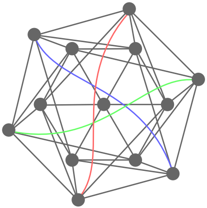
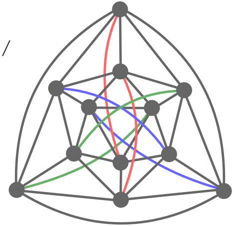
Combinatorics Using Computational Methods

Derrick Stolee

University of Nebraska–Lincoln
s-dstolee1@math.unl.edu
http://www.math.unl.edu/∼s-dstolee1/





March 13, 2012
Dissertation Defense

Supported by NSF grants DMS-0354008, DMS-0914815, and CCF-0916525,

and a University of Nebraska Presidential Fellowship.

Advisors and Committee

Stephen Hartke
Mathematics

Vinod Variyam
Computer Science and Engineering

Jamie Radcliffe
Mathematics

Stephen Scott
CSE

Christina Falci
Sociology

Thanks to...

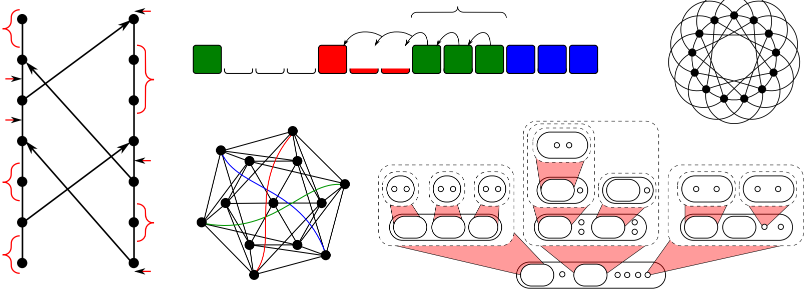
Katie Stolee

Eric Allender, Pranav Anand, David Barrington, Brian Bockleman, Chris Bourke, Jane Butterfield, James Carraher, Henry Escuadro, Michael Ferrara, Lance Fortnow, Brady Garvin, Ralucca Gera, Joe Geisbauer, Ellen Gethner, Steve Goddard, Adam S. Jobson, Travis Johnston, André Kézdy, Elizabeth Kupin, Timothy D. LeSaulnier, Jared Nishikawa, Kevin G. Milans, Andrew Ray, Ben Reiniger, Tyler Seacrest, Hannah (Kolb) Spinoza, Brendon Stanton, David Swanson, Raghunath Tewari, Judy Walker, Derek Weitzel, Paul S. Wenger, Douglas B. West, Zahava Wilstein, Matthew Yancey, and

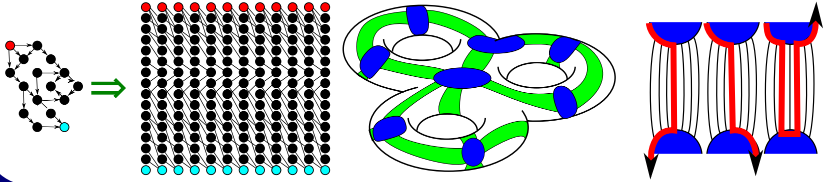
all UNL Math or CSE graduate students, staff, and faculty.

Full Acknowledgements available on my web page.

