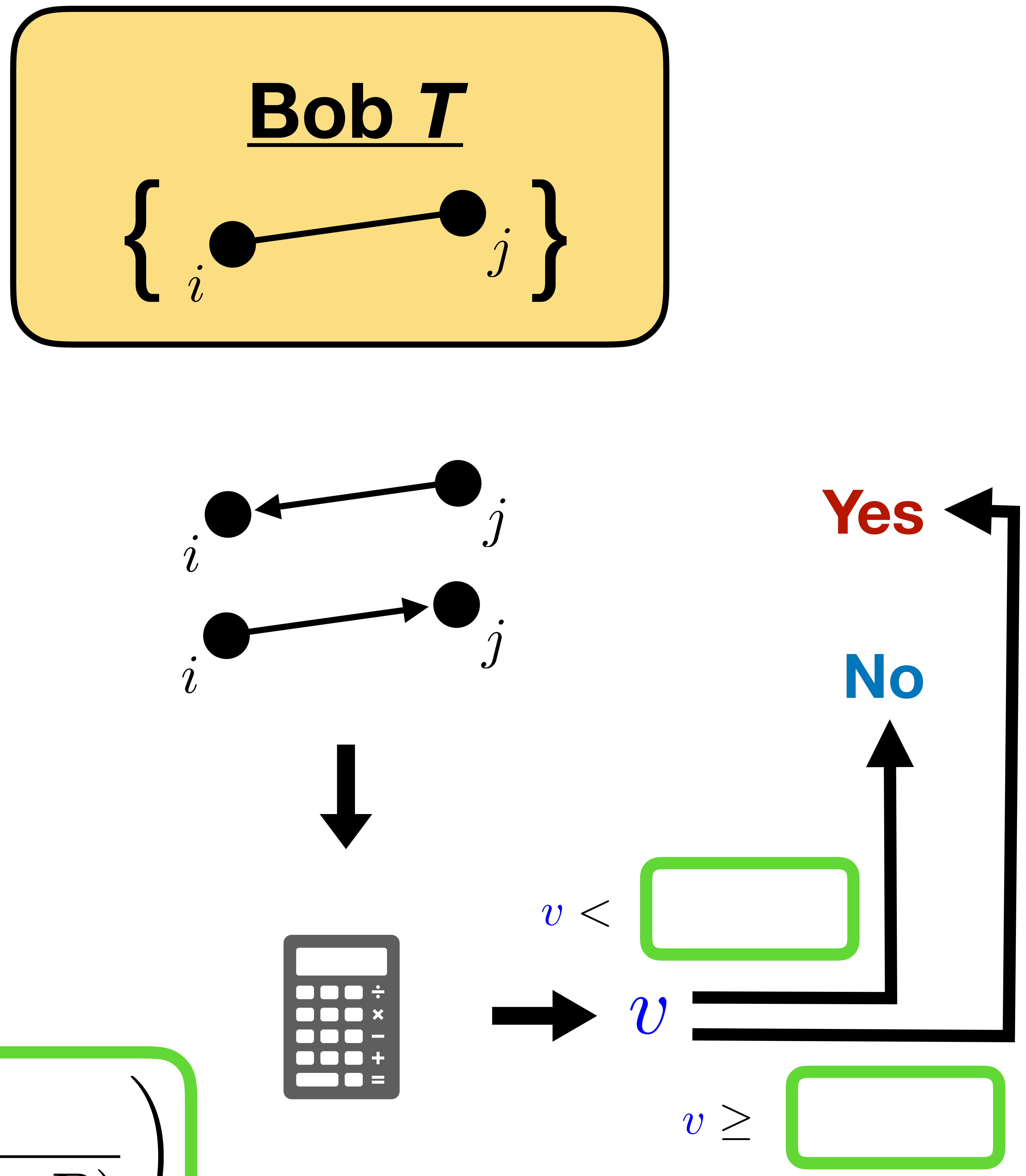




$\text{val}_{\mathcal{C}} \geq (1 - o(1)) \left(\frac{m}{2} + \frac{B}{4} \right)$ $\text{val}_{\mathcal{C}} < (1 + o(1)) \left(\frac{m}{4} + \frac{B^2}{16(m-B)} \right)$

Reducing DBHP to Max-DICUT (Max-2AND)





$$\mathrm{val}_{\mathcal{C}} \geq (1 - o(1)) \left(\frac{m}{2} + \frac{B}{4} \right)$$

$$\mathrm{val}_{\mathcal{C}} < (1 + o(1)) \left(\frac{m}{4} + \frac{B^2}{16(m - B)} \right)$$

Boolean 2CSP

Λ	α_Λ	Previous	Reference
2XOR	$\frac{1}{2}$	$\frac{1}{2}$	Trivial
2EOR	$\frac{3}{4}$	$\left[\frac{3}{4}, 1\right]$	Trivial
2AND	$\frac{4}{9}$	$\left[\frac{2}{5}, \frac{1}{2}\right]$	Biased sampling
2OR	$\frac{\sqrt{2}}{2}$	$\left[\frac{1}{2}, 1\right]$	Biased sampling

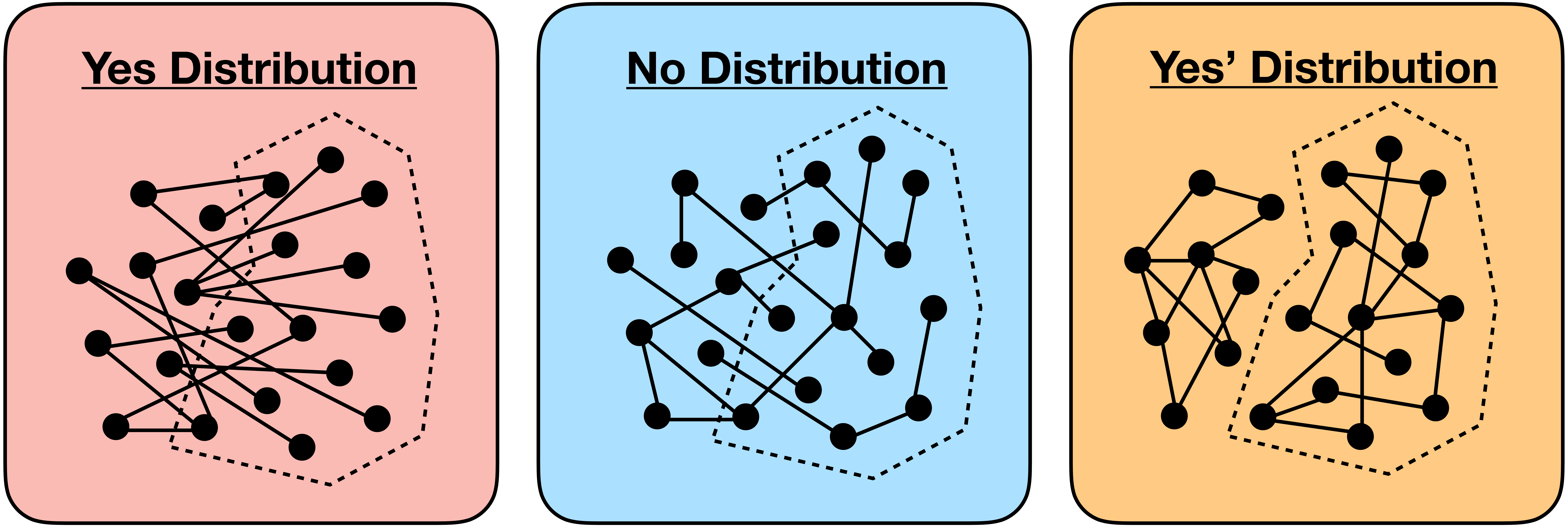
Summary of the DBHP Technique

Summary of the DBHP Technique

- **Step 1**: Identify the gap instances for Max-CSP of type Λ.

Summary of the DBHP Technique

- **Step 1**: Identify the gap instances for Max-CSP of type Λ.
- **Step 2**: Connect one of the three distributions of DBHP to the gap instances.



