Gadgets and Anti-Gadgets Leading to a Complexity Dichotomy

Tyson Williams
University of Wisconsin-Madison

Joint with:
Jin-Yi Cai (University of Wisconsin-Madison)
Michael Kowalczyk (Northern Michigan University)

To appear at ITCS 2012
Definition

A vertex cover of a graph is a set of vertices such that each edge of the graph is incident to at least one vertex in the set.
A vertex cover of a graph is a set of vertices such that each edge of the graph is incident to at least one vertex in the set.
A vertex cover of a graph is a set of vertices such that each edge of the graph is incident to at least one vertex in the set.
A vertex cover of a graph is a set of vertices such that each edge of the graph is incident to at least one vertex in the set.
$G = (V, E)$
Systematic Approach to $\texttt{\#VERTEXCOVER}$

- $G = (V, E)$
Systematic Approach to \textit{\#VertexCover}

- $G = (V, E)$
- $\sigma : V \rightarrow \{0, 1\}$
Systematic Approach to \(#\text{VertexCover}\)

- $G = (V, E)$
- $\sigma : V \rightarrow \{0, 1\}$
Systematic Approach to \#\textsc{VertexCover}

- $G = (V, E)$
- $\sigma : V \rightarrow \{0, 1\}$

\[
\prod_{(u,v) \in E} \text{OR}(\sigma(u), \sigma(v)) = 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 = 1
\]
Systematic Approach to \#\textsc{VertexCover}

- $G = (V, E)$
- $\sigma : V \rightarrow \{0, 1\}$

\[ \prod_{(u,v) \in E} \text{OR}(\sigma(u), \sigma(v)) = 1 \cdot 1 \cdot 0 \cdot 1 \cdot 1 \cdot 1 = 0 \]
Systematic Approach to \#VertexCover

- \( G = (V, E) \)
- \( \sigma : V \rightarrow \{0, 1\} \)

\[
\#\text{VertexCover}(G) = \sum_{\sigma : V \rightarrow \{0, 1\}} \prod_{(u, v) \in E} \text{OR}(\sigma(u), \sigma(v))
\]
\[
\sum_{\sigma:V \rightarrow \{0,1\}} \prod_{(u,v) \in E} \text{OR}(\sigma(u), \sigma(v))
\]
Generalize

\[ \sum_{\sigma:V \rightarrow \{0,1\}} \prod_{(u,v) \in E} \text{OR} (\sigma(u), \sigma(v)) \]

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>(p)</td>
<td>(q)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Generalize

\[ \sum \prod_{\sigma: V \rightarrow \{0,1\}, (u,v) \in E} f(\sigma(u), \sigma(v)) \]

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
<td>( q )</td>
</tr>
<tr>
<td>0 0</td>
<td>0</td>
</tr>
<tr>
<td>0 1</td>
<td>1</td>
</tr>
<tr>
<td>1 0</td>
<td>1</td>
</tr>
<tr>
<td>1 1</td>
<td>1</td>
</tr>
</tbody>
</table>

where \( w, x, y, z \in \mathbb{C} \)
Partition Function: $Z(\cdot)$

$$Z(G) = \sum_{\sigma: V \rightarrow \{0,1\}} \prod_{(u,v) \in E} f(\sigma(u),\sigma(v))$$

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$</td>
<td>$q$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

where $w, x, y, z \in \mathbb{C}$
Main Result

Theorem (Dichotomy Theorem)

Over 3-regular graphs $G$, the counting problem for any (binary) complex-weighted function $f$

$$Z(G) = \sum_{\sigma: V \rightarrow \{0,1\}} \prod_{(u,v) \in E} f(\sigma(u), \sigma(v))$$

is either computable in polynomial time or \#P-hard.
Main Result

**Theorem (Dichotomy Theorem)**

Over 3-regular graphs $G$, the counting problem for any (binary) complex-weighted function $f$

$$Z(G) = \sum_{\sigma: V \rightarrow \{0,1\}} \prod_{(u,v) \in E} f(\sigma(u), \sigma(v))$$

is either computable in polynomial time or $\#P$-hard. Furthermore, the complexity is efficiently decidable.
Outline