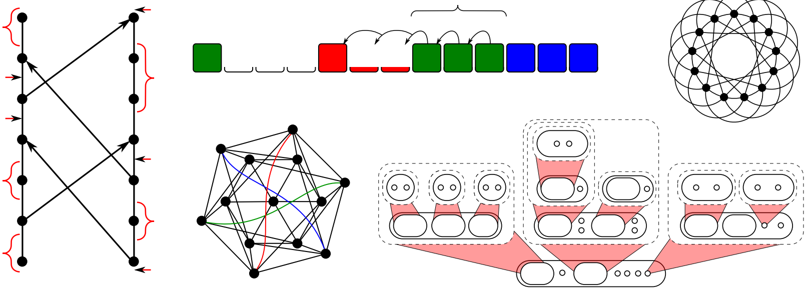
Computational Combinatorics



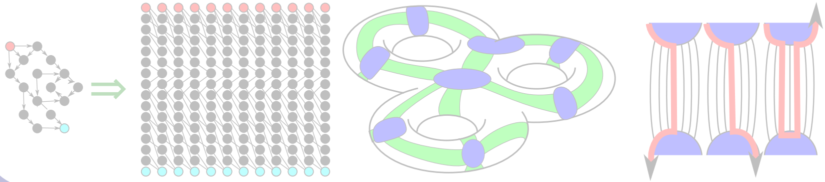
Computational Complexity



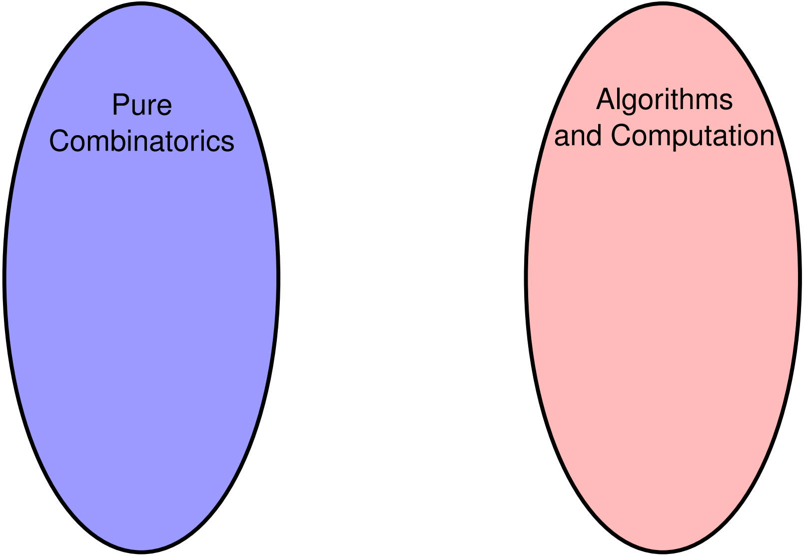
Computational Combinatorics



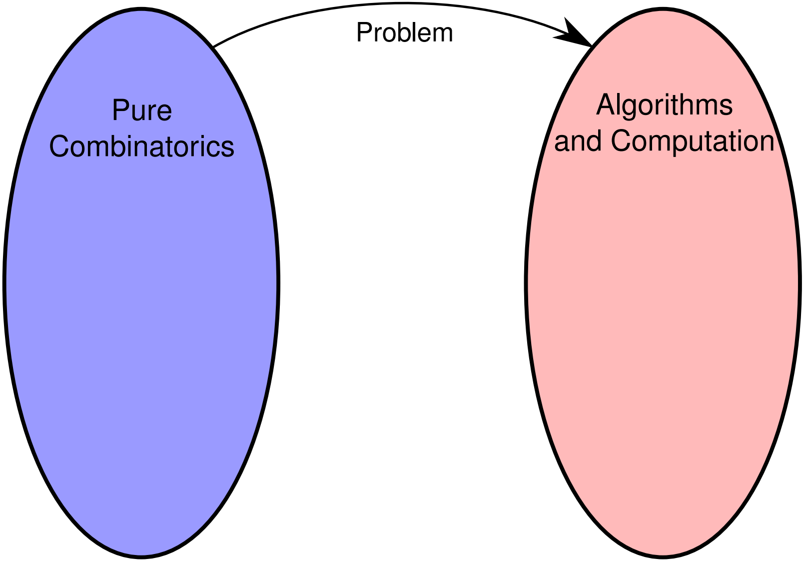
Computational Complexity



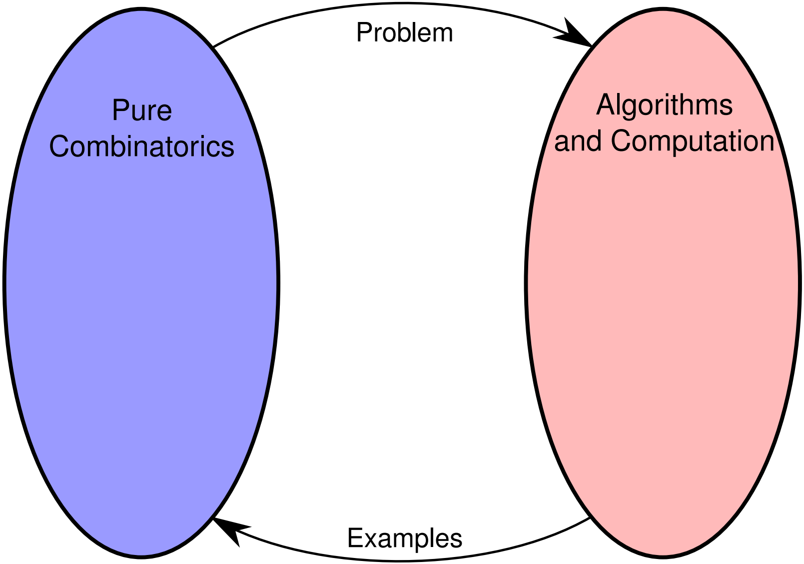
Computational Combinatorics



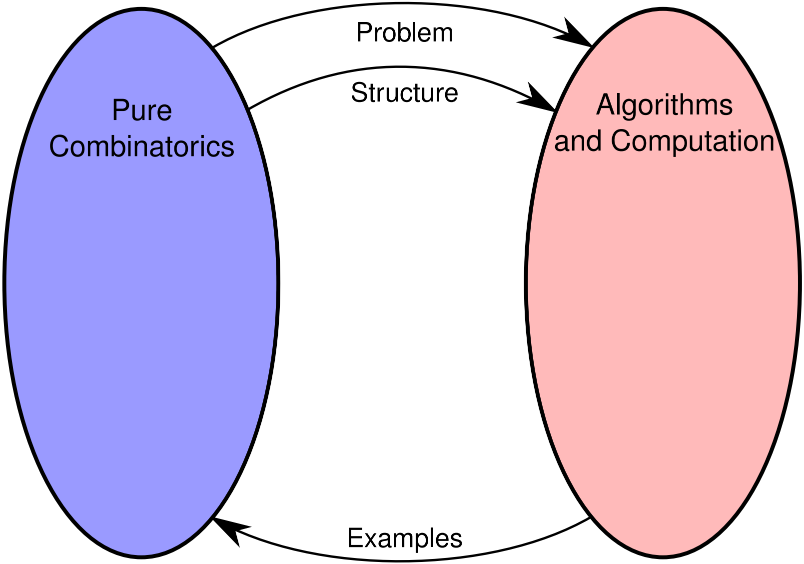
Computational Combinatorics



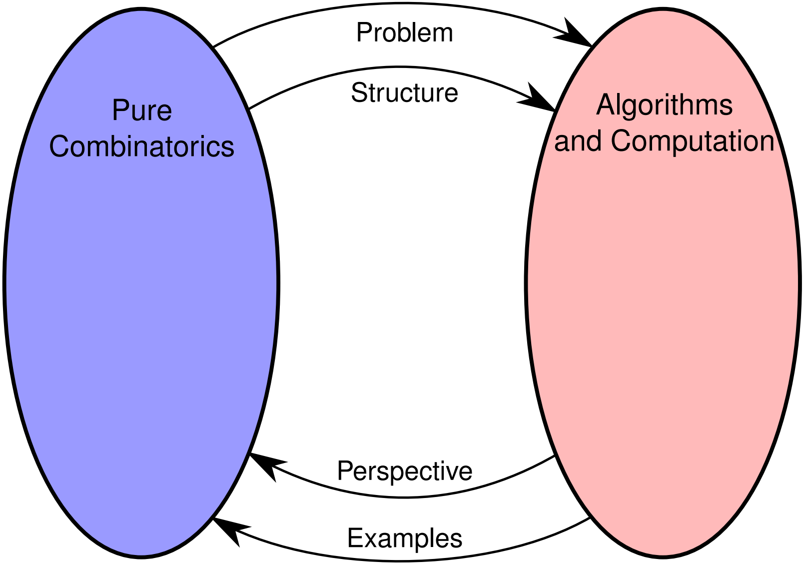
Computational Combinatorics



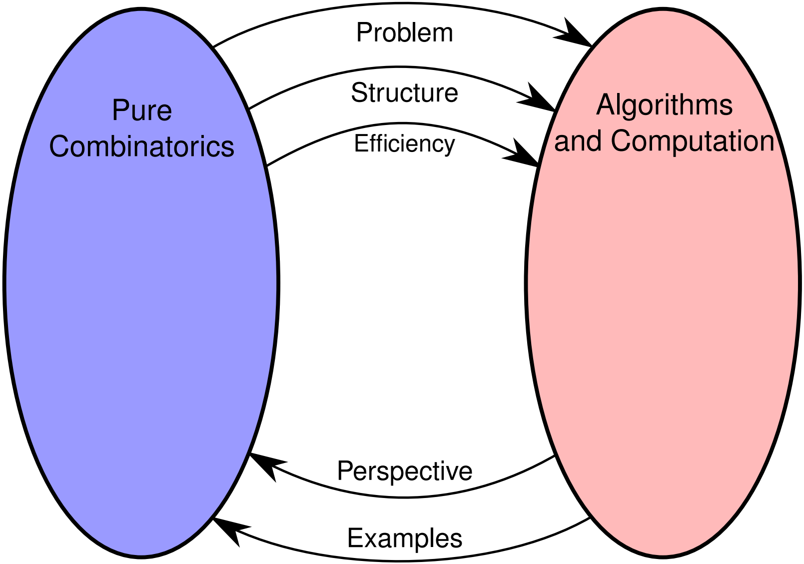
Computational Combinatorics



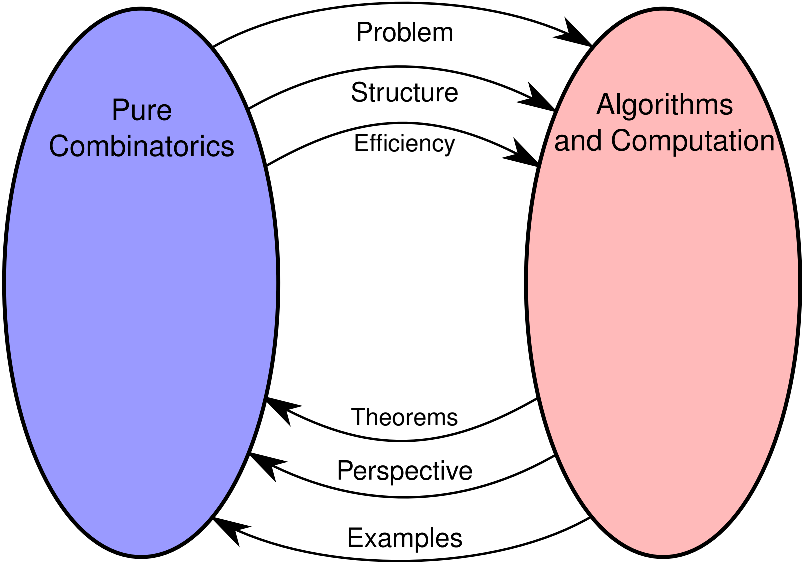
Computational Combinatorics



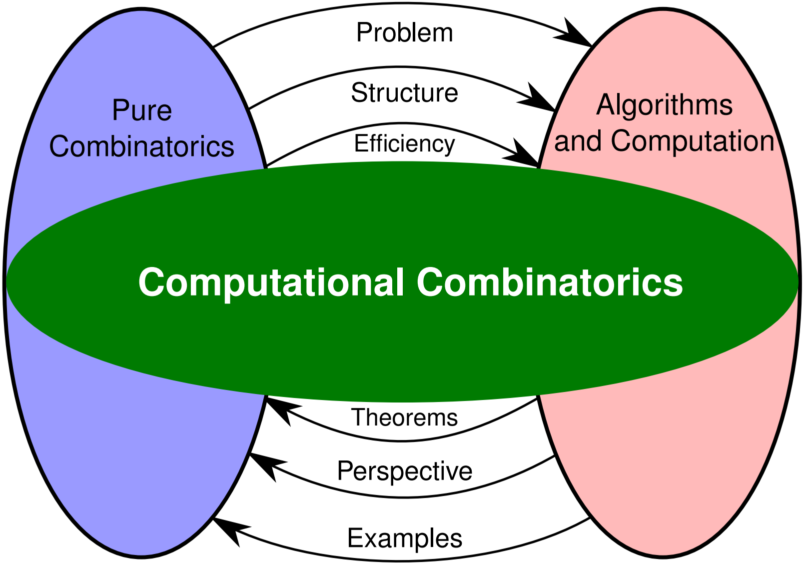
Computational Combinatorics



Computational Combinatorics