Conclusion

Conclusion

Conclusion

Theorem

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda$ such that $\forall \epsilon > 0$,

(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Conclusion

Theorem

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda$ such that $\forall \epsilon > 0$,

(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Λ	α_Λ	τ_Λ	Reference
XOR	$\frac{1}{2}$	1	[KK19]
AND	$\frac{4}{9}$	$\frac{1}{2}$	This work
EOR	$\frac{3}{4}$	$\frac{1}{2}$	This work
OR	$\frac{\sqrt{2}}{2}$	$\frac{1}{2}$	This work

Conclusion

Theorem

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda$ such that $\forall \epsilon > 0$,

(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Λ	α_Λ	τ_Λ	Reference
XOR	$\frac{1}{2}$	1	[KK19]
AND	$\frac{4}{9}$	$\frac{1}{2}$	This work
EOR	$\frac{3}{4}$	$\frac{1}{2}$	This work
OR	$\frac{\sqrt{2}}{2}$	$\frac{1}{2}$	This work

Can be extended to Max k-SAT!

Conclusion

Theorem

For any boolean 2CSP of type Λ, there exist $\alpha_\Lambda, \tau_\Lambda$ such that $\forall \epsilon > 0$,

(i) there's a $(\alpha_\Lambda - \epsilon)$-approx. in $O(\log n)$ space and

(ii) no $(\alpha_\Lambda + \epsilon)$-approx. in $\Omega(n^{\tau_\Lambda})$ space.

Λ	α_Λ	τ_Λ	Reference
XOR	$\frac{1}{2}$	1	[KK19]
AND	$\frac{4}{9}$	$\frac{1}{2}$	This work
EOR	$\frac{3}{4}$	$\frac{1}{2}$	This work
OR	$\frac{\sqrt{2}}{2}$	$\frac{1}{2}$	This work

Local random sampling is optimal!

Future Directions

Future Directions

- **(Short term)** Improve the $\Omega(\sqrt{n})$ space lower bounds.

Future Directions

- **(Short term)** Improve the $\Omega(\sqrt{n})$ space lower bounds.
  - Maybe need new communication lower bound?

Future Directions

- **(Short term)** Improve the $\Omega(\sqrt{n})$ space lower bounds.
  - Maybe need new communication lower bound?
- **(Mid term)** Investigate the Max-2CSP with larger alphabet set.

Future Directions

- **(Short term)** Improve the $\Omega(\sqrt{n})$ space lower bounds.
  - ✦ Maybe need new communication lower bound?
- **(Mid term)** Investigate the Max-2CSP with larger alphabet set.
  - ✦ [Guruswami-Tao 19]: the ratio of UG on size p alphabet set is 1/p.

Future Directions

- **(Short term)** Improve the $\Omega(\sqrt{n})$ space lower bounds.
  - Maybe need new communication lower bound?
- (**Mid term)** Investigate the Max-2CSP with larger alphabet set.
  - [Guruswami-Tao 19]: the ratio of UG on size p alphabet set is 1/p.
- **(Mid term)** Investigate boolean Max-CSP with larger arity.

Future Directions

- **(Short term)** Improve the $\Omega(\sqrt{n})$ space lower bounds.
  - Maybe need new communication lower bound?
- **(Mid term)** Investigate the Max-2CSP with larger alphabet set.
  - [Guruswami-Tao 19]: the ratio of UG on size p alphabet set is 1/p.
- **(Mid term)** Investigate boolean Max-CSP with larger arity.
  - In the standard model + Unique Games Conjecture, Max-3CSP has ratio 5/8 and Max-kCSP has ratio .

Future Directions

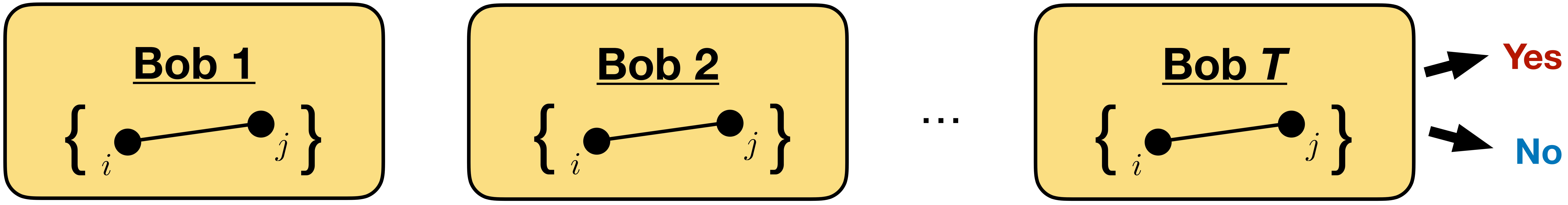
- **(Short term)** Improve the $\Omega(\sqrt{n})$ space lower bounds.
  - ✦ Maybe need new communication lower bound?
- **(Mid term)** Investigate the Max-2CSP with larger alphabet set.
  - ✦ [Guruswami-Tao 19]: the ratio of UG on size p alphabet set is 1/p.
- **(Mid term)** Investigate boolean Max-CSP with larger arity.
  - ✦ In the standard model + Unique Games Conjecture, Max-3CSP has ratio 5/8 and Max-kCSP has ratio $\Theta(2^k/k)$.
- **(Long term)** The limit of local random sampling in streaming Max-CSP?

Future Directions

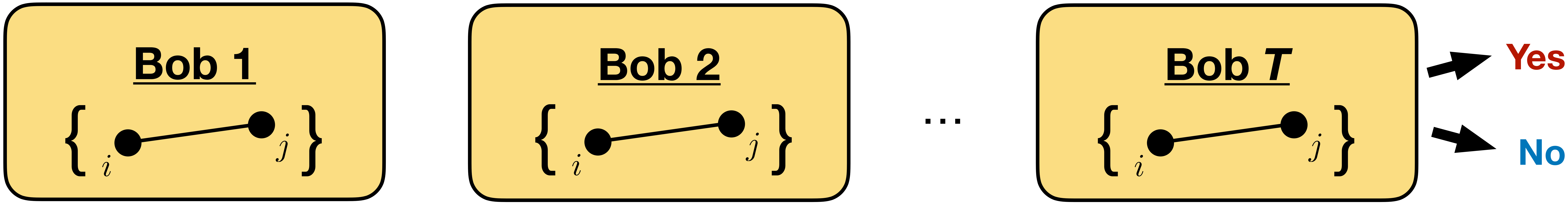
- **(Short term)** Improve the $\Omega(\sqrt{n})$ space lower bounds.
  - Maybe need new communication lower bound?
- **(Mid term)** Investigate the Max-2CSP with larger alphabet set.
  - [Guruswami-Tao 19]: the ratio of UG on size p alphabet set is 1/p.
- **(Mid term)** Investigate boolean Max-CSP with larger arity.
  - In the standard model + Unique Games Conjecture, Max-3CSP has ratio 5/8 and Max-kCSP has ratio $\Theta(2^k/k)$.
- **(Long term)** The limit of local random sampling in streaming Max-CSP?

Thanks for your attention, questions?