1. Related work
2. Define Holant function
3. Proof sketch
   - Anti-gadgets
Related Work: Dichotomy Theorems

- **Symmetric $f$**
  - $f(0, 1) = f(1, 0)$
- **3-regular graphs with outputs in**
  - $\{0, 1\}$ [Cai, Lu, Xia 08]
  - $\{0, 1, -1\}$ [Kowalczyk 09]
  - $\mathbb{R}$ [Cai, Lu, Xia 09]
  - $\mathbb{C}$ [Cai, Kowalczyk 10]
- **$k$-regular graphs with outputs in**
  - $\mathbb{R}$ [Cai, Kowalczyk 10]
  - $\mathbb{C}$ [Cai, Kowalczyk 11]
Related Work: Dichotomy Theorems

- **Symmetric** $f$
  - $f(0, 1) = f(1, 0)$

- 3-regular graphs with outputs in
  - $\{0, 1\}$ [Cai, Lu, Xia 08]
  - $\{0, 1, -1\}$ [Kowalczyk 09]
  - $\mathbb{R}$ [Cai, Lu, Xia 09]
  - $\mathbb{C}$ [Cai, Kowalczyk 10]

- $k$-regular graphs with outputs in
  - $\mathbb{R}$ [Cai, Kowalczyk 10]
  - $\mathbb{C}$ [Cai, Kowalczyk 11]

**This work:**

- **Asymmetric** $f$
- 3-regular graphs with outputs in
  - $\mathbb{C}$
Definition of Holant Function

- Partition Function

\[ \sum_{\sigma: V \rightarrow \{0,1\}} \prod_{(u,v) \in E} f(\sigma(u), \sigma(v)) \]
Definition of Holant Function

- **Partition Function**
  - Assignments to vertices
  - Functions on edges

\[
\sum_{\sigma: V \rightarrow \{0, 1\}} \prod_{(u, v) \in E} f(\sigma(u), \sigma(v))
\]
Definition of Holant Function

- **Partition Function**
  - Assignments to vertices
  - Functions on edges

- **Holant Function**
  - Assignment to edges
  - Functions on vertices

\[
\sum_{\sigma: V \to \{0,1\}} \prod_{(u,v) \in E} f(\sigma(u), \sigma(v))
\]
Definition of Holant Function

- **Partition Function**
  - Assignments to vertices
  - Functions on edges

\[ \sum_{\sigma:V \rightarrow \{0,1\}} \prod_{(u,v) \in E} f(\sigma(u), \sigma(v)) \]

- **Holant Function**
  - Assignment to edges
  - Functions on vertices

\[ \sum_{\sigma:E \rightarrow \{0,1\}} \prod_{v \in V} g_v(\sigma | E(v)) \]
Definition of Holant Function

- Holant($\{f\} \mid \{=3\}$) is a counting problem defined over (2,3)-regular bipartite graphs.

Holant Function
- Assignment to edges
- Functions on vertices

\[
\sum_{\sigma: E \rightarrow \{0,1\}} \prod_{v \in V} g_v (\sigma | E(v))
\]
Holant($\{f\} | \{=3\}$) is a counting problem defined over (2,3)-regular bipartite graphs.

- Degree 2 vertices take $f$.
- Degree 3 vertices take $=3$.

\[
\sum_{\sigma : E \to \{0,1\}} \prod_{v \in V} g_v (\sigma | E(v))
\]
Holant(\{\text{OR}_2\} \mid \{=3\}) \text{ is } \#\text{VERTEXCOVER on 3-regular graphs}.
Example Holant Problems

- Holant($\{\text{OR}_2\} | \{=3\}$) is $\#\text{VERTEXCOVER}$ on 3-regular graphs.

- Holant($\{\text{NAND}_2\} | \{=3\}$) is $\#\text{INDEPENDENTSET}$ on 3-regular graphs.
Example Holant Problems

- Holant($\{\text{OR}_2\} | \{=3\}$) is $\#\text{VERTEXCOVER}$ on 3-regular graphs.
- Holant($\{\text{NAND}_2\} | \{=3\}$) is $\#\text{INDEPENDENTSET}$ on 3-regular graphs.
- Holant($\{=2\} | \{\text{AT-MOST-ONE}\}$) is $\#\text{MATCHING}$.
Example Holant Problems

- Holant(\{\text{OR}_2\} | \{=3\}) is \text{'\#VertexCover'} on 3-regular graphs.