Computational Combinatorics



The Goal

Determine if certain **combinatorial objects** exist with given **structural** or **extremal** properties.

The Goal

Determine if certain **combinatorial objects** exist with given **structural** or **extremal** properties.

Examples:

1. Is there a **projective plane** of order 10?
2. When do **strongly regular graphs** exist?
3. How many **Steiner triple systems** are there of order 19?

The Goal

Determine if certain **combinatorial objects** exist with given **structural** or **extremal** properties.

Examples:

1. Is there a **projective plane** of order 10?
   (Lam, Thiel, Swiercz, 1989)
2. When do **strongly regular graphs** exist?
   (Spence 2000, Coolsaet, Degraer, Spence 2006, many others)
3. How many **Steiner triple systems** are there of order 19?
   (Kaski, Östergård, 2004)

Problems Tackled in This Thesis

1. Which numbers are representable as the number of chains in a width-two poset?
2. Which colorings of $\{1, \ldots, n\}$ avoid monochromatic progressions?
3. How many edges can exist in a graph with p perfect matchings?
4. What graphs are uniquely K_r-saturated?

Problems Tackled in This Thesis

1. Which numbers are representable as the number of chains in a width-two poset?
   (with Kupin, Reiniger) Chapter 4
2. Which colorings of $\{1, \ldots, n\}$ avoid monochromatic progressions?
   (with Jobson, Kézdy) Chapter 5
3. How many edges can exist in a graph with p perfect matchings?
   (with Hartke, West, Yancey) Chapter 9
4. What graphs are uniquely K_r-saturated?
   (with Hartke) Chapter 11

Problems Tackled in This Thesis

1. Which numbers are representable as the number of chains in a width-two poset?
   (with Kupin, Reiniger) Chapter 4
2. Which colorings of $\{1, \ldots, n\}$ avoid monochromatic progressions?
   (with Jobson, Kézdy) Chapter 5
3. How many edges can exist in a graph with p perfect matchings?
   (with Hartke, West, Yancey) Chapter 9
4. What graphs are uniquely K_r-saturated?
   (with Hartke) Chapter 11

Main Technique: Combinatorial Search

Goal: Determine if certain combinatorial objects exist with given structural or extremal properties.

Idea: Build objects *piece-by-piece* from *base examples* to enumerate all desired examples of a given order.