Reducing DBHP to Max-2EOR

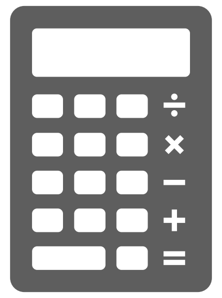


Reducing DBHP to Max-2EOR

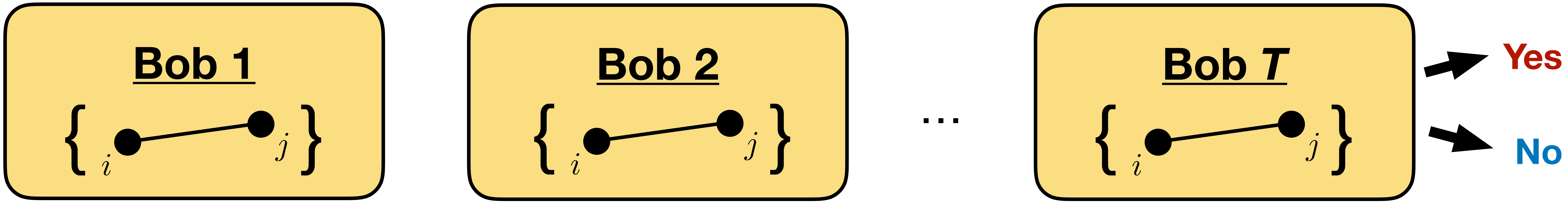


$$x_i \vee x_j$$

$$\neg x_i \vee \neg x_j$$

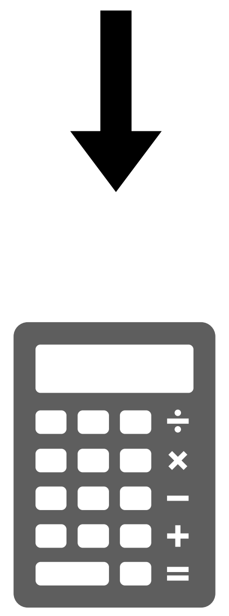


Reducing DBHP to Max-2EOR

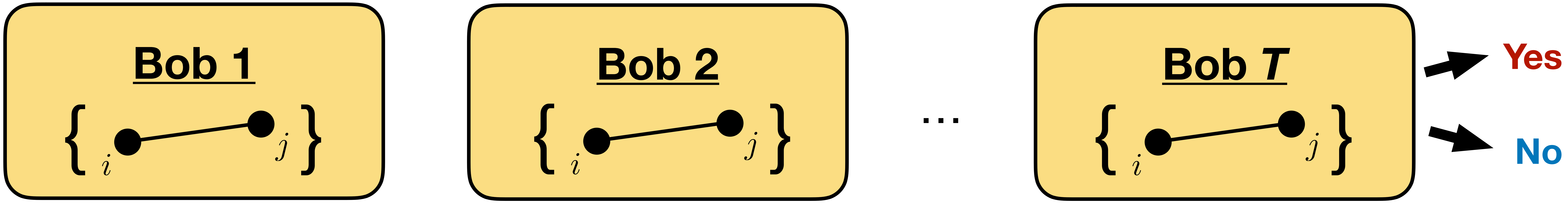


$$x_i \vee x_j$$

$$\neg x_i \vee \neg x_j$$

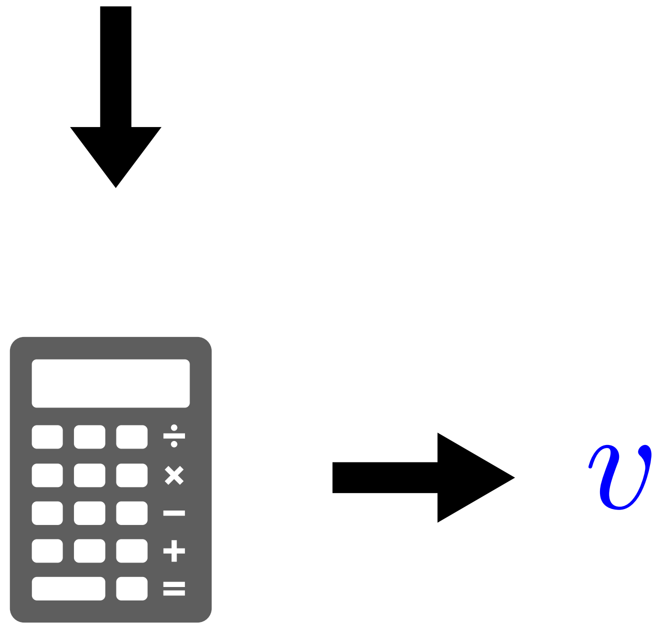


Reducing DBHP to Max-2EOR

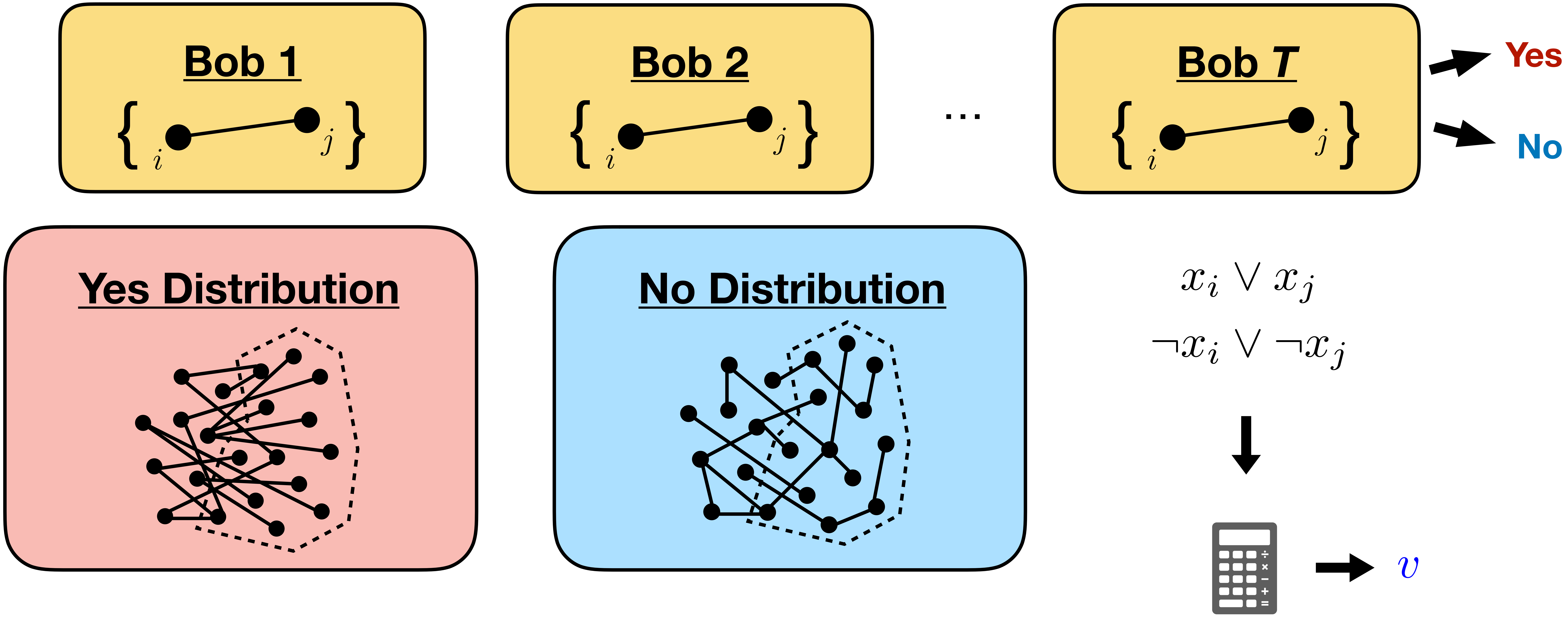


$$x_i \vee x_j$$

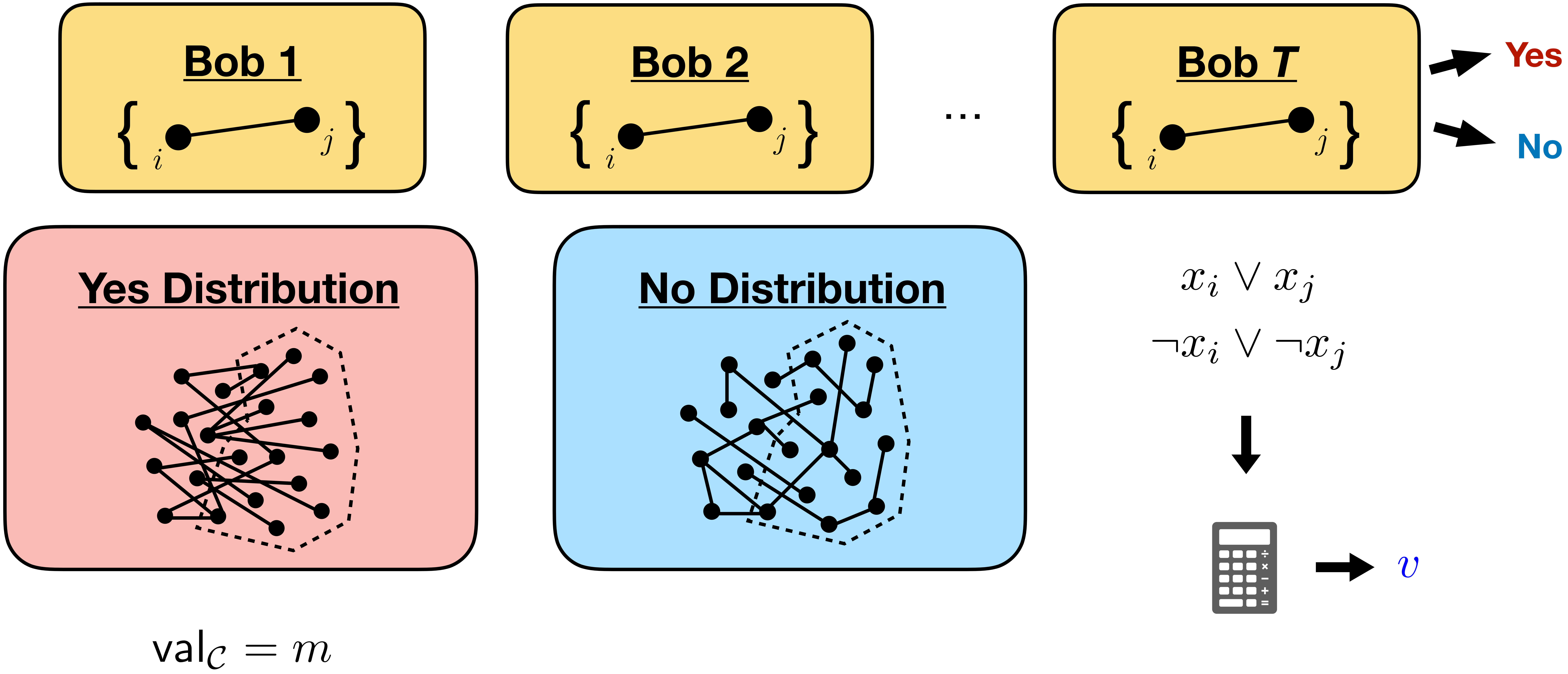
$$\neg x_i \vee \neg x_j$$



