

# On Partitioning Graphs via Single Commodity Flows

**Lorenzo Orecchia**  
UC Berkeley

**Leonard J. Schulman**  
Caltech

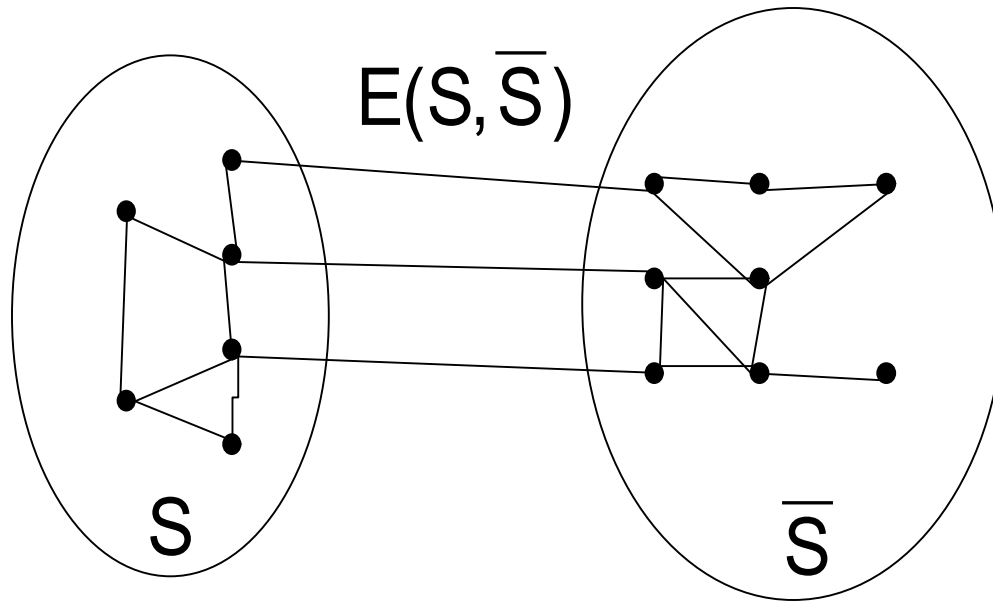
**Umesh V. Vazirani**  
UC Berkeley

**Nisheeth K. Vishnoi**  
IBM Delhi – work done while visiting UC Berkeley

# The SPARSEST CUT problem

Given a graph  $G=(V,E)$  and partition  $(S, \bar{S})$

$$\text{Expansion of } (S, \bar{S}) = \frac{|E(S, \bar{S})|}{\min\{|S|, |\bar{S}|\}}$$



# The SPARSEST CUT problem

Given a graph  $G=(V,E)$  and partition  $(S, \bar{S})$

$$\text{Expansion of } (S, \bar{S}) = \frac{|E(S, \bar{S})|}{\min\{|S|, |\bar{S}|\}}$$

## **SPARSEST CUT:**

find  $(S, \bar{S})$  with minimum expansion  $\phi(G)$ .

**Applications:** Divide-and-Conquer, Image Segmentation, VLSI design, Clustering.

**Theoretical Importance:** Metric Embeddings, Spectral Methods.

# The SPARSEST CUT problem

Given a graph  $G=(V,E)$  and partition  $(S, \bar{S})$

$$\text{Expansion of } (S, \bar{S}) = \frac{|E(S, \bar{S})|}{\min\{|S|, |\bar{S}|\}}$$

## **SPARSEST CUT:**

find  $(S, \bar{S})$  with minimum expansion  $\phi(G)$ .

**Applications:** Divide-and-Conquer, Image Segmentation, VLSI design, Clustering.

**Theoretical Importance:** Metric Embeddings, Spectral Methods.

The **SPARSEST CUT** problem is **NP-hard**.

# Approximation Algorithms for SPARSEST CUT

Algorithm	Output Expansion	Running Time *
Spectral	$2\sqrt{d\phi}$ **	$O\left(\frac{d^2n}{\phi^2}\right)$ **
Leighton-Rao	$\phi \log n$	$\tilde{O}(n^2)$
ARV	$\phi\sqrt{\log n}$	[ARV] poly(n) [AHK] $\tilde{O}(n^2)$

\*All graphs have been sparsified to  $\tilde{O}(n)$  edges.      \*\* For a d-regular graph G.

# Approximation Algorithms for SPARSEST CUT

Algorithm	Output Expansion	Running Time *
Spectral	$2\sqrt{d\phi}$ **	$O\left(\frac{d^2n}{\phi^2}\right)$ **
Leighton-Rao	$\phi \log n$	$\tilde{O}(n^2)$
ARV	$\phi\sqrt{\log n}$	[ARV] poly(n) [AHK] $\tilde{O}(n^2)$

**IN PRACTICE:** Too slow for massive data sets.

Spectral and heuristics like METIS used instead.

\*All graphs have been sparsified to  $\tilde{O}(n)$  edges.    \*\* For a d-regular graph G.

# Fast Approximation Algorithms

Algorithm	Output Expansion	Running Time *
Spectral	$2\sqrt{d\phi}$ **	$O\left(\frac{d^2n}{\phi^2}\right)$ **
Leighton-Rao	$\phi \log n$	$\tilde{O}(n^2)$
ARV	$\phi\sqrt{\log n}$	[ARV] poly(n) [AHK] $\tilde{O}(n^2)$
<b>KRV</b>	$\phi(\log n)^2$	$\tilde{O}(n^{3/2})$

**CUT-MATCHING GAME: FRAMEWORK FOR COMPUTING APPROX USING s-t MAXFLOW COMPUTATIONS**

\*All graphs have been sparsified to  $\tilde{O}(n)$  edges.    \*\* For a d-regular graph G.

# Fast Approximation Algorithms

Algorithm	Output Expansion	Running Time
Leighton-Rao	$\phi \log n$	$\tilde{O}(n^2)$
ARV	$\phi \sqrt{\log n}$	[ARV] $\text{poly}(n)$ [AHK] $\tilde{O}(n^2)$
KRV	$\phi (\log n)^2$	$\tilde{O}(n^{3/2})$
<b>AK</b>	$\phi \log n$	$\tilde{O}(n^{3/2})$

\*All graphs have been sparsified to  $\tilde{O}(n)$  edges.

\*\* For a  $d$ -regular graph  $G$ .

# Our Contribution

Algorithm	Output Expansion	Running Time
KRV	$\phi(\log n)^2$	$\tilde{O}(n^{3/2})$
AK	$\phi \log n$	$\tilde{O}(n^{3/2})$
<b>THIS PAPER</b>	$\phi \log n$	$\tilde{O}(n^{3/2})$

IN KRV CUT-MATCHING  
GAME FRAMEWORK

# Our Contribution

Algorithm	Output Expansion	Running Time
KRV	$\phi(\log n)^2$	$\tilde{O}(n^{3/2})$
AK	$\phi \log n$	$\tilde{O}(n^{3/2})$
<b>THIS PAPER</b>	$\phi \log n$	$\tilde{O}(n^{3/2})$

IN KRV CUT-MATCHING  
GAME FRAMEWORK

**LOWER BOUND**

No better approx than  $\Omega((\log n)^{1/2})$  in KRV framework.

# Our Contribution

Algorithm	Output Expansion	Running Time
KRV	$\phi(\log n)^2$	$\tilde{O}(n^{3/2})$
AK	$\phi \log n$	$\tilde{O}(n^{3/2})$
<b>THIS PAPER</b>	$\phi \log n$	$\tilde{O}(n^{3/2})$

## LOWER BOUND

No better approx than  $\Omega((\log n)^{1/2})$  in KRV framework.

Best integrality gap for SDP is  $\Omega(\log \log(n))$ .

# Our Contribution

Algorithm	Output Expansion	Running Time
KRV	$\phi(\log n)^2$	$\tilde{O}(n^{3/2})$
AK	$\phi \log n$	$\tilde{O}(n^{3/2})$
<b>THIS PAPER</b>	$\phi \log n$	$\tilde{O}(n^{3/2})$

## LOWER BOUND

No better approx than  $\Omega((\log n)^{1/2})$  in KRV framework.

Best integrality gap for SDP is  $\Omega(\log \log(n))$ .

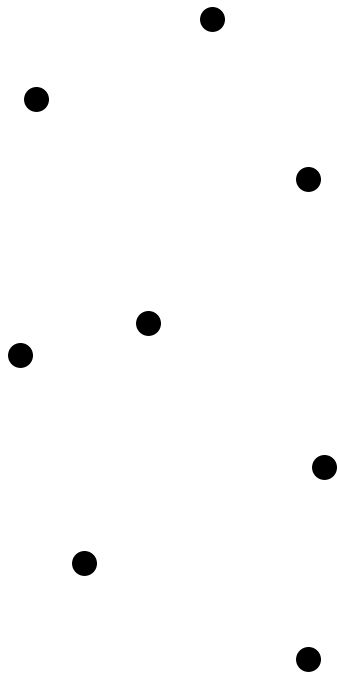
CUT-MATCHING RIGHT ABSTRACTION?