Main Technique: Combinatorial Search

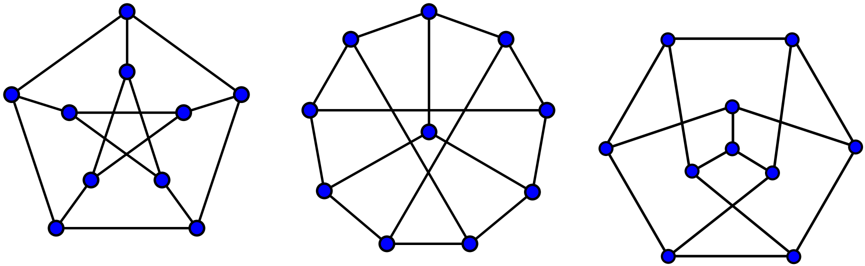
Goal: Determine if certain combinatorial objects exist with given structural or extremal properties.

Idea: Build objects *piece-by-piece* from *base examples* to enumerate all desired examples of a given order.

Most interesting properties are invariant under **isomorphism***.*

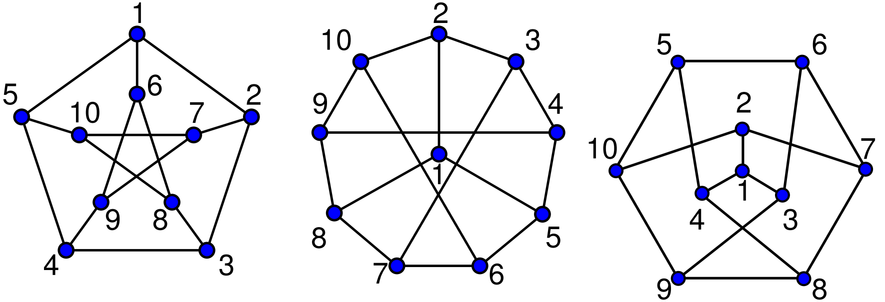
Combinatorial Object: Graphs

A **graph** G of **order** n is composed of a set $V(G)$ of n vertices and a set $E(G)$ of edges, where the edges are unordered pairs of vertices.



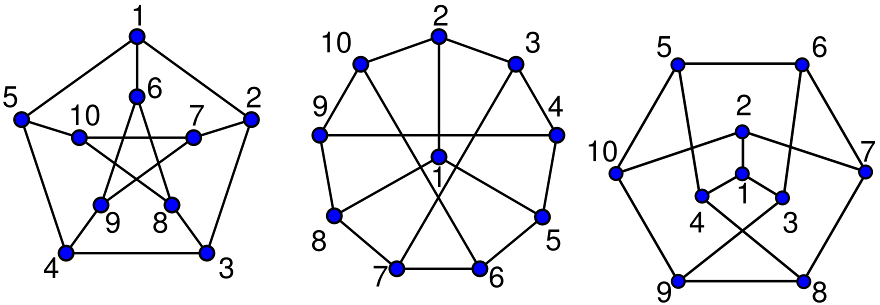
Combinatorial Object: Graphs

A **graph** G of **order** n is composed of a set $V(G)$ of n vertices and a set $E(G)$ of edges, where the edges are unordered pairs of vertices.



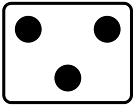
Combinatorial Object: Graphs

An **isomorphism** between G_1 and G_2 is a bijection from $V(G_1)$ to $V(G_2)$ that induces a bijection from $E(G_1)$ to $E(G_2)$.



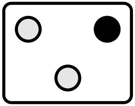
Example: Generating Graphs by Edges

We can build graphs starting at $\overline{K_n}$ by adding edges.



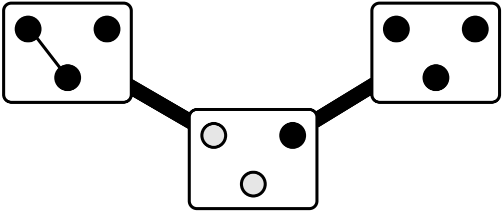
Example: Generating Graphs by Edges

We can build graphs starting at $\overline{K_n}$ by adding edges.



