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

STOC 2008, Victoria
The SPARSEST CUT problem

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

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 $(S, \overline{S})$

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

SPARSEST CUT: find $(S, \overline{S})$ with minimum expansion $\phi(G)$.

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

The SPARSEST CUT problem

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

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

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

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

** $d$-regular for a d-regular graph $G$.
## Approximation Algorithms for SPARSEST CUT

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Output Expansion</th>
<th>Running Time</th>
<th>*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spectral</td>
<td>$2\sqrt{d}\phi$</td>
<td>$O\left(\frac{d^2n}{\phi^2}\right)$</td>
<td>**</td>
</tr>
<tr>
<td>Leighton-Rao</td>
<td>$\phi \log n$</td>
<td>$\tilde{O}(n^2)$</td>
<td></td>
</tr>
<tr>
<td>ARV</td>
<td>$\phi \sqrt{\log n}$</td>
<td>$[\text{ARV}] \ poly(n)$ [\text{AHK}] $\tilde{O}(n^2)$</td>
<td></td>
</tr>
</tbody>
</table>

**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

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

**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

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

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

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Output Expansion</th>
<th>Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>KRV</td>
<td>$\phi(\log n)^2$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
<tr>
<td>AK</td>
<td>$\phi \log n$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
<tr>
<td><strong>THIS PAPER</strong></td>
<td>$\phi \log n$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
</tbody>
</table>

*IN KRV CUT-MATCHING GAME FRAMEWORK*
Our Contribution

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Output Expansion</th>
<th>Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>KRV</td>
<td>$\phi(\log n)^2$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
<tr>
<td>AK</td>
<td>$\phi \log n$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
<tr>
<td>THIS PAPER</td>
<td>$\phi \log n$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
</tbody>
</table>

**IN KRV CUT-MATCHING GAME FRAMEWORK**

**LOWER BOUND**

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

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Output Expansion</th>
<th>Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>KRV</td>
<td>$\phi(\log n)^2$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
<tr>
<td>AK</td>
<td>$\phi \log n$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
<tr>
<td>THIS PAPER</td>
<td>$\phi \log n$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
</tbody>
</table>

**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

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Output Expansion</th>
<th>Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>KRV</td>
<td>$\phi(\log n)^2$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
<tr>
<td>AK</td>
<td>$\phi \log n$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
<tr>
<td>THIS PAPER</td>
<td>$\phi \log n$</td>
<td>$\tilde{O}(n^{3/2})$</td>
</tr>
</tbody>
</table>

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

$H_0$

CUT PLAYER

MATCHING PLAYER

$(S_1, \overline{S_1})$

50-50 Cut
The KRV Cut-Matching Game

CUT PLAYER

MATCHING PLAYER

\( H_0 \)

\((S_1, \overline{S_1})\)

50-50 Cut

\( M_1 \)

Perfect Matching
The KRV Cut-Matching Game

H₁

CUT PLAYER

MATCHING PLAYER

(S₁, ¯S₁)
50-50 Cut

M₁
Perfect Matching
The KRV Cut-Matching Game

CUT PLAYER

MATCHING PLAYER

\[ (S_1, \overline{S_1}) \]
50-50 Cut

\[ (S_2, \overline{S_2}) \]
50-50 Cut

\[ M_1 \]
Perfect Matching
The KRV Cut-Matching Game

CUT PLAYER

\( (S_1, \overline{S_1}) \)
50-50 Cut

MATCHING PLAYER

\( M_1 \)
Perfect Matching

\( (S_2, \overline{S_2}) \)
50-50 Cut

\( M_1 \)
Perfect Matching
The KRV Cut-Matching Game

\[ H_2 \]

\[ \begin{align*}
&\text{CUT PLAYER} \\
&\quad \rightarrow (S_1 , \overline{S}_1) \\
&\quad \quad \text{50-50 Cut} \\
&\quad \rightarrow (S_2 , \overline{S}_2) \\
&\quad \quad \text{50-50 Cut} \\
&\quad \rightarrow \quad M_1 \\
&\quad \quad \text{Perfect Matching} \\
&\quad \rightarrow \quad M_1 \\
&\quad \quad \text{Perfect Matching} \\
&\quad \ldots
\end{align*} \]
The KRV Cut-Matching Game

**GOAL:**
- Minimize $T$
- Maximize $T$

**CUT PLAYER**
- $(S_1, \overline{S}_1)$
  - 50-50 Cut

**MATCHING PLAYER**
- $M_1$
  - Perfect Matching

... Go until time $T$ when $\phi(H_T) \geq \frac{1}{4}$
The KRV Cut-Matching Game

CUT PLAYER

MAPPING PLAYER

50-50 Cut

50-50 Cut

GOAL: Maximize $T$ Go until time $T$ when

$\phi(H_T) \geq \frac{1}{4}$

GOAL: Minimize $T$

KRV: there exists a cut strategy achieving $T = O((\log n)^2)$. 
The KRV Cut-Matching Game

Runs in time \( c(n) \)
per iteration

CUT PLAYER STRATEGY

\( T = t(n) \)
The KRV Cut-Matching Game

- Runs in time $c(n)$ per iteration
- Running time: $t(n) \cdot (T_{\text{maxflow}} + c(n))$
The KRV Cut-Matching Game

Runs in time $c(n)$ per iteration

CUT PLAYER STRATEGY

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

APPROXIMATION ALGORITHM

Time to compute s-t maxflow in $G$

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

$T = t(n)$

Approx Ratio: $t(n)$
The KRV Cut-Matching Game

CUT PLAYER STRATEGY

Runs in time $c(n)$ per iteration

APPROXIMATION ALGORITHM

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

Approx Ratio: $t(n)$

$O((\log n)^2)$

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

KRV strategy has $c(n) = \tilde{O}(n)$ and $t(n) = O((\log n)^2)$. $\tilde{O}(n^{3/2})$
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