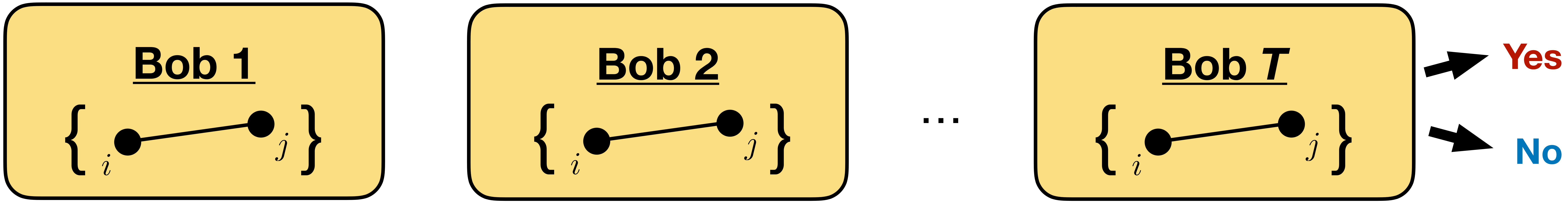
Reducing DBHP to Max-2EOR



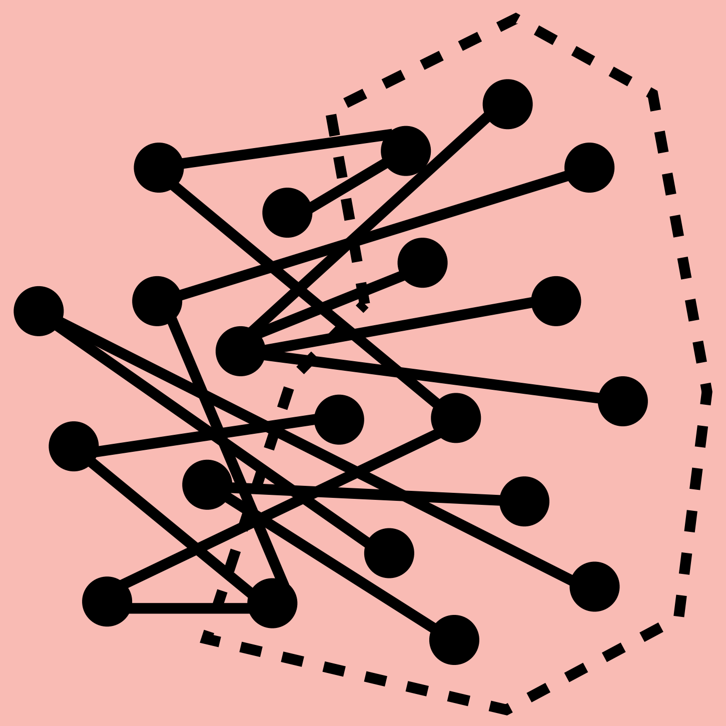
Reducing DBHP to Max-2EOR



Reducing DBHP to Max-2EOR

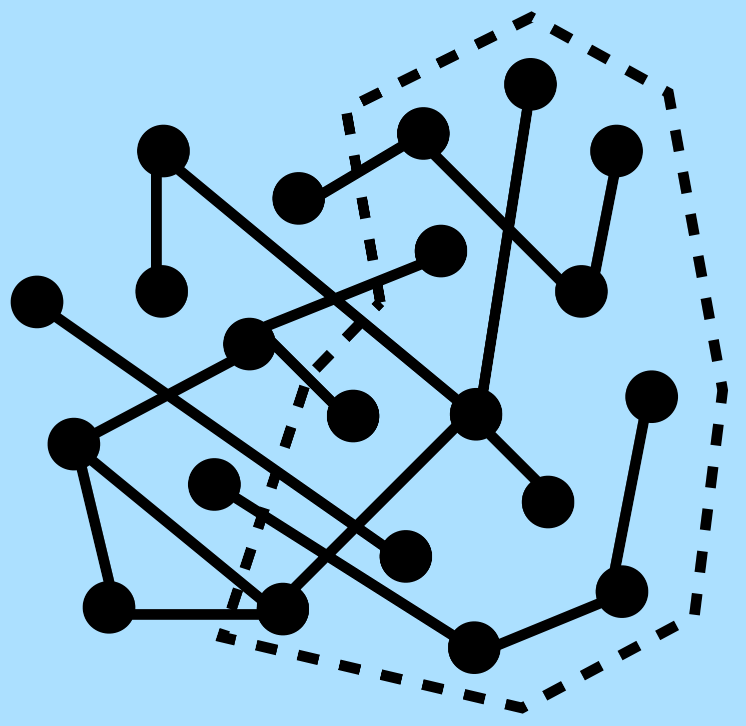


Yes Distribution



$\mathsf{val}_{\mathcal{C}} = m$

No Distribution



$\mathsf{val}_{\mathcal{C}} < \left(\frac{3}{4} + o(1)\right) \cdot m$

$x_i \vee x_j$

$\neg x_i \vee \neg x_j$

$\downarrow$

$\rightarrow v$

Reducing DBHP to Max-2EOR

Reducing DBHP to Max-2EOR

Max-2OR

Max-2OR

- **Observation**: The expected value of 1-clause < 2-clause!