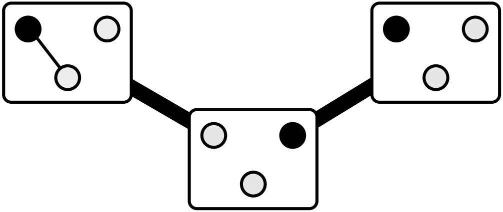
Example: Generating Graphs by Edges

We can build graphs starting at $\overline{K_n}$ by adding edges.



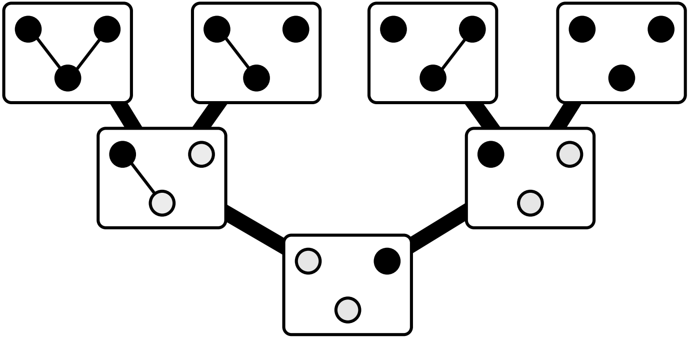
Example: Generating Graphs by Edges

We can build graphs starting at $\overline{K_n}$ by adding edges.



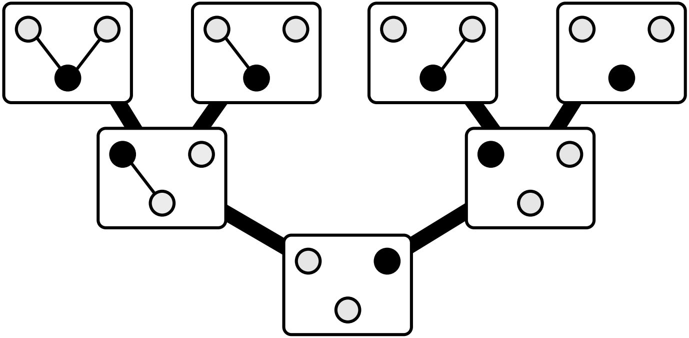
Example: Generating Graphs by Edges

We can build graphs starting at $\overline{K_n}$ by adding edges.



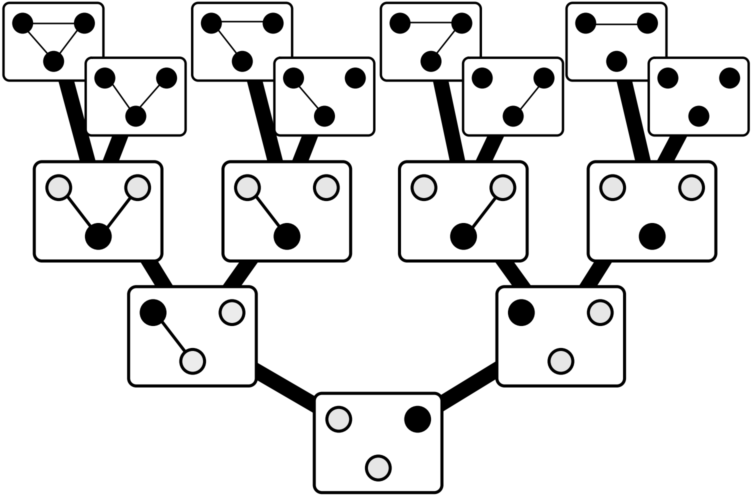
Example: Generating Graphs by Edges

We can build graphs starting at $\overline{K_n}$ by adding edges.