<table>
<thead>
<tr>
<th>_strategy</th>
<th>( \frac{\phi(H_T)}{T} )</th>
<th>( \frac{\Omega(\log n)}{\Omega(\log^2 n)} )</th>
<th>( \Omega \left( \frac{1}{\log n} \right) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUT STRATEGY:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MATCHING STRATEGY:</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Cut Strategies: Finding Cuts Quickly

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

- $+$1 charge
- $-$1 charge

Random assignment of charge
Cut Strategies: Finding Cuts Quickly

After \( t \) iterations, \( H_t = \{ M_1, M_2, \ldots, M_t \} \).

- \( \bigcirc \) = +1 charge  
- \( \bullet \) = −1 charge

Random assignment of charge

Mix the charges along the matchings \( \{ M_1, M_2, \ldots, M_t \} \)

\((x+y)/2\)

\((x+y)^{1/2}\)
Cut Strategies: Finding Cuts Quickly

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

- Red dot = +1 charge
- Blue dot = -1 charge

Random assignment of charge
Cut Strategies: Finding Cuts Quickly

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

- $\bullet$ = +1 charge
- $\bigcirc$ = −1 charge

Random assignment of charge
Cut Strategies: Finding Cuts Quickly

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

- $\bullet$ = +1 charge  Random assignment of charge
- $\bullet$ = −1 charge

Mix the charges along the matchings $\{ M_1, M_2, \ldots, M_t \}$
Cut Strategies: Finding Cuts Quickly

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

- $\bullet = +1$ charge
- $\bullet = -1$ charge

Random assignment of charge

If cut is small, unbalance remains.

Mix the charges along the matchings $\{ M_1, M_2, \ldots, M_t \}$
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 $t$:

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

Lazy random walk traversing matchings in order.
The KRV mixing walk

KRV-walk

- At round $t$:

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

Lazy random walk traversing matchings in order.
The KRV mixing walk

KRV-walk

· At round $t$:

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

Lazy random walk traversing matchings in order.

Averaging along $M_2$
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 $M_1, M_2, …, M_k$ mixes perfectly on $S$ and $V-S$
and
no edge cross $(S,V-S)$
Why KRV cannot do better

SUPPOSE:
Walk on $M_1, M_2, \ldots, M_k$ mixes perfectly on $S$ and $V-S$
and
no edge cross $(S,V-S)$

Now walk mixes perfectly.
Expansion is $O(1)$.

Recall:
Approximation is
Can KRV get better than $O(1)$ expansion?
Our Cut Strategy: a Different Walk

**IDEA:** use lazy natural random walk

\[
P(t) = \frac{1}{2} + \frac{M_1 + M_2 + \ldots + M_{t-1}}{2(t - 1)}
\]

**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

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

**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:

\[ N_i = \frac{d}{d+1} I + \frac{1}{d+1} M_i \]

\[ P(t) = (N_1 N_2 \ldots N_{t-1} N_{t-1} N_{t-2} \ldots N_1)^d \]

Apply matrix inequality:

\[ \left\| (ABA)^t \right\| \leq \left\| A^t B^t A^t \right\| \]
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:

\[ N_i = \frac{d}{d+1} I + \frac{1}{d+1} M_i \]

\[ P(t) = (N_1 N_2 \ldots N_{t-1} N_{t-1} N_{t-2} \ldots N_1)^d \]

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:
\[ N_i = \frac{d}{d+1}I + \frac{1}{d+1}M_i \]

Apply matrix inequality:
\[ \left\| (ABA)^t \right\| \leq \left\| A^t B^t A^t \right\| \]

\[ \Psi_t = \Psi_{t-1} - L(M_t) \cdot P(t) \]
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 \( \mathcal{O}((\log n)^2) \) rounds,
conductance \( (1/\log n) \) by Cheeger.

\( \Omega(\log n) \) expansion in \( \mathcal{O}((\log n)^2) \) rounds.

**TIME:** only polylog factors worse than KRV
Matching player yielding

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

against any \textbf{Cut player}.

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

\textbf{in KRV Cut-Matching game}
Lower Bound Idea

A NAÏVE MATCHING PLAYER:

Fix a cut \((S,V-S)\). Keep it as sparse as possible.
A NAÏVE MATCHING PLAYER:

Fix a cut \((S,V-S)\). Keep it sparse.
Lower Bound Idea

A NAÏVE MATCHING PLAYER:

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

Cut player plays...

GAME OVER
A NAÏVE MATCHING PLAYER:
Fix a cut \((S, V-S)\). Keep it sparse.

**IDEA:** hedge over many cuts
Lower Bound Idea

THE REAL PLAYER - AT START:
Matching player selects $\log(n)$ ‘orthogonal’ 50-50 cuts in $V$.

THE REAL PLAYER - THROUGHOUT THE GAME:
Matching player adds matchings to *minimize average expansion*.
Main Lemma

\[ \forall \text{ 50-50 cut } (S, V-S), \]

\[ H_d = \left\{ -1, +1 \right\}^d \]
Main Lemma

∀ 50-50 cut \((S,V-S)\),

∃ a perfect matching \(M\), s.t.

\[ H_d = \{-1,+1\}^d \]
Main Lemma

∀ 50-50 cut \((S, V - S)\),

∃ 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?