Max-2OR

- **Observation**: The expected value of 1-clause < 2-clause!
  - 1-clause: $\mathbb{E}[x] = 1/2$; 2-clause: $\mathbb{E}[x \vee y] = 3/4$.

Max-2OR

- **Observation**: The expected value of 1-clause < 2-clause!
  - 1-clause: $\mathbb{E}[x] = 1/2$; 2-clause: $\mathbb{E}[x \vee y] = 3/4$.
- Thus, the 3/4-approx. for Max-2EOR does not work 😢

Max-2OR

- **Observation**: The expected value of 1-clause < 2-clause!
  - 1-clause: $\mathbb{E}[x] = 1/2$; 2-clause: $\mathbb{E}[x \vee y] = 3/4$.
- Thus, the 3/4-approx. for Max-2EOR does not work 😢
- **Biased sampling**:

Max-2OR

- **Observation**: The expected value of 1-clause < 2-clause!
  - 1-clause: $\mathbb{E}[x] = 1/2$; 2-clause: $\mathbb{E}[x \vee y] = 3/4$.
- Thus, the 3/4-approx. for Max-2EOR does not work 😢
- **Biased sampling**:
  - Random sample with the **biases** of 1-clause and 2-clause.

Max-2OR

- **Observation**: The expected value of 1-clause < 2-clause!
  - 1-clause: $\mathbb{E}[x] = 1/2$; 2-clause: $\mathbb{E}[x \vee y] = 3/4$.
- Thus, the 3/4-approx. for Max-2EOR does not work 😢
- **Biased sampling**:
  - Random sample with the **biases** of 1-clause and 2-clause.
  - This gives $\sqrt{2}/2$-approx. 😳

Max-2OR

- **Observation**: The expected value of 1-clause < 2-clause!
  - 1-clause: $\mathbb{E}[x] = 1/2$; 2-clause: $\mathbb{E}[x \vee y] = 3/4$.
- Thus, the 3/4-approx. for Max-2EOR does not work 😢
- **Biased sampling**:
  - Random sample with the **biases** of 1-clause and 2-clause.
  - This gives $\sqrt{2}/2$-approx. 😳
- **Gap instances**:

Max-2OR

- **Observation**: The expected value of 1-clause < 2-clause!
  - 1-clause: $\mathbb{E}[x] = 1/2$; 2-clause: $\mathbb{E}[x \vee y] = 3/4$.
- Thus, the 3/4-approx. for Max-2EOR does not work 😢
- **Biased sampling**:
  - Random sample with the **biases** of 1-clause and 2-clause.
  - This gives $\sqrt{2}/2$-approx. 😳
- **Gap instances**:

$$\{x_i\} \ \{\neg x_i \vee \neg x_j\}$$

$$\mathsf{val}_{\mathcal{C}} < \left(\frac{\sqrt{2}}{2} + o(1)\right) \cdot m$$

Max-2OR

- **Observation**: The expected value of 1-clause < 2-clause!
  - 1-clause: $\mathbb{E}[x] = 1/2$; 2-clause: $\mathbb{E}[x \vee y] = 3/4$.
- Thus, the 3/4-approx. for Max-2EOR does not work 😢
- **Biased sampling**:
  - Random sample with the **biases** of 1-clause and 2-clause.
  - This gives $\sqrt{2}/2$-approx. 😳
- **Gap instances**:

$$\{x_i\} \ \{\neg x_i \vee \neg x_j\}$$

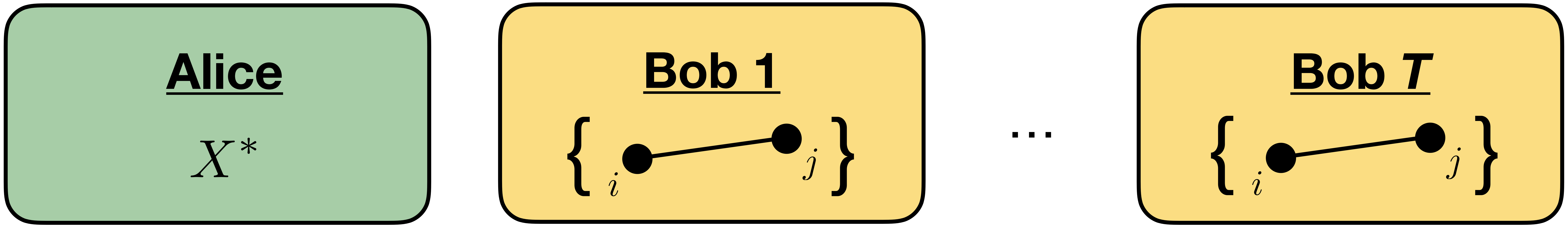
$$\mathsf{val}_{\mathcal{C}} < \left(\frac{\sqrt{2}}{2} + o(1)\right) \cdot m$$