- Holant(\{\text{NAND}_2\} | \{=3\}) is \text{'\#IndependentSet'} on 3-regular graphs.

- Holant(\{=2\} | \{\text{AT-MOST-ONE}\}) is \text{'\#Matching'}.

- Holant(\{=2\} | \{\text{EXACTLY-ONE}\}) is \text{'\#PerfectMatching'}. 
More generally, $\text{Holant}(\mathcal{G} | \mathcal{R})$ is a counting problem defined over bipartite graphs.
More generally, Holant($G | R$) is a counting problem defined over bipartite graphs.

\[
\sum_{\sigma: E \rightarrow \{0,1\}} \prod_{v \in V} f_v (\sigma | E(v))
\]
Symmetric vs Asymmetric Function

\[
\begin{array}{ccc}
\text{Input} & \text{Output} \\
 p & q & f(p, q) \\
0 & 0 & w \\
0 & 1 & x \\
1 & 0 & y \\
1 & 1 & z \\
\end{array}
\]
Symmetric vs Asymmetric Function

Define \( p \) to be on the tail
Define \( q \) to be on the head

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
<td>( q )</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Symmetric vs Asymmetric Function

- **(2,3)-regular**

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>q</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- **Directed 3-regular**

  - Define \( p \) to be on the tail
  - Define \( q \) to be on the head
Strategy for Proving \#P-hardness

- \#\text{VertexCover} is \#P-hard over 3-regular graphs.
- Holant(\{\text{OR}_2\} \mid \{=3\}) is \#\text{VertexCover} on 3-regular graphs.
Strategy for Proving $\#P$-hardness

- $\#\text{VertexCover}$ is $\#P$-hard over 3-regular graphs.
- Holant($\{\text{OR}_2\} \mid \{=3\}$) is $\#\text{VertexCover}$ on 3-regular graphs.

Our problem is Holant($\{f\} \mid \{=3\}$).
Goal: simulate OR$_2$ using $f$. 
Strategy for Proving \( \#P \)-hardness

- \( \#\text{VertexCover} \) is \( \#P \)-hard over 3-regular graphs.
- \( \text{Holant}(\{\text{OR}_2\} \mid \{=3\}) \) is \( \#\text{VertexCover} \) on 3-regular graphs.

Our problem is \( \text{Holant}(\{f\} \mid \{=3\}) \).

Goal: simulate \( \text{OR}_2 \) using \( f \).

First step:

\[
\text{Holant}(\{\text{OR}_2\} \mid \{=3\}) \leq^P \text{Holant}(\{f\} \cup \mathcal{U} \mid \{=3\})
\]

where \( \mathcal{U} \) is the set of all unary functions.
Strategy for Proving $\#P$-hardness

- $\texttt{#VertexCover}$ is $\#P$-hard over 3-regular graphs.
- $\text{Holant}(\{\text{OR}_2\} \mid \{=3\})$ is $\texttt{#VertexCover}$ on 3-regular graphs.

Our problem is $\text{Holant}(\{f\} \mid \{=3\})$.

Goal: simulate $\text{OR}_2$ using $f$.

First step:

$$\text{Holant}(\{\text{OR}_2\} \mid \{=3\}) \leq^P \text{Holant}(\{f\} \cup \mathcal{U} \mid \{=3\})$$

where $\mathcal{U}$ is the set of all unary functions.

Second step:

$$\text{Holant}(\{f\} \cup \mathcal{U} \mid \{=3\}) \leq^P \text{Holant}(\{f\} \mid \{=3\})$$
Strategy for Proving \( \#P \)-hardness

- \( \#\text{VertexCover} \) is \( \#P \)-hard over 3-regular graphs.
- \( \text{Holant}(\{\text{OR}_2\} | \{=3\}) \) is \( \#\text{VertexCover} \) on 3-regular graphs.

Our problem is \( \text{Holant}(\{f\} | \{=3\}) \).
Goal: simulate \( \text{OR}_2 \) using \( f \).