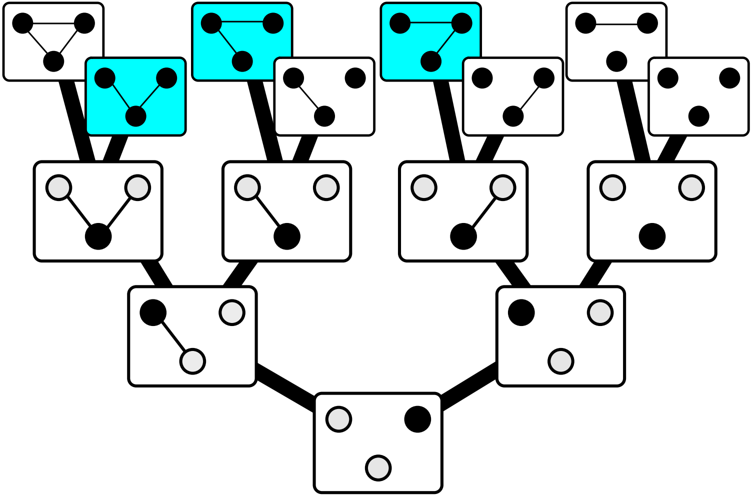
Example: Generating Graphs by Edges

We can build graphs starting at $\overline{K_n}$ by adding edges.



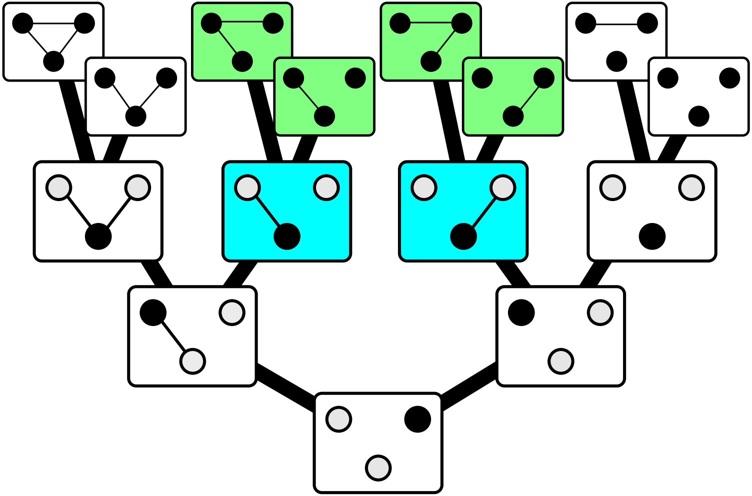
Example: Generating Graphs by Edges

We can build graphs starting at $\overline{K_n}$ by adding edges.



Example: Generating Graphs by Edges

We can build graphs starting at $\overline{K_n}$ by adding edges.



Two Techniques for Isomorphs

1. Canonical Deletion (McKay 1998)
   - Removes all isomorphs.
   - Not known how to integrate with constraint propagation.
   - High cost per object.

2. Orbital Branching (Ostrowski, Linderoth, Rossi, Smriglio 2007)
   - Removes some, but not all isomorphs.
   - Naturally integrates with constraint propagation.
   - Low cost per object.

Two Techniques for Isomorphs

1. Canonical Deletion (McKay 1998)
   - Removes all isomorphs.
   - Not known how to integrate with constraint propagation.
   - High cost per object.

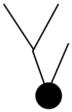
   Overview in Chapter 6

2. Orbital Branching (Ostrowski, Linderoth, Rossi, Smriglio 2007)
   - Removes some, but not all isomorphs.
   - Naturally integrates with constraint propagation.
   - Low cost per object.

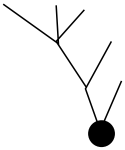
   Overview in Chapter 10

Search by Augmentations

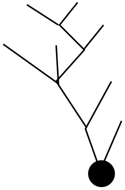
Search by Augmentations



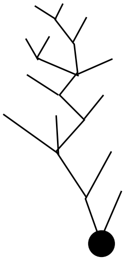
Search by Augmentations