$$\{x_i\} \ \{\neg y_i \vee \neg y_j\}$$

$$\mathsf{val}_{\mathcal{C}} = m$$

Reducing DBHP to Max-2OR

Reducing DBHP to Max-2OR

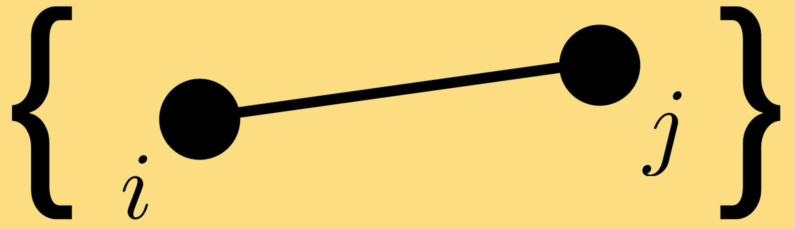


Reducing DBHP to Max-2OR

Alice

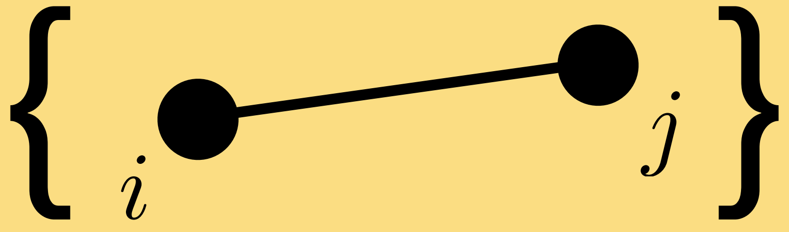
X^*

Bob 1



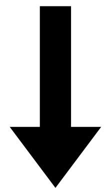
…

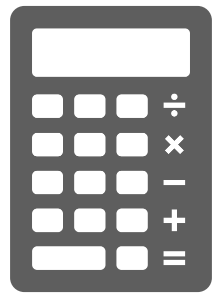
Bob *T*



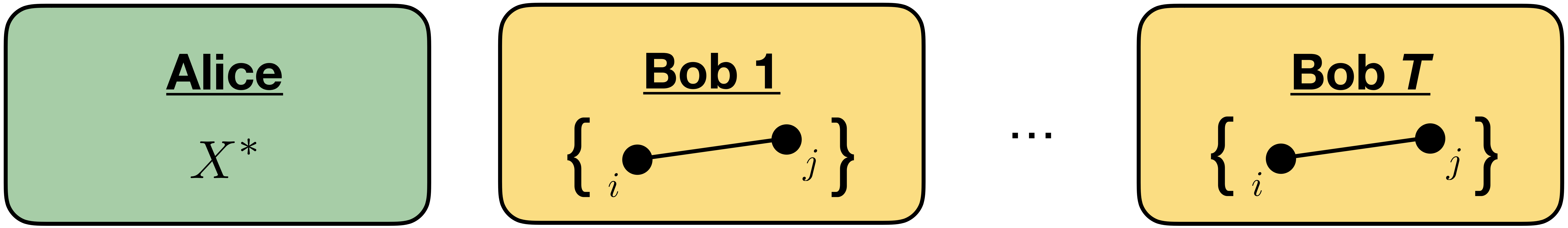
x_i

$\neg x_j$



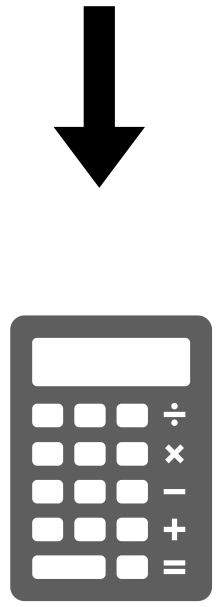


Reducing DBHP to Max-2OR

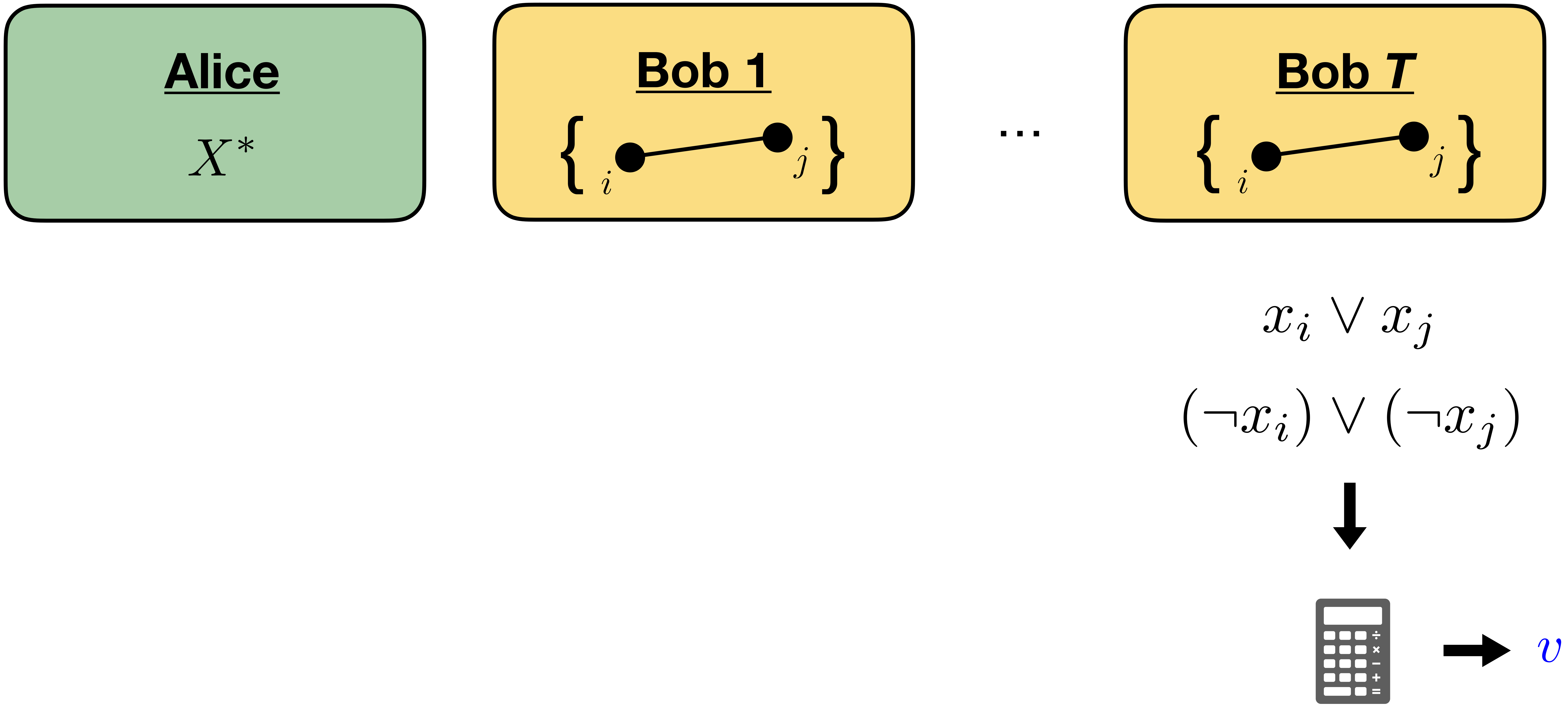