First step:

\[
\text{Holant}(\{\text{OR}_2\} | \{=3\}) \leq^P \text{Holant}(\{f\} \cup \mathcal{U} | \{=3\})
\]

where \( \mathcal{U} \) is the set of all unary functions.

Second step:

\[
\text{Holant}(\{f\} \cup \mathcal{U} | \{=3\}) \leq^P \text{Holant}(\{f\} | \{=3\})
\]

Obtain \( \mathcal{U} \) via interpolation.
Interpolation

- A degree $n$ polynomial is uniquely defined by

\[ g_i(0) \quad g_i(1) . \]

Distinct evaluation points $\iff$ unary functions pairwise linearly independent (as length-2 vectors).
A degree $n$ polynomial is uniquely defined by $n + 1$ coefficients.
A degree $n$ polynomial is uniquely defined by
- $n + 1$ coefficients, or
- evaluations at $n + 1$ (different) points.
Interpolation

- A degree $n$ polynomial is uniquely defined by
  - $n + 1$ coefficients, or
  - evaluations at $n + 1$ (different) points.

- Interpolation is the process of converting from evaluations to coefficients.
Interpolation

- A degree \( n \) polynomial is uniquely defined by
  - \( n + 1 \) coefficients, or
  - evaluations at \( n + 1 \) (different) points.

- Interpolation is the process of converting from evaluations to coefficients.

- We construct unary functions \( g_i \) such that the evaluation points are
  \[
  \frac{g_i(0)}{g_i(1)}.
  \]
Interpolation

- A degree $n$ polynomial is uniquely defined by
  - $n + 1$ coefficients, or
  - evaluations at $n + 1$ (different) points.

- Interpolation is the process of converting from evaluations to coefficients.

- We construct unary functions $g_i$ such that the evaluation points are $\frac{g_i(0)}{g_i(1)}$.

- Distinct evaluation points $\iff$ unary functions pairwise linearly independent (as length-2 vectors).
Construction of Unary Functions

Projective Gadget

Recursive Gadget

Unary Function
Matrix Representation

- Left side indexes the row.
- Right side indexes the column.
- High order bit on top.

\[
\begin{bmatrix}
  w & x & y & z \\
  \otimes 2 \\
  \begin{bmatrix}
    w & 0 & 0 & 0 \\
    0 & 0 & 0 \\
    0 & 0 & 0 \\
    0 & 0 & 0 & z
  \end{bmatrix}
\end{bmatrix}
\]

Matrix of the composition is the product of the component matrices.
Matrix Representation

- Left side indexes the row.
- Right side indexes the column.
- High order bit on top.

\[
\begin{bmatrix}
w & x \\
y & z
\end{bmatrix} \otimes 2
\begin{bmatrix}
w & 0 & 0 & 0 \\
0 & x & 0 & 0 \\
0 & 0 & y & 0 \\
0 & 0 & 0 & z
\end{bmatrix}
\]
Matrix Representation

- Left side indexes the row.
- Right side indexes the column.
- High order bit on top.

\[
\begin{bmatrix}
  w & x \\
  y & z
\end{bmatrix} \otimes 2
\begin{bmatrix}
  w & 0 & 0 & 0 \\
  0 & x & 0 & 0 \\
  0 & 0 & y & 0 \\
  0 & 0 & 0 & z
\end{bmatrix}
\]
Matrix Representation

- Left side indexes the row.
- Right side indexes the column.
- High order bit on top.

Matrix of the composition is the product of the component matrices.
Matrix Representation

- Left side indexes the row.
- Right side indexes the column.
- High order bit on top.

Matrix of the composition is the product of the component matrices.
Want set of matrix powers to form an infinite set of pairwise linearly independent matrices.
Anti-Gadget Construction

- Want set of matrix powers to form an infinite set of pairwise linearly independent matrices.
- If this matrix has this property, then we are done.

\[
\begin{bmatrix}
w & x \\
y & z
\end{bmatrix} \otimes^2 \begin{bmatrix}
w & 0 & 0 & 0 \\
0 & x & 0 & 0 \\
0 & 0 & y & 0 \\
0 & 0 & 0 & z
\end{bmatrix}
\]
Anti-Gadget Construction

- Want set of matrix powers to form an infinite set of pairwise linearly independent matrices.
- If this matrix has this property, then we are done.