# The KRV Cut-Matching Game

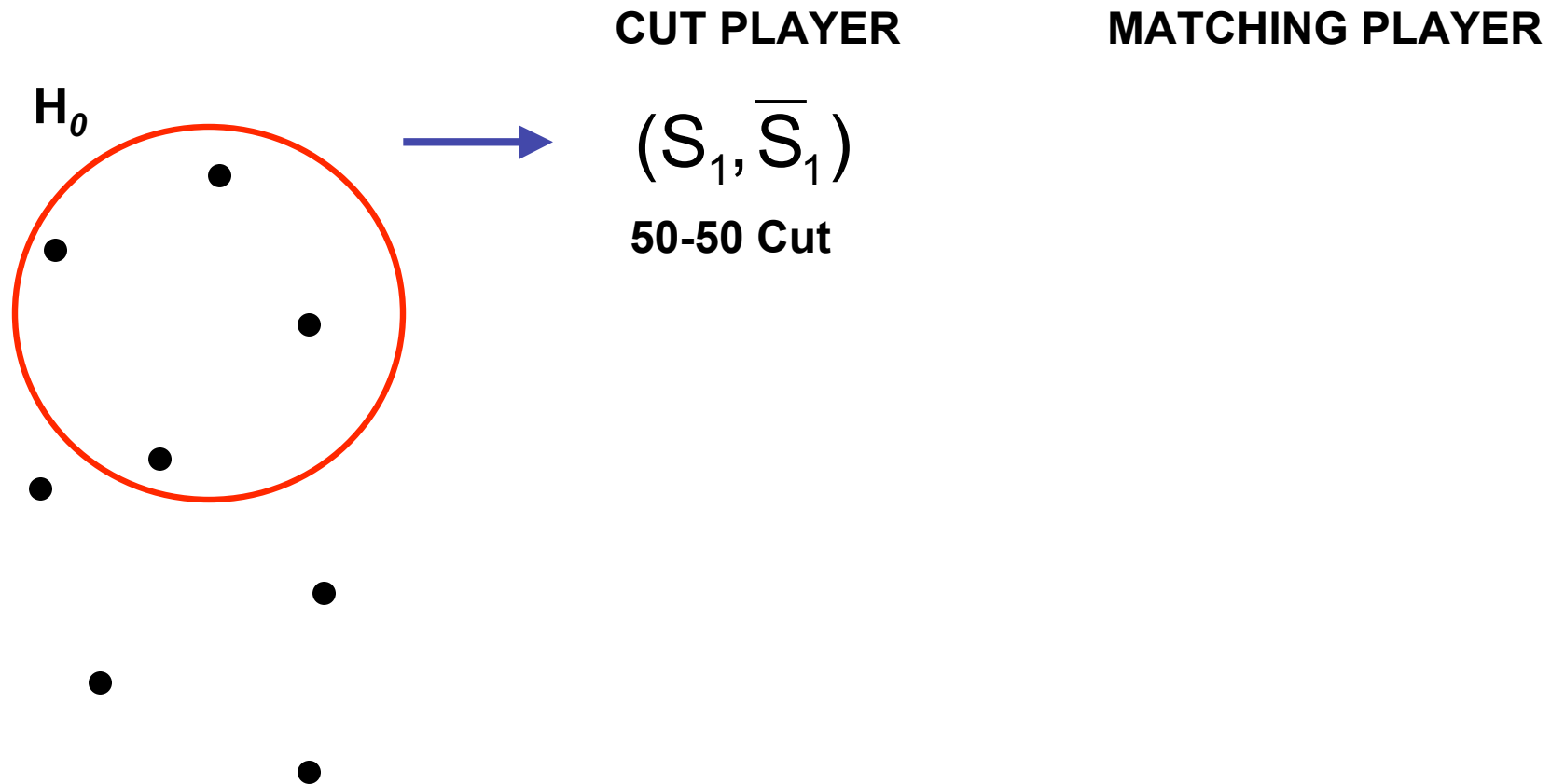
CUT PLAYER

MATCHING PLAYER

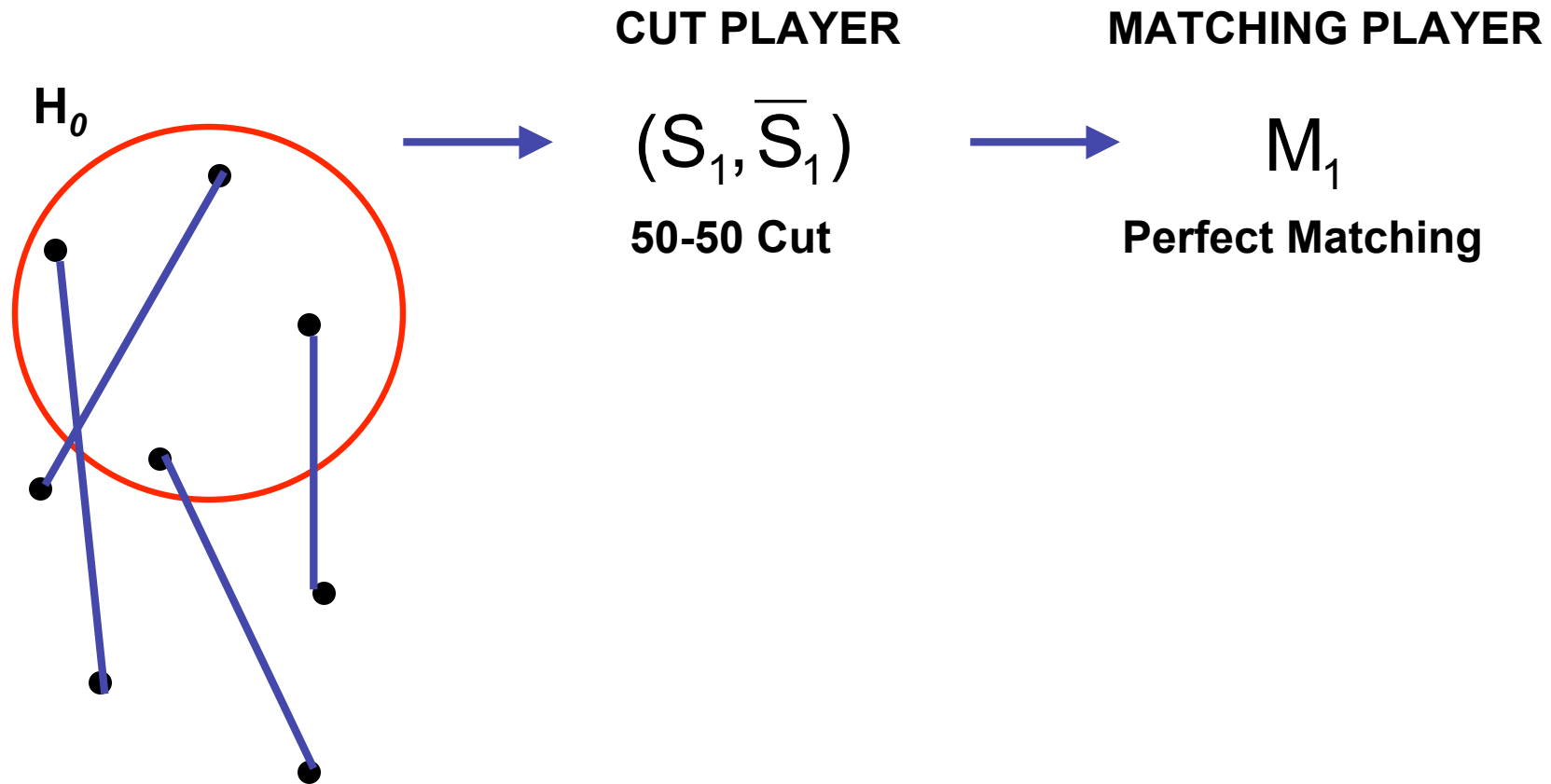
$H_0$



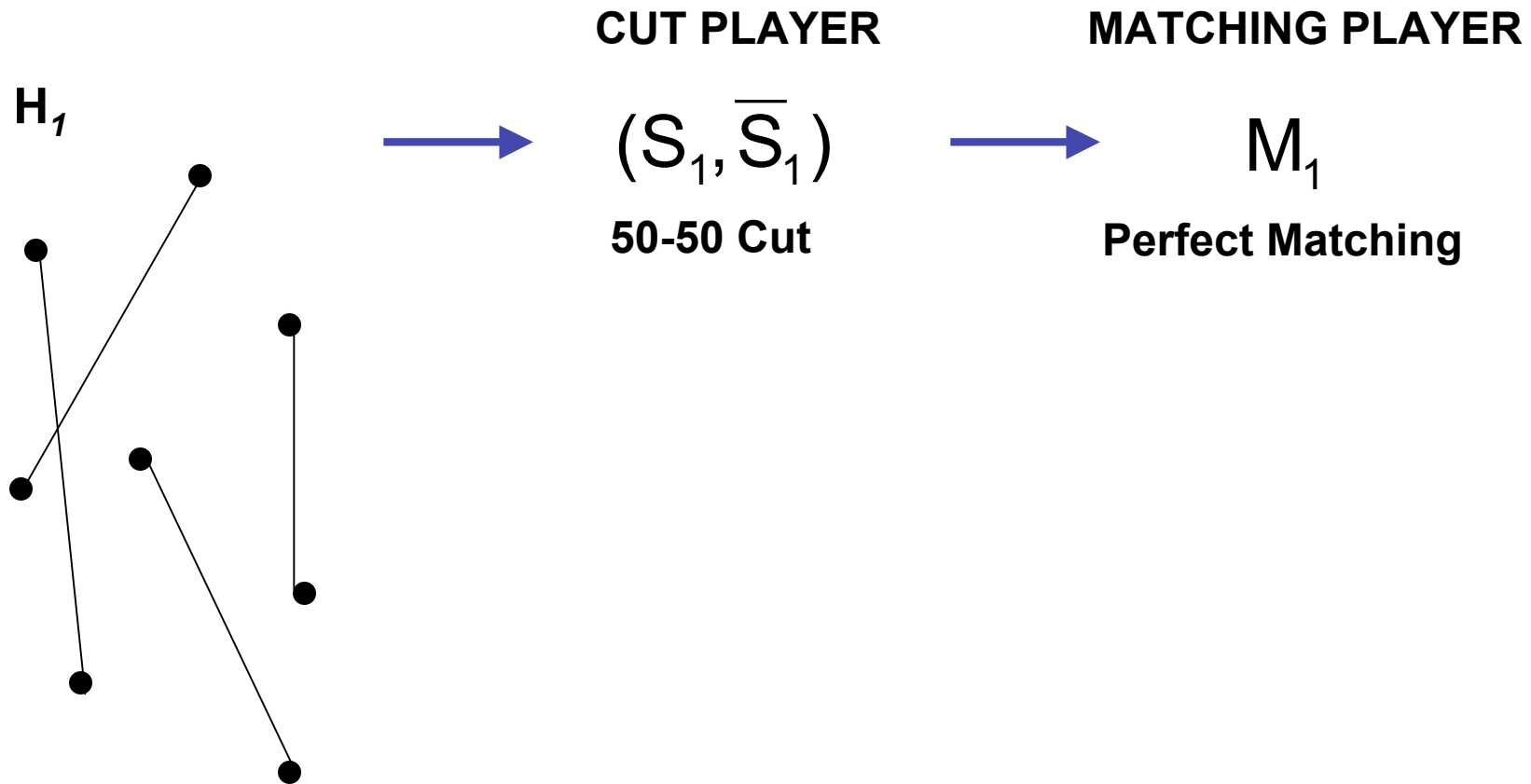
# The KRV Cut-Matching Game



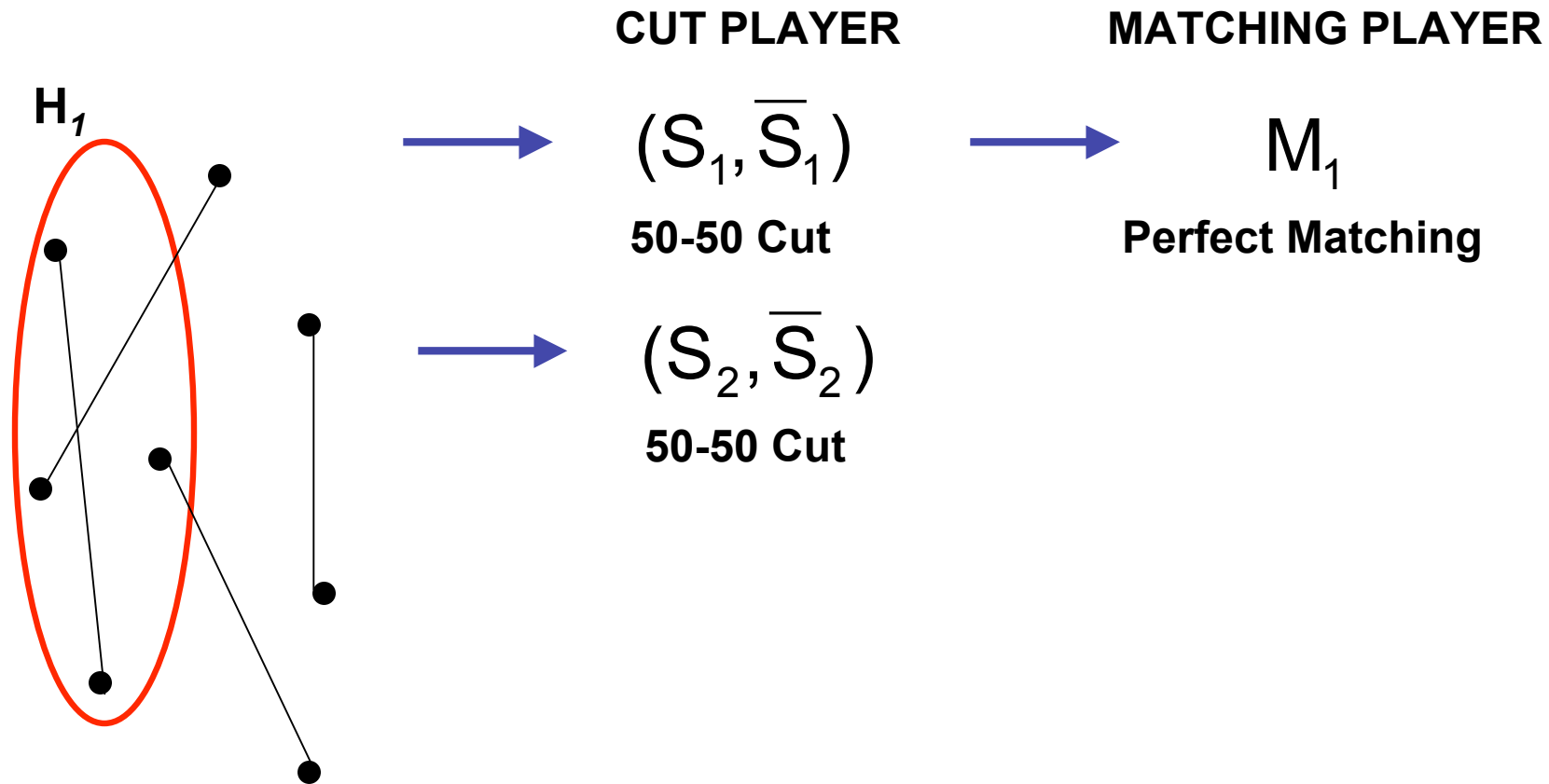
# The KRV Cut-Matching Game



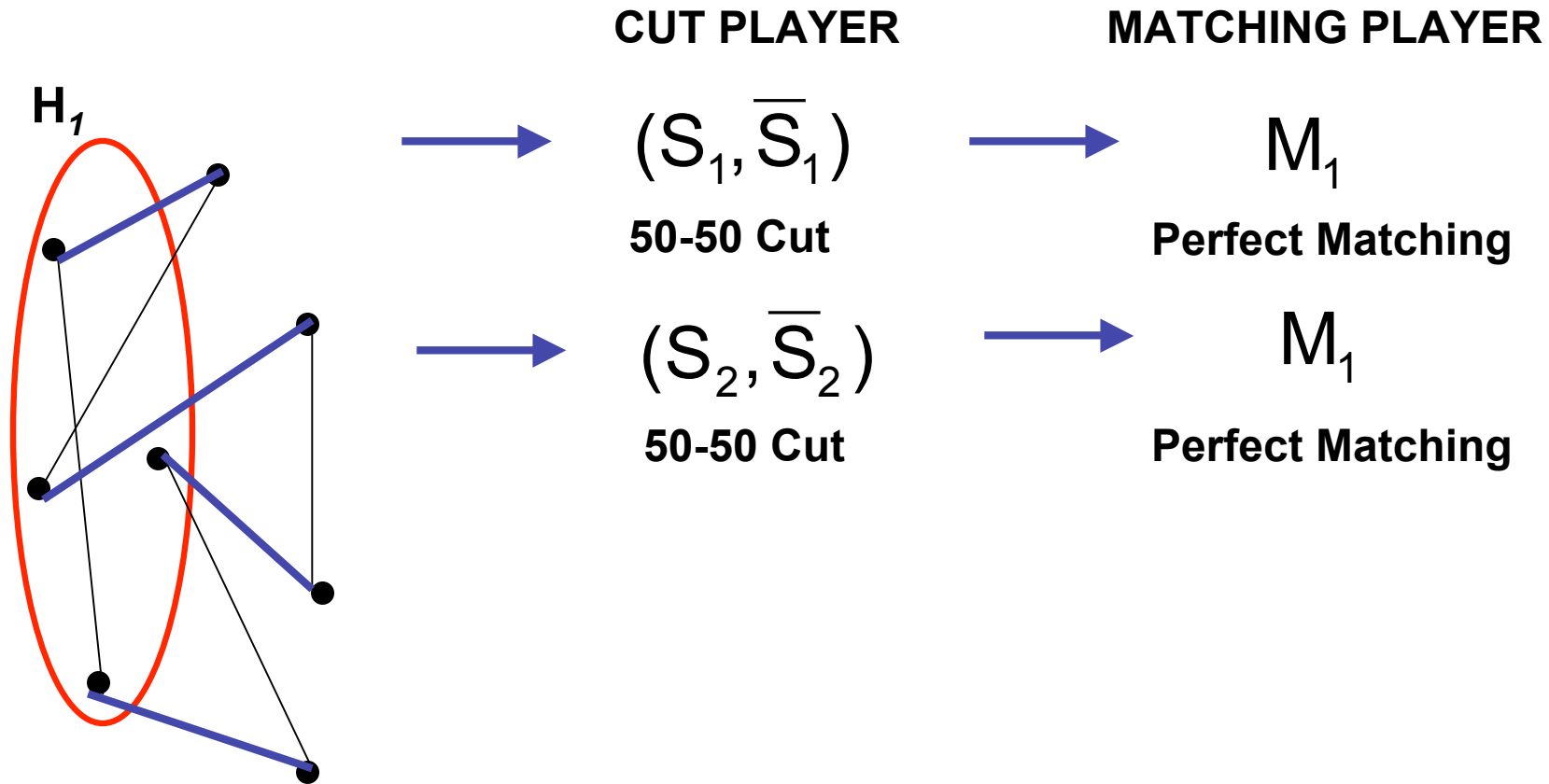
# The KRV Cut-Matching Game



# The KRV Cut-Matching Game

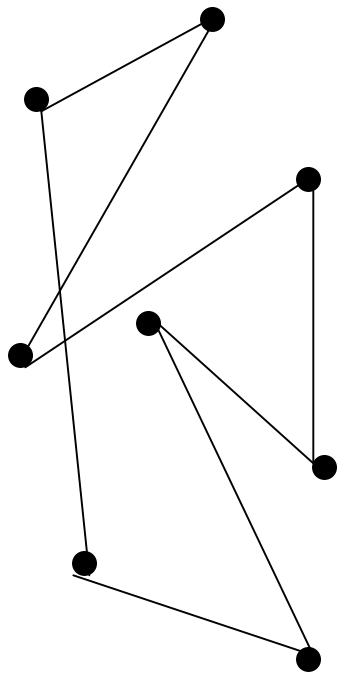


# The KRV Cut-Matching Game



# The KRV Cut-Matching Game

$H_2$



**CUT PLAYER**

$(S_1, \bar{S}_1)$

50-50 Cut



$(S_2, \bar{S}_2)$

50-50 Cut



**MATCHING PLAYER**

$M_1$

Perfect Matching

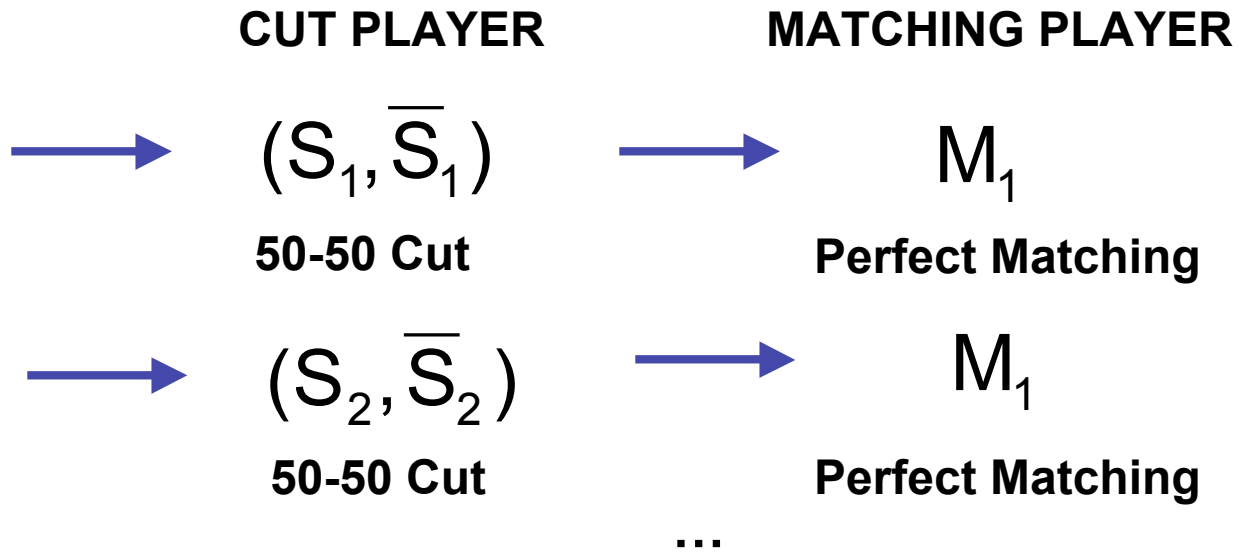
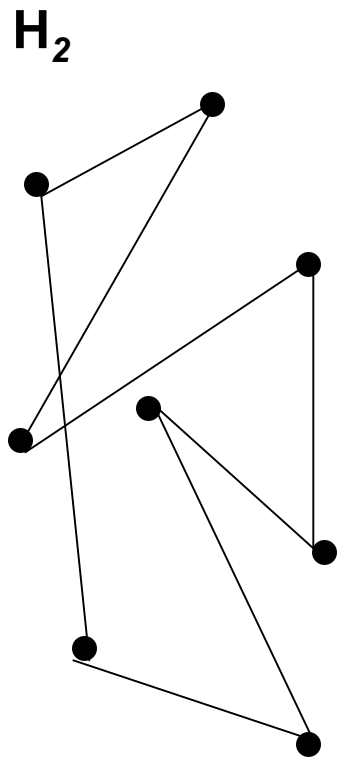


$M_1$

Perfect Matching

...

# The KRV Cut-Matching Game

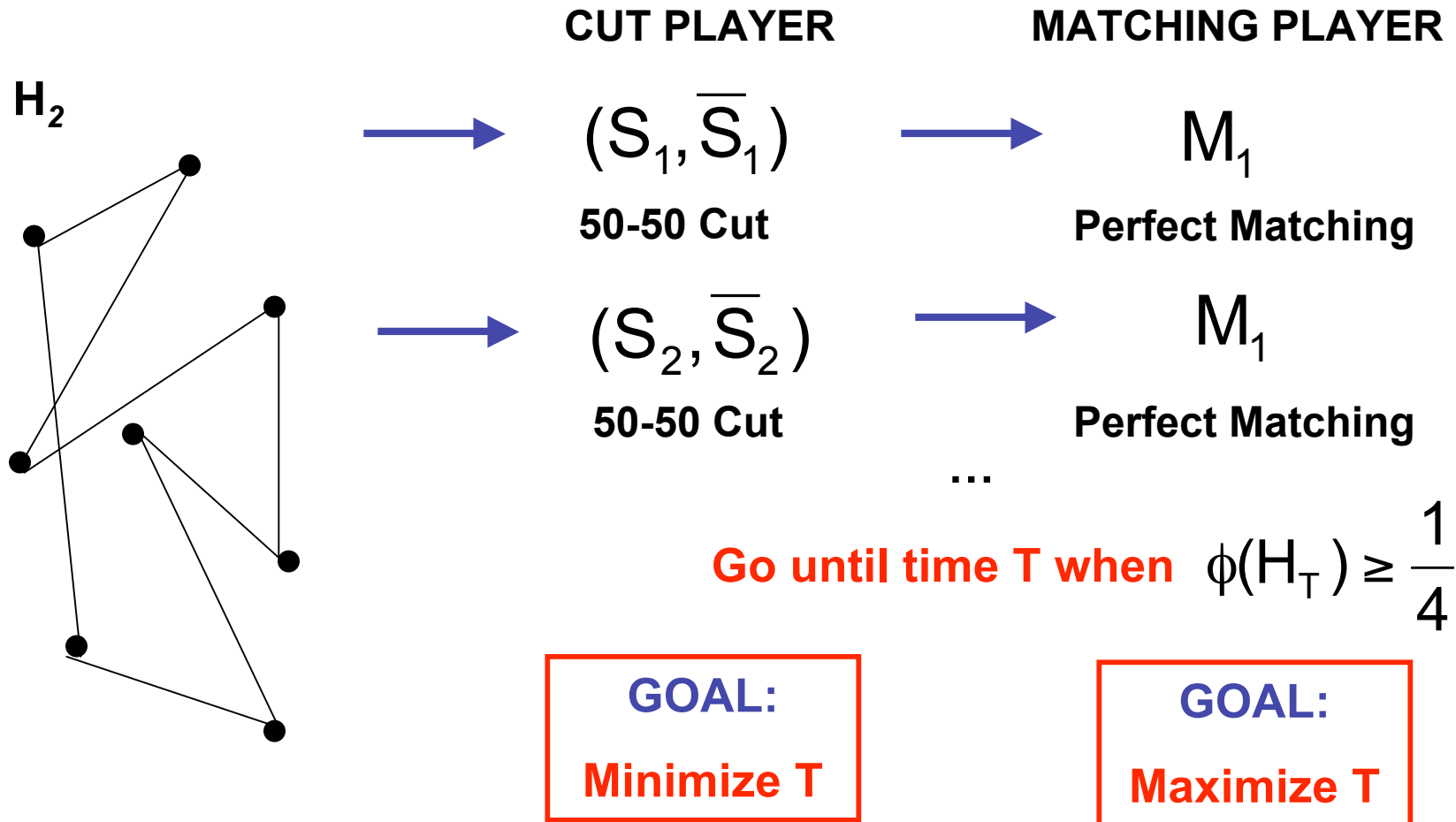


Go until time  $T$  when  $\phi(H_T) \geq \frac{1}{4}$

**GOAL:**  
**Minimize  $T$**

**GOAL:**  
**Maximize  $T$**

# The KRV Cut-Matching Game



**KRV:** there exists a cut strategy achieving  $T = O((\log n)^2)$ .

# The KRV Cut-Matching Game

Runs in time  $\mathbf{c(n)}$   
per iteration

**CUT PLAYER STRATEGY**

$T = t(n)$

# The KRV Cut-Matching Game

Runs in time  $\mathbf{c(n)}$   
per iteration

**CUT PLAYER STRATEGY**

$\mathbf{T = t(n)}$

Running time:  
 $\mathbf{t(n) \cdot (T_{\max\text{flow}} + c(n))}$

**APPROXIMATION  
ALGORITHM**

Approx Ratio:  
 $\mathbf{t(n)}$

# The KRV Cut-Matching Game

Runs in time  $\mathbf{c(n)}$   
per iteration

**CUT PLAYER STRATEGY**

$\mathbf{T = t(n)}$



Running time:  
 $\mathbf{t(n) \cdot (T_{\max\text{flow}} + c(n))}$

**APPROXIMATION  
ALGORITHM**

Approx Ratio:  
 $\mathbf{t(n)}$

Time to  
compute s-t  
maxflow in  $\mathbf{G}$

$\tilde{O}(n^{3/2})$

# The KRV Cut-Matching Game

Runs in time  $c(n)$   
per iteration

CUT PLAYER STRATEGY

$T = t(n)$



Running time:  
 $t(n) \cdot (T_{\max\text{flow}} + c(n))$

APPROXIMATION  
ALGORITHM

Approx Ratio:  
 $t(n)$

$\tilde{O}(n^{3/2})$

KRV strategy has  $c(n) = \tilde{O}(n)$  and  $t(n) = O((\log n)^2)$ .

**TOTAL RUNNING TIME:**  $\tilde{O}(n^{3/2})$

# Our Version of the Cut-Matching Game

- MODIFIED GAME

1.No Stopping Condition

2.Value of Game is  $\frac{\phi(H_T)}{T}$

# Our Version of the Cut-Matching Game

- MODIFIED GAME

1.No Stopping Condition

2.Value of Game is  $\frac{\phi(H_T)}{T}$

- STILL YIELDS  
APPROX ALGORITHM

**Approximation Ratio =  $\frac{\phi(H_T)}{T}$**

# Our Version of the Cut-Matching Game

- MODIFIED GAME

1.No Stopping Condition

2.Value of Game is  $\frac{\phi(H_T)}{T}$

- STILL YIELDS APPROX ALGORITHM

**Approximation Ratio** =  $\frac{\phi(H_T)}{T}$

- RESULTS

**CUT STRATEGY:**

$$\frac{\phi(H_T)}{T} = \frac{\Omega(\log n)}{O(\log^2 n)} = \Omega\left(\frac{1}{\log n}\right)$$

**MATCHING STRATEGY:**

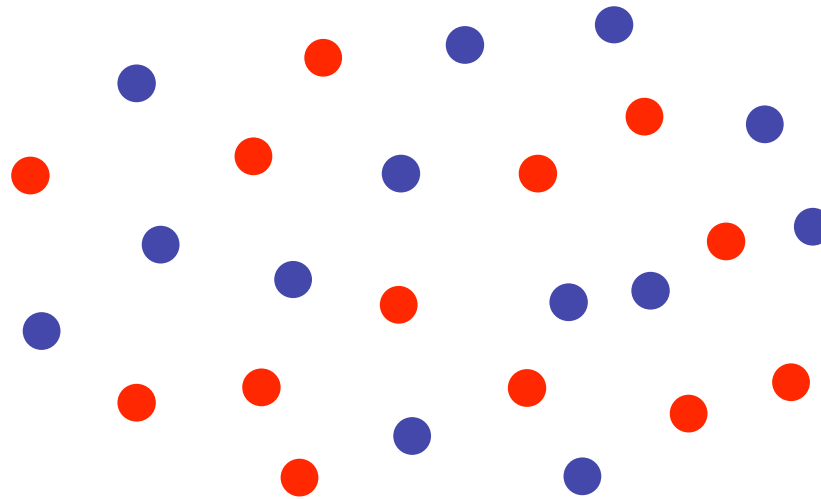
$$\frac{\phi(H_T)}{T} = O\left(\frac{1}{\sqrt{\log n}}\right)$$

# Cut Strategies: Finding Cuts Quickly

After  $t$  iterations,  $H_t = \{ M_1, M_2, \dots, M_t \}$ .

● = +1 charge      Random assignment of charge

● = -1 charge



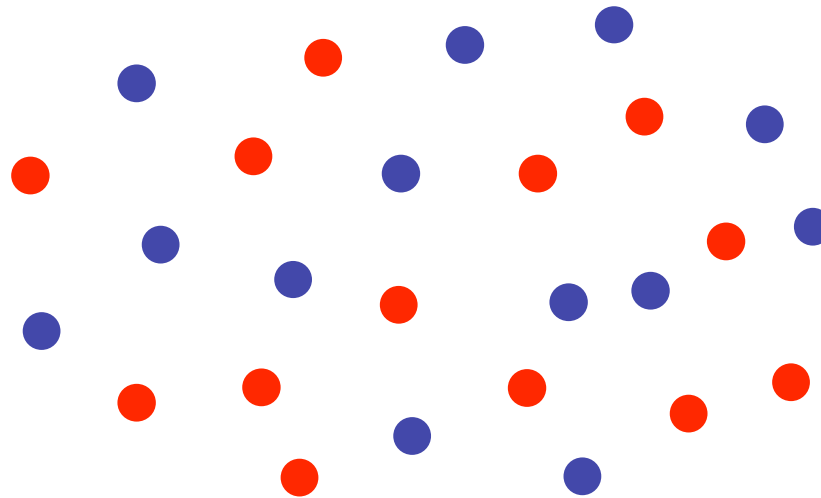
# Cut Strategies: Finding Cuts Quickly

After  $t$  iterations,  $H_t = \{ M_1, M_2, \dots, M_t \}$ .

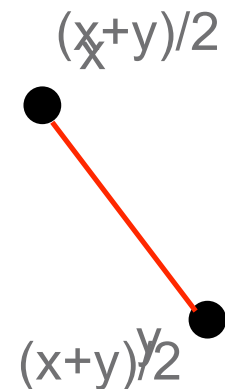
● = +1 charge

Random assignment of charge

● = -1 charge



Mix the charges along  
the matchings  $\{ M_1, M_2, \dots, M_t \}$



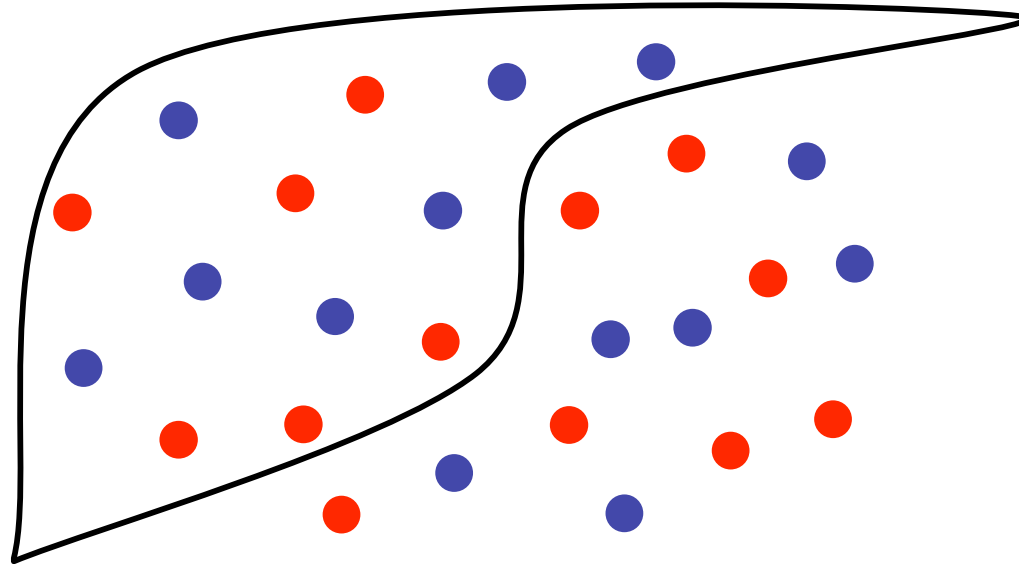
# Cut Strategies: Finding Cuts Quickly

After  $t$  iterations,  $H_t = \{ M_1, M_2, \dots, M_t \}$ .

● = +1 charge

Random assignment of charge

● = -1 charge

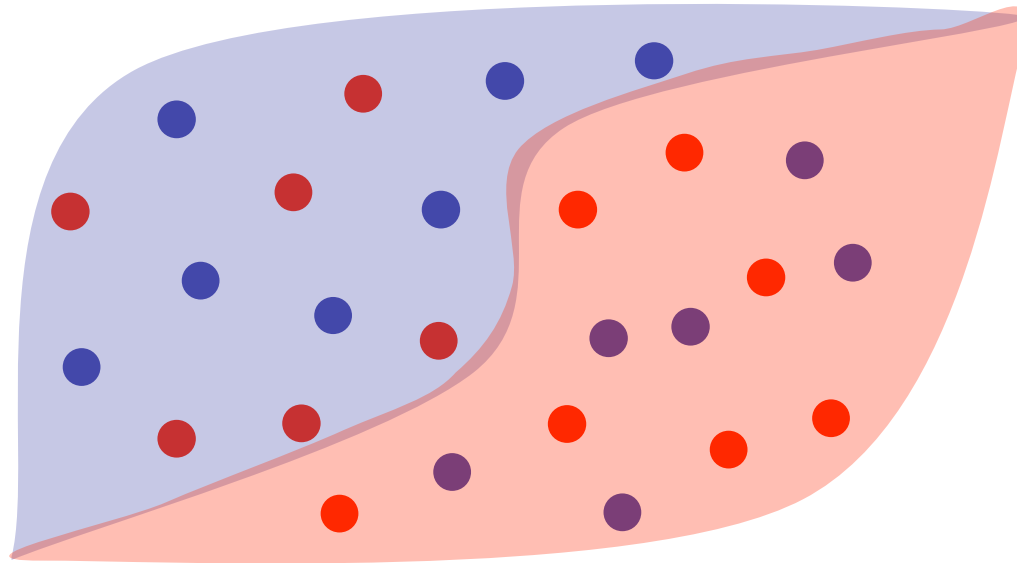


# Cut Strategies: Finding Cuts Quickly

After  $t$  iterations,  $H_t = \{ M_1, M_2, \dots, M_t \}$ .

● = +1 charge      Random assignment of charge

● = -1 charge

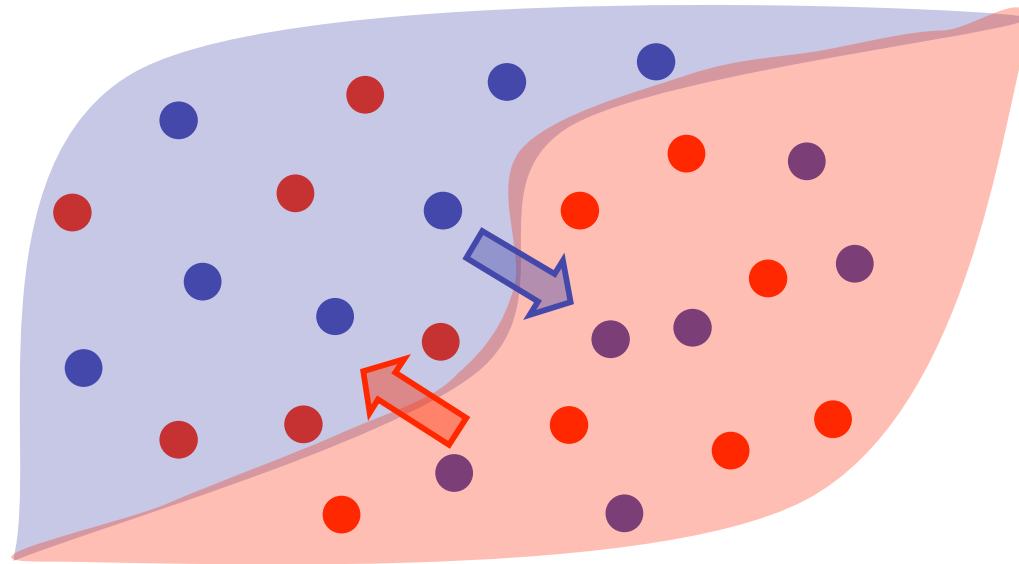


# Cut Strategies: Finding Cuts Quickly

After  $t$  iterations,  $H_t = \{ M_1, M_2, \dots, M_t \}$ .

● = +1 charge      Random assignment of charge

● = -1 charge



Mix the charges along  
the matchings  $\{ M_1, M_2, \dots, M_t \}$

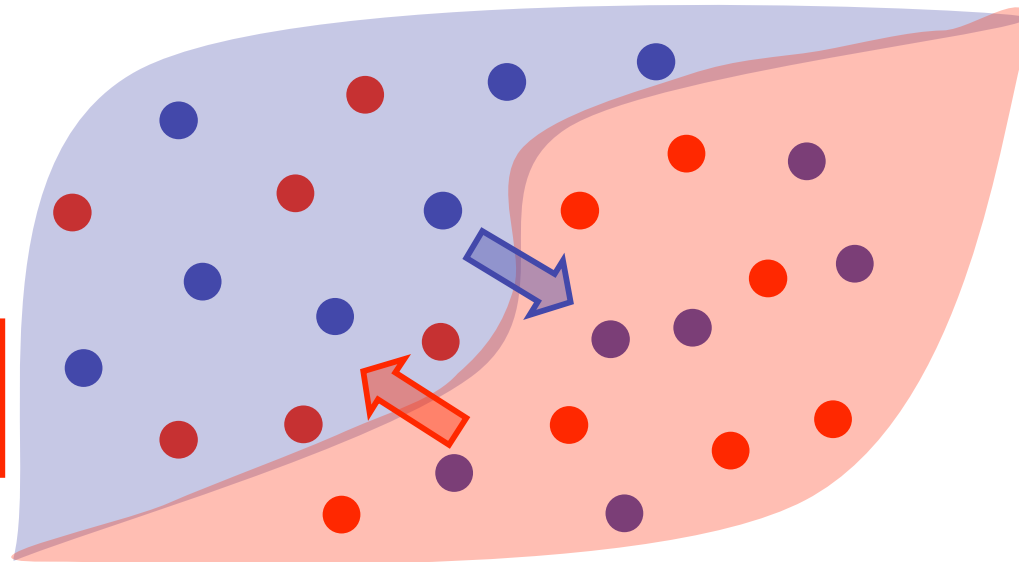
# Cut Strategies: Finding Cuts Quickly

After  $t$  iterations,  $H_t = \{ M_1, M_2, \dots, M_t \}$ .

● = +1 charge      Random assignment of charge

● = -1 charge

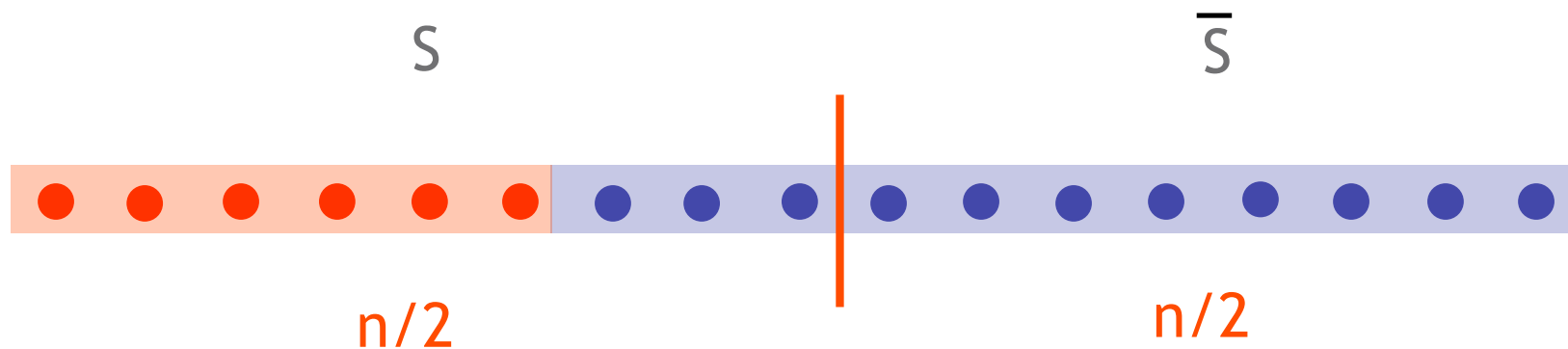
If cut is small,  
unbalance remains.



Mix the charges along  
the matchings  $\{ M_1, M_2, \dots, M_t \}$

# Cut Strategies: Finding Cuts Quickly

Order the vertices according to the final charge present and cut in half.



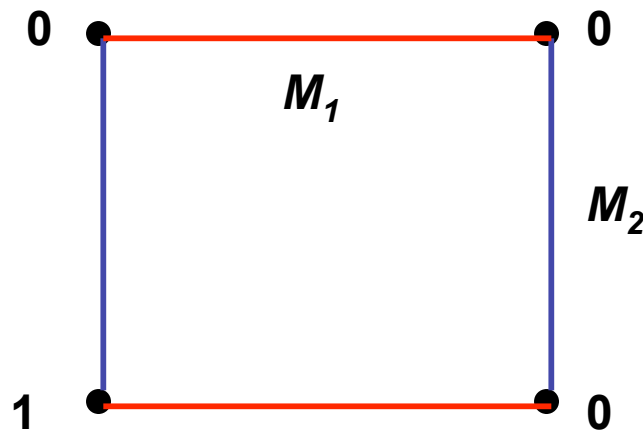
# The KRV mixing walk

## KRV-walk

At round  $t$ :

$$P(t) = \left( \frac{I + M_{t-1}}{2} \right) \left( \frac{I + M_{t-2}}{2} \right) \cdots \left( \frac{I + M_1}{2} \right)$$

Lazy random walk traversing matchings in order.



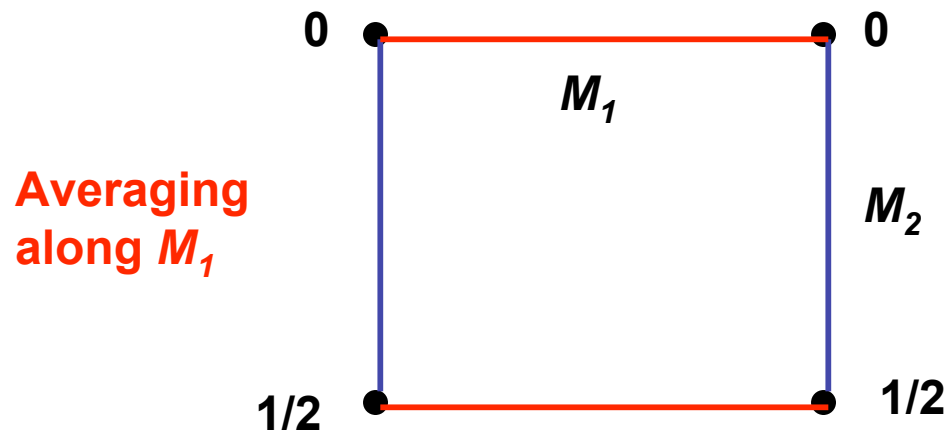
# The KRV mixing walk

## KRV-walk

At round  $t$ :

$$P(t) = \left( \frac{I + M_{t-1}}{2} \right) \left( \frac{I + M_{t-2}}{2} \right) \cdots \left( \frac{I + M_1}{2} \right)$$

Lazy random walk traversing matchings in order.



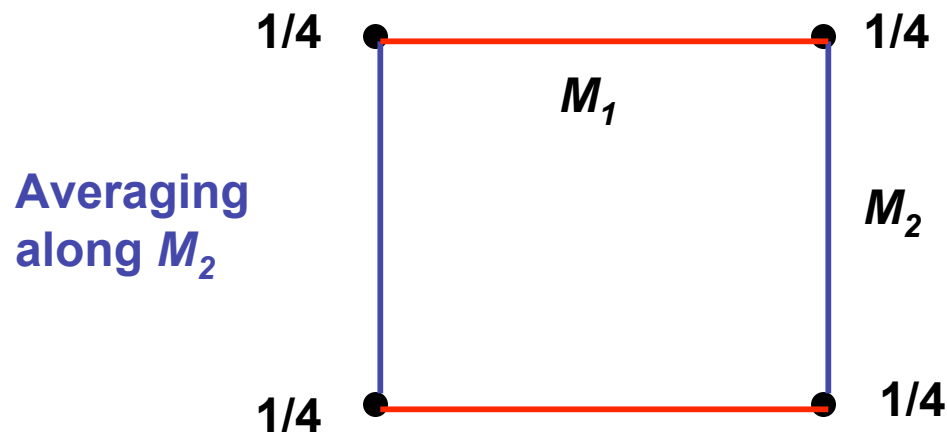
# The KRV mixing walk

## KRV-walk

At round  $t$ :

$$P(t) = \left( \frac{I + M_{t-1}}{2} \right) \left( \frac{I + M_{t-2}}{2} \right) \cdots \left( \frac{I + M_1}{2} \right)$$

Lazy random walk traversing matchings in order.



# Sketch of KRV Analysis

1. Mixing of  $P(t)$  measured by **potential function**

$$\Psi_t = \| P(t) - J/n \|_F^2$$

2. If  $P(t)$  mixes well,  $H_t$  has good expansion.  
Possible to embed  $K_n$  in  $H_t$ .

3. Potential Reduction at every iteration

$$\Psi_t = \Psi_{t-1} - L(M_t) \cdot P(t)$$

Mixing due to matching  $M_t$

Decomposition possible as KRV walks matchings in order.

4. Cut-finding procedure reduces potential by a fixed factor

$$\Psi_t = \Psi_{t-1} \left( 1 - \frac{1}{\log n} \right)$$

Yields expander in  $O((\log n)^2)$  rounds

# Why KRV cannot do better

**Recall:**

**Approximation**

**is**

$$\frac{\phi(H_T)}{T}$$

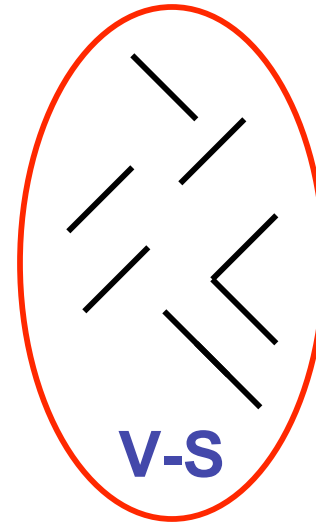
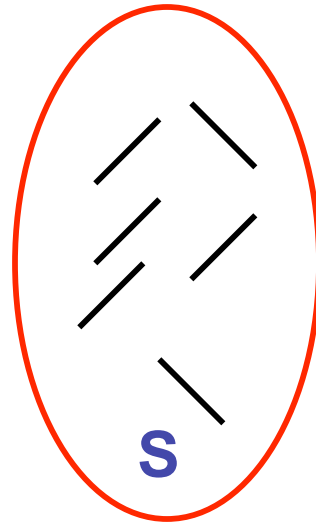
# Why KRV cannot do better

**Recall:**

**Approximation  
is**

**Can KRV get  
better than  $O(1)$   
expansion?**

# Why KRV cannot do better



**Recall:**  
Approximation  
is

Can KRV get  
better than  $O(1)$   
expansion?

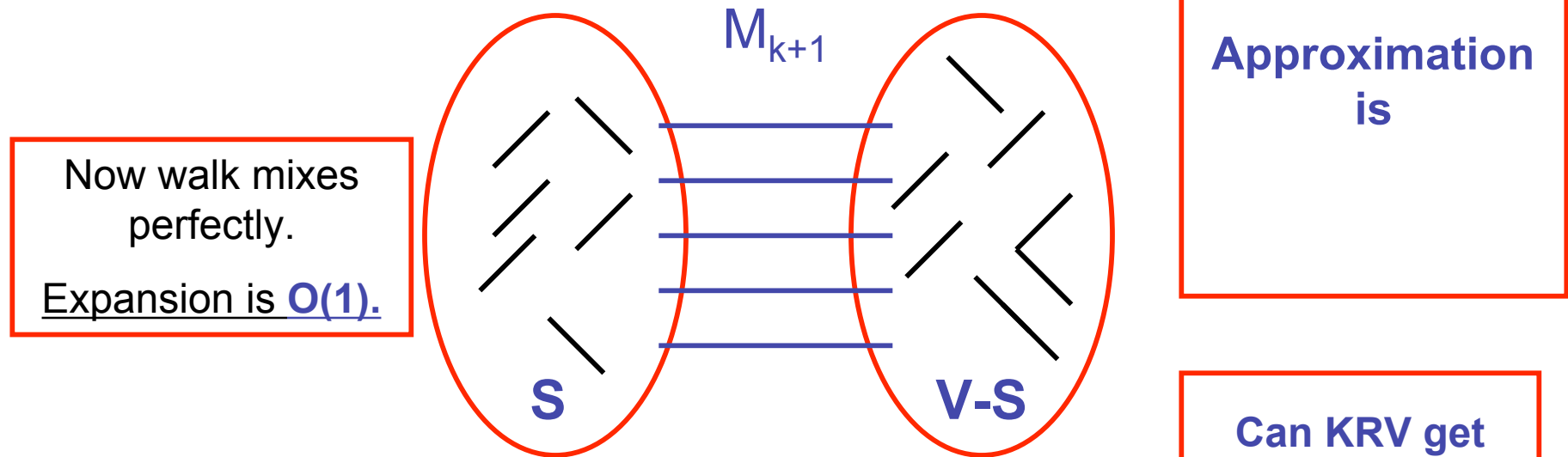
**SUPPOSE:**

Walk on  $M_1, M_2, \dots, M_k$  mixes perfectly on S and V-S

and

no edge cross (S,V-S)

# Why KRV cannot do better



**SUPPOSE:**

Walk on  $M_1, M_2, \dots, M_k$  mixes perfectly on  $S$  and  $V-S$

and

no edge cross  $(S, V-S)$

Can KRV get better than  $O(1)$  expansion?

# Our Cut Strategy: a Different Walk

IDEA: use lazy natural random walk

$$P(t) = \frac{1}{2} + \frac{M_1 + M_2 + \dots + M_{t-1}}{2(t-1)}$$

## ADVANTAGES:

- Eliminates bad case: possible to get better expansion.
- Better handle on expansion through mixing by Cheeger's Inequality.

## CHALLENGE:

- Impossible to decompose potential as in KRV.

$$\Psi_t = \Psi_{t-1} - L(M_t) \cdot P(t)$$

**Additional matching modifies all steps of walk.**

# Our Cut Strategy: a Different Walk

IDEA: use lazy natural random walk

$$P(t) = \left( \frac{1}{2} + \frac{M_1 + M_2 + \dots + M_{t-1}}{2(t-1)} \right)^d$$

ADVANTAGES:

- Eliminates bad case: possible to get better expansion.
- Better handle on expansion through mixing by Cheeger's Inequality.

CHALLENGE:

- Impossible to decompose potential as in KRV.

~~$$\Psi_t = \Psi_{t-1} - L(M_t) \cdot P(t)$$~~

Additional matching modifies all steps of walk.

# Modified Walk and Matrix Inequalities

## CHALLENGE:

Impossible to decompose potential as in  
**KRV.**

Additional matching modifies all steps of walk.

## SOLUTION:

Use ~~round-robin~~ walk close to natural walk:

$$N_i = \frac{d}{d+1}I + \frac{1}{d+1}M_i$$

$$P(t) = (N_1 N_2 \dots N_{t-1} N_{t-1} N_{t-2} \dots N_1)^d$$

Apply **matrix inequality:**

$$\|(ABA)^t\| \leq \|A^t B^t A^t\|$$

# Modified Walk and Matrix Inequalities

## CHALLENGE:

Impossible to decompose potential as in  
**KRV.**

Additional matching modifies all steps of walk.

## SOLUTION:

Use ~~round-robin walk~~ close to natural walk:

$$N_i = \frac{d}{d+1}I + \frac{1}{d+1}M_i$$

$$P(t) = (N_1 N_2 \dots N_{t-1} N_{t-1} N_{t-2} \dots N_1)^d$$

Apply ~~matrix inequality~~:

$$\|(ABA)^t\| \leq \|A^t B^t A^t\|$$

~~$$\Psi_t = \Psi_{t-1} - L(M_t) \cdot P(t)$$~~

# Modified Walk and Matrix Inequalities

## CHALLENGE:

Impossible to decompose potential as in  
**KRV.**

Additional matching modifies all steps of walk.

## SOLUTION:

Use ~~round-robin~~ walk close to natural walk:

$$N_i = \frac{d}{d+1}I + \frac{1}{d+1}M_i$$

$$P(t) = (N_1 N_2 \dots N_{t-1} N_{t-1} N_{t-2} \dots N_1)^d$$

Apply **matrix inequality:**

$$\|(ABA)^t\| \leq \|A^t B^t A^t\|$$

$$\Psi_t = \Psi_{t-1} - L(M_t) \cdot P(t)$$

# Modified Walk and Matrix Inequalities

## SOLUTION:

Use **round-robin walk** close to natural.  
Apply **matrix inequality**.

Yields **same potential** reduction as KRV.

But our walk is **better related to expansion**:

In  $O((\log n)^2)$  rounds,  
conductance  $(1/\log n)$  by Cheeger.

# Modified Walk and Matrix Inequalities

## SOLUTION:

Use **round-robin walk** close to natural.  
Apply **matrix inequality**.

Yields **same potential** reduction as KRV.

But our walk is **better related to expansion**:

In  $O((\log n)^2)$  rounds,  
conductance  $(1/\log n)$  by Cheeger.

$\Omega(\log n)$  expansion in  $O((\log n)^2)$  rounds.

# Modified Walk and Matrix Inequalities

## SOLUTION:

Use **round-robin walk** close to natural.  
Apply **matrix inequality**.

Yields **same potential** reduction as KRV.

But our walk is **better related to expansion**:

In  $O((\log n)^2)$  rounds,  
conductance  $(1/\log n)$  by Cheeger.

$\Omega(\log n)$  expansion in  $O((\log n)^2)$  rounds.

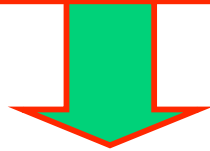
TIME: only polylog factors worse than KRV

# Lower Bound

Matching player yielding

$$\frac{\phi(H_T)}{T} = O\left(\frac{1}{\sqrt{\log n}}\right)$$

against any **Cut player**.



No better approximation than  $O((\log n)^{1/2})$

in **KRV Cut-Matching game**

# Lower Bound Idea

**A NAÏVE MATCHING PLAYER:**

**Fix a cut  $(S, V-S)$ . Keep it as sparse as possible.**

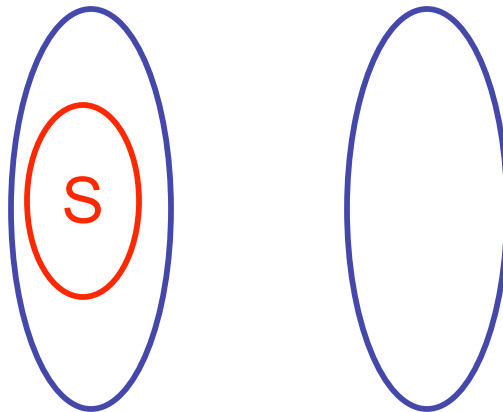


# Lower Bound Idea

**A NAÏVE MATCHING PLAYER:**

**Fix a cut  $(S, V-S)$ . Keep it sparse.**

Cut player  
plays...

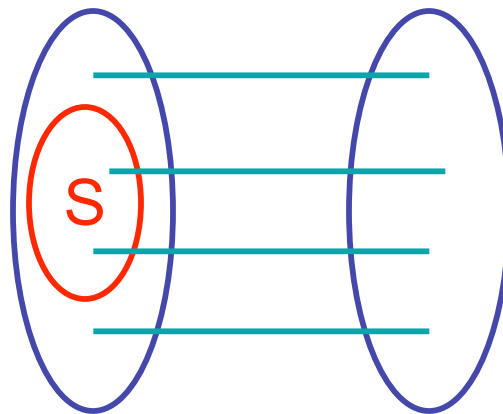


# Lower Bound Idea

## A NAÏVE MATCHING PLAYER:

Fix a cut  $(S, V-S)$ . Keep it sparse.

Cut player  
plays...



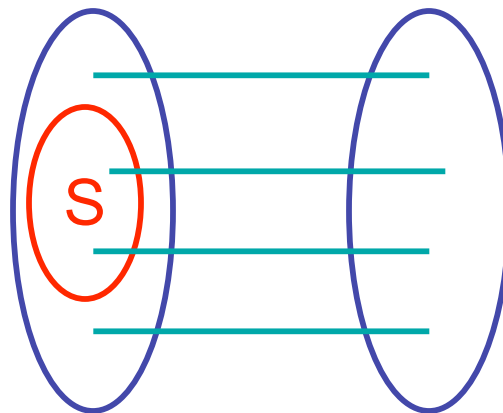
GAME OVER

# Lower Bound Idea

## A NAÏVE MATCHING PLAYER:

Fix a cut  $(S, V-S)$ . Keep it sparse.

Cut player  
plays...



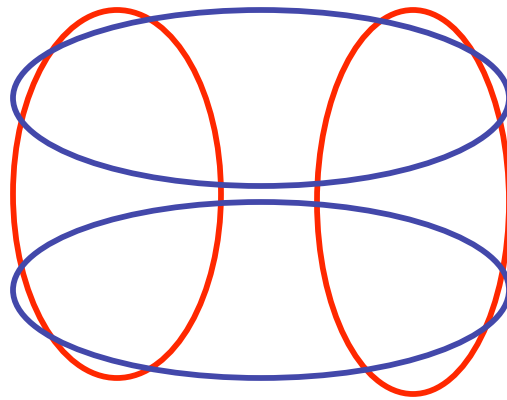
GAME OVER

IDEA: hedge over many cuts

# Lower Bound Idea

**THE REAL PLAYER - AT START:**

**Matching player selects  $\log(n)$  'orthogonal' 50-50 cuts in  $V$ .**



**Orthogonal 50-50 cuts**

**Minimum correlation**



**Cut player cannot 'kill'**

**many cuts at once**

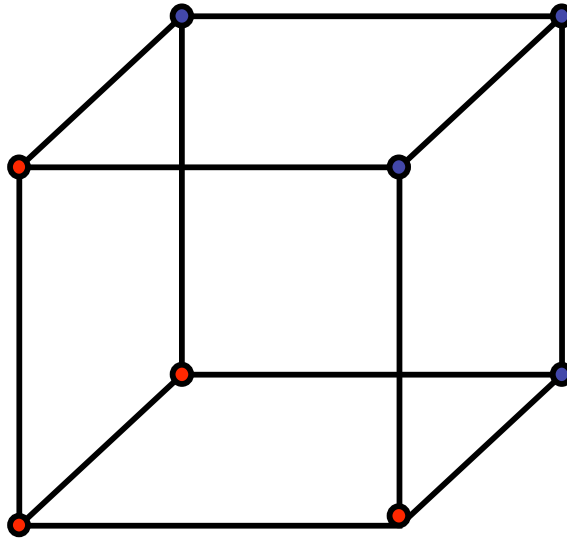
**THE REAL PLAYER - THROUGHOUT THE GAME:**

**Matching player adds matchings to minimize average expansion.**

# Main Lemma

$\forall$  50-50 cut (**S**, **V-S**),

$$H_d = \{-1, +1\}^d$$

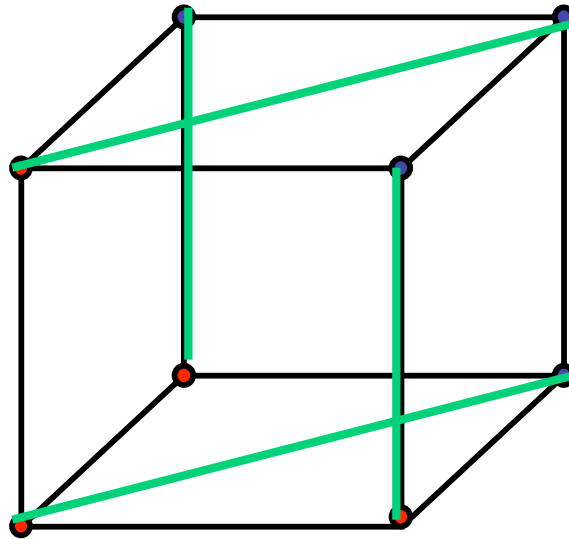


# Main Lemma

$\forall$  50-50 cut (**S**, **V-S**),

$\exists$  a perfect matching **M**, s.t.

$$H_d = \{-1, +1\}^d$$



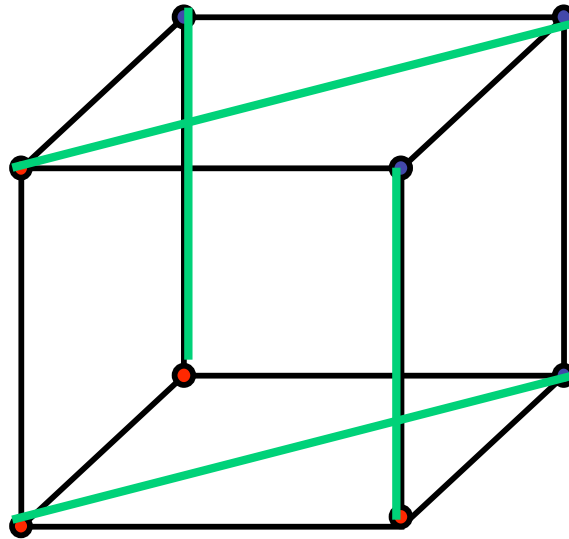
# Main Lemma

$\forall$  50-50 cut (**S**, **V-S**),

$\exists$  a perfect matching **M**, s.t.

$$\sum_{(u,v) \in M} |u - v|_1 = O(\sqrt{d})$$

$$H_d = \{-1, +1\}^d$$



# Conclusion and Open Problems

## POWER OF CUT-MATCHING GAME:

Simple yet powerful framework for **SPARSEST CUT**.

## OPEN QUESTION:

Can we use Cut-Matching to get fast  $(\log n)^{1/2}$  approximation?