$$x_i \vee x_j$$

$$(\neg x_i) \vee (\neg x_j)$$



Reducing DBHP to Max-2OR



Reducing DBHP to Max-2OR

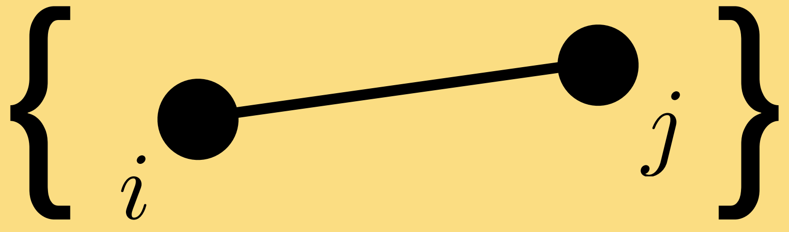
Alice

X^*

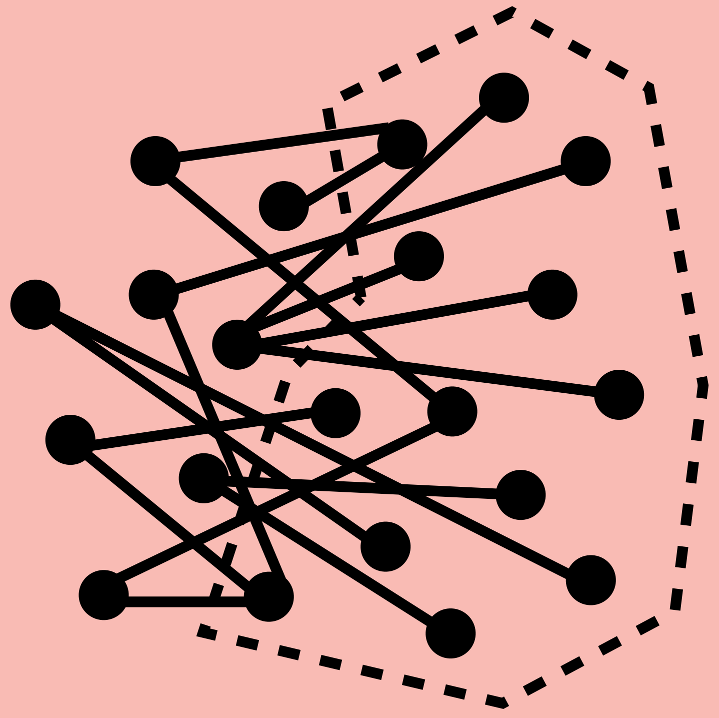
Bob 1

…

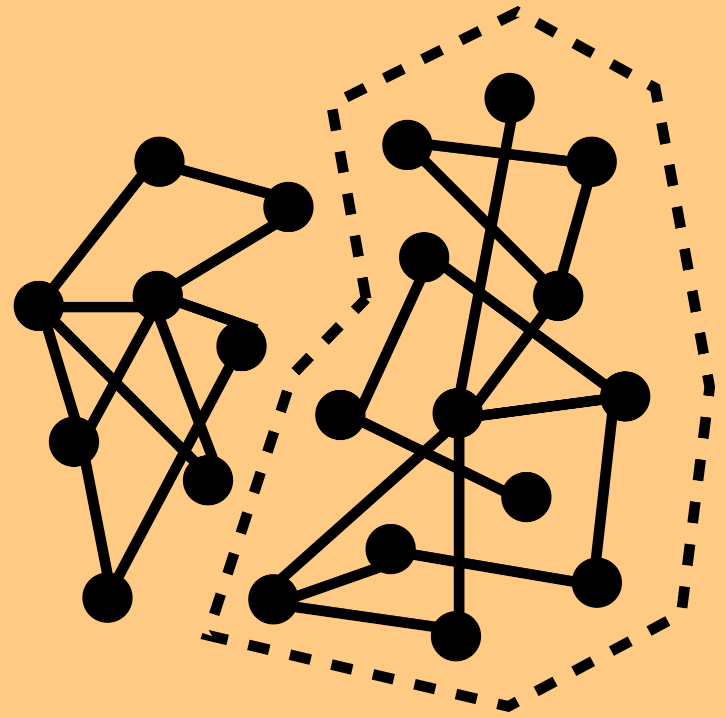
Bob T



Yes Distribution



Yes' Distribution



$x_i \vee x_j$

$(\neg x_i) \vee (\neg x_j)$

↓

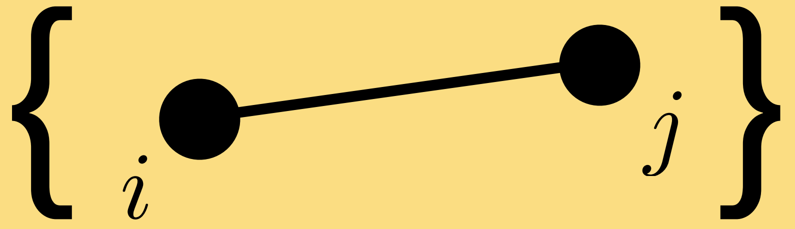
→ v

Reducing DBHP to Max-2OR

Alice

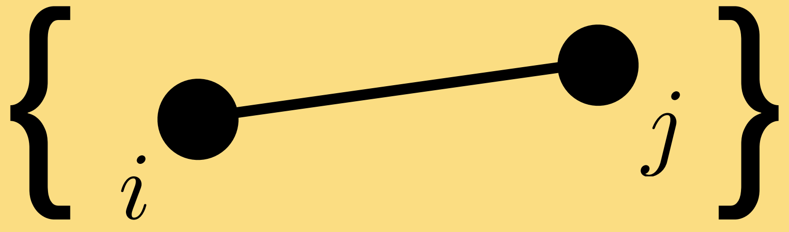
X^*

Bob 1

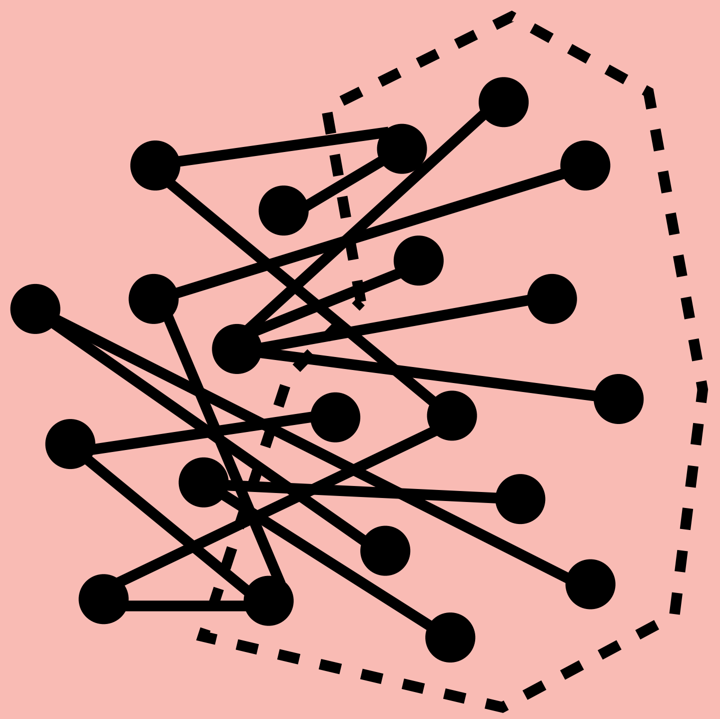


...

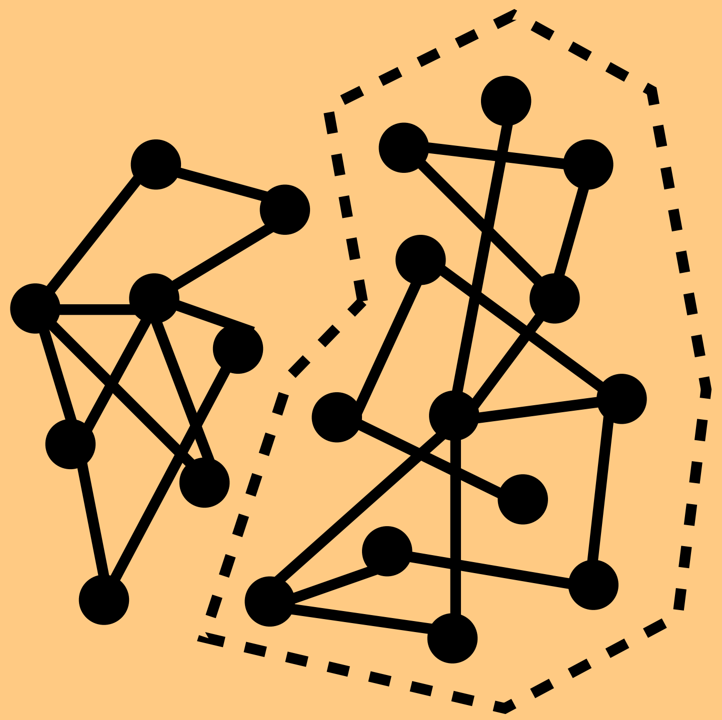
Bob *T*



Yes Distribution



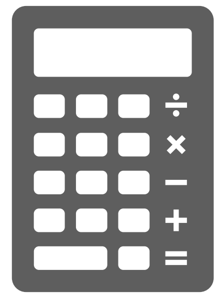
Yes' Distribution



$x_i \vee x_j$

$(\neg x_i) \vee (\neg x_j)$

↓



→ v

$\mathrm{val}_{\mathcal{C}} < \left(\frac{\sqrt{2}}{2} + o(1)\right) \cdot m$

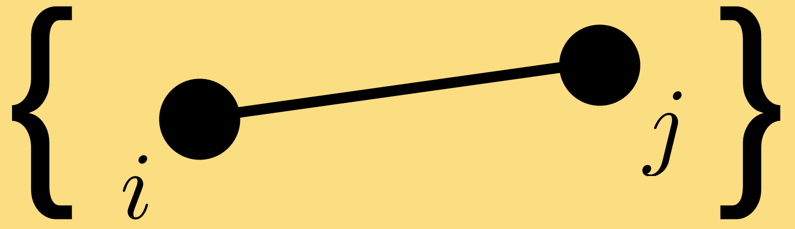
$\mathrm{val}_{\mathcal{C}} = m$

Reducing DBHP to Max-2OR

Alice

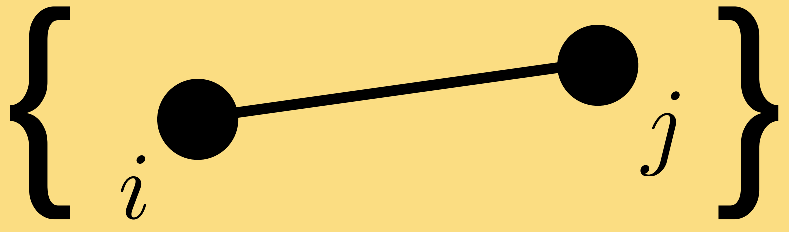
X^*

Bob 1

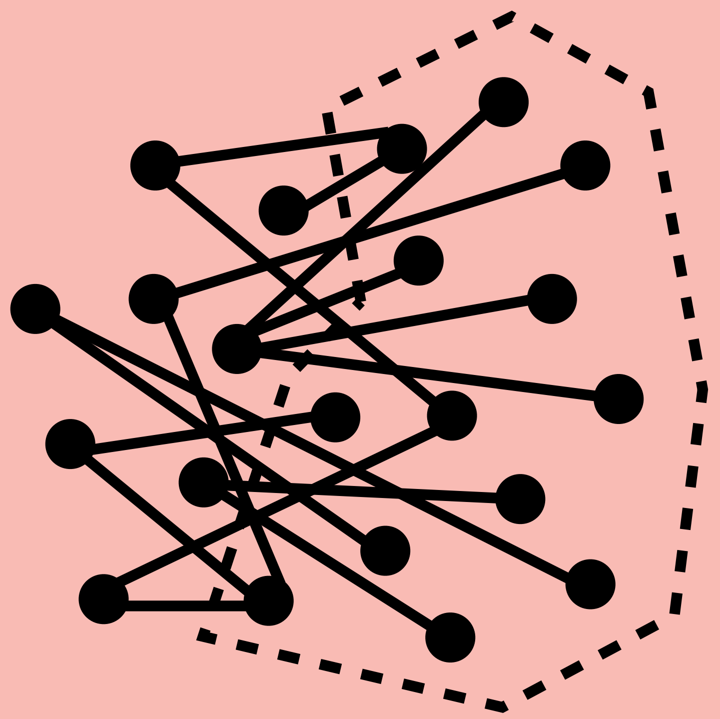


…

Bob *T*

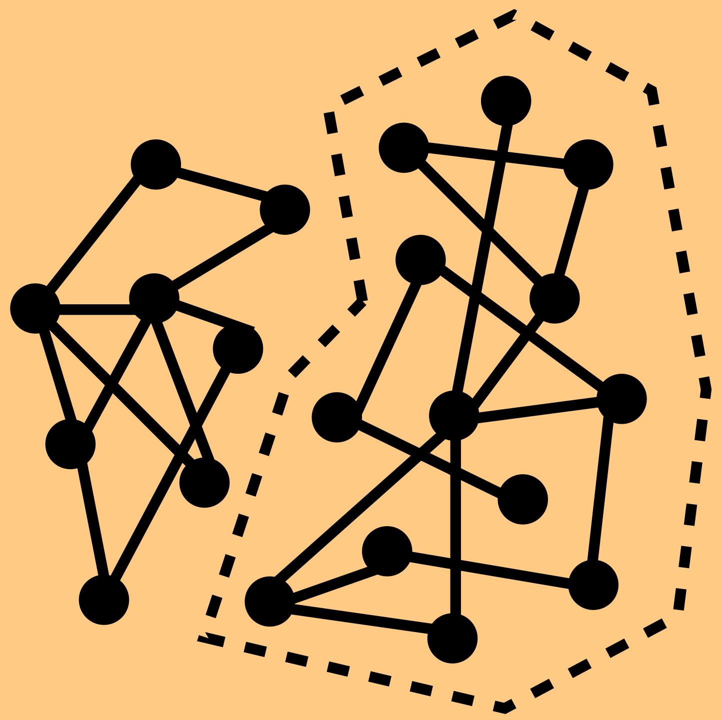


Yes Distribution



$\mathsf{val}_{\mathcal{C}} < \left(\frac{\sqrt{2}}{2} + o(1)\right) \cdot m$

Yes' Distribution



$\mathsf{val}_{\mathcal{C}} = m$

$x_i \vee x_j$

$(\neg x_i) \vee (\neg x_j)$

v

$v < \left(\frac{\sqrt{2}}{2} + \epsilon\right) \cdot m$

Yes

Reducing DBHP to Max-2OR

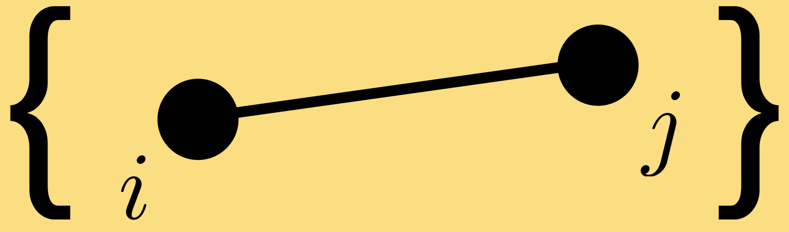
Alice

X^*

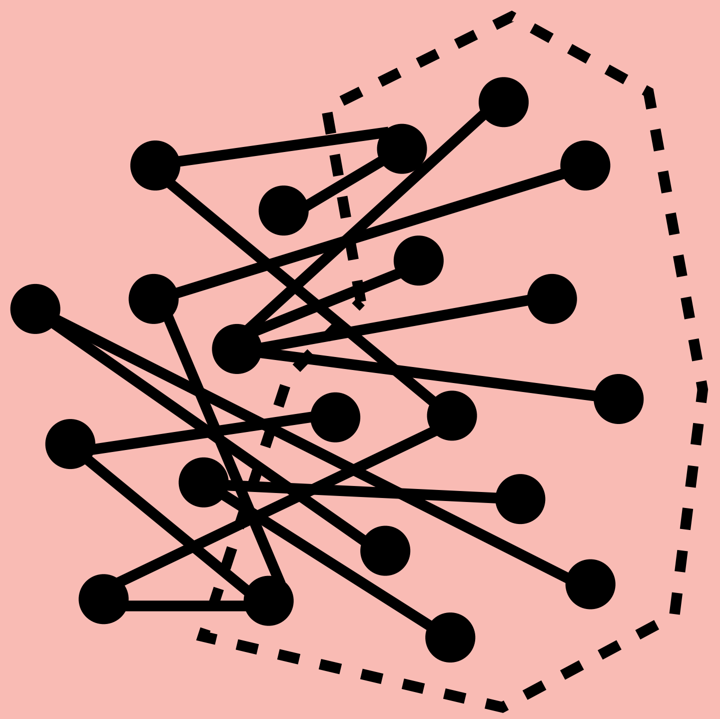
Bob 1

...

Bob T

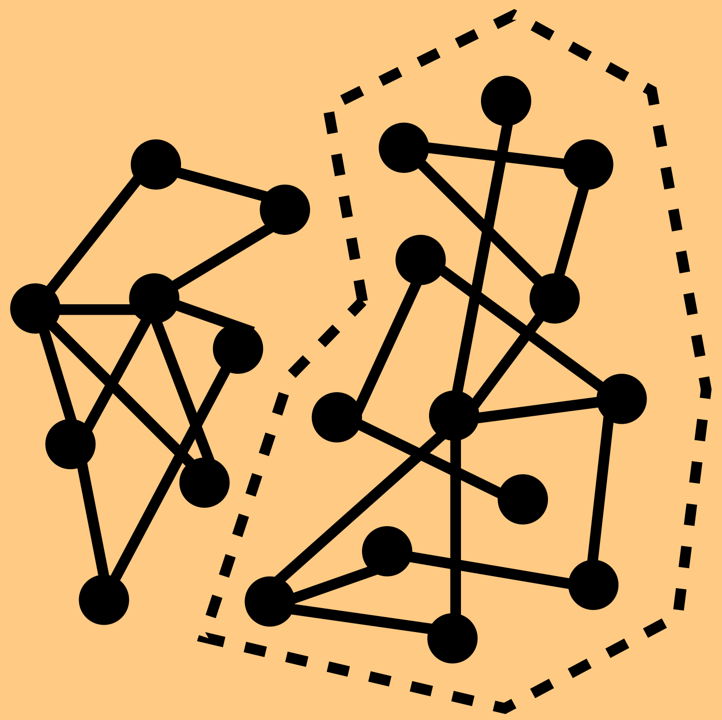


Yes Distribution



$\text{val}_{\mathcal{C}} < \left(\frac{\sqrt{2}}{2} + o(1)\right) \cdot m$

Yes' Distribution



$\text{val}_{\mathcal{C}} = m$

$x_i \vee x_j$

$(\neg x_i) \vee (\neg x_j)$

v

$v < \left(\frac{\sqrt{2}}{2} + \epsilon\right) \cdot m$ → No

$v \geq \left(\frac{\sqrt{2}}{2} + \epsilon\right) \cdot m$ → Yes