\[
\begin{bmatrix}
w & x \\
y & z
\end{bmatrix} \otimes 2
\begin{bmatrix}
w & 0 & 0 & 0 \\
0 & x & 0 & 0 \\
0 & 0 & y & 0 \\
0 & 0 & 0 & z
\end{bmatrix}
\]

- Otherwise, some power \( k \) is a multiple of the identity matrix.
Anti-Gadget Construction

- Want set of matrix powers to form an infinite set of pairwise linearly independent matrices.
- If this matrix has this property, then we are done.

\[
\begin{bmatrix}
w & x \\
y & z
\end{bmatrix} \otimes^2 \begin{bmatrix}
w & 0 & 0 & 0 \\
0 & x & 0 & 0 \\
0 & 0 & y & 0 \\
0 & 0 & 0 & z
\end{bmatrix}
\]

- Otherwise, some power $k$ is a multiple of the identity matrix.
- Using only $k - 1$ compositions creates an anti-gadget.
Want set of matrix powers to form an infinite set of pairwise linearly independent matrices.

If this matrix has this property, then we are done.

\[
\begin{bmatrix}
  w & x \\
  y & z
\end{bmatrix} \otimes^2
\begin{bmatrix}
w & 0 & 0 & 0 \\
0 & x & 0 & 0 \\
0 & 0 & y & 0 \\
0 & 0 & 0 & z
\end{bmatrix}
\]

Otherwise, some power $k$ is a multiple of the identity matrix.

Using only $k - 1$ compositions creates an anti-gadget.
Want set of matrix powers to form an infinite set of pairwise linearly independent matrices.

If this matrix has this property, then we are done.

\[
\begin{bmatrix}
w & x \\
y & z
\end{bmatrix} \otimes^2 
\begin{bmatrix}
w & 0 & 0 & 0 \\
0 & x & 0 & 0 \\
0 & 0 & y & 0 \\
0 & 0 & 0 & z
\end{bmatrix}
\]

Otherwise, some power \( k \) is a multiple of the identity matrix.

Using only \( k - 1 \) compositions creates an anti-gadget.
Anti-Gadget Technique

\[
\begin{pmatrix}
  w & 0 & 0 & 0 \\
  0 & x & 0 & 0 \\
  0 & 0 & y & 0 \\
  0 & 0 & 0 & z
\end{pmatrix}^{-1}
\begin{pmatrix}
  w \\
  x \\
  y \\
  z
\end{pmatrix}^{-1}
\]
The composition of these two gadgets yields...

\[
\begin{pmatrix}
w & 0 & 0 & 0 \\
0 & x & 0 & 0 \\
0 & 0 & y & 0 \\
0 & 0 & 0 & z
\end{pmatrix}
\]

\(\otimes 2\)

\[
\begin{pmatrix}
w & x \\
y & z
\end{pmatrix}
\]

\(\otimes 2\)

\[
\begin{pmatrix}
w & 0 & 0 & 0 \\
0 & y & 0 & 0 \\
0 & 0 & x & 0 \\
0 & 0 & 0 & z
\end{pmatrix}
\]
The composition of these two gadgets yields...
The composition of these two gadgets yields...
The First Anti-Gadget Lemma

Lemma

For \( w, x, y, z \in \mathbb{C} \), if

- \( wz \neq xy \),
- \( wxyz \neq 0 \), and
- \( |x| \neq |y| \),

then there exists a recursive gadget whose matrix powers form an infinite set of pairwise linearly independent matrices.

Corollary

For \( w, x, y, z \in \mathbb{C} \) as above,

\[
\text{Holant}\left(\{ f \} \big| \lambda = 3\right) \]

is \#P-hard.
The First Anti-Gadget Lemma

Lemma

For $w, x, y, z \in \mathbb{C}$, if

- $wz \neq xy$,
- $wxyz \neq 0$, and
- $|x| \neq |y|$,

then there exists a recursive gadget whose matrix powers form an infinite set of pairwise linearly independent matrices.

Corollary

For $w, x, y, z \in \mathbb{C}$ as above, Holant($\{f\} | \{=3\}$) is $#P$-hard.
Thank You
Thank You

Paper and slides available on my website.
www.cs.wisc.edu/~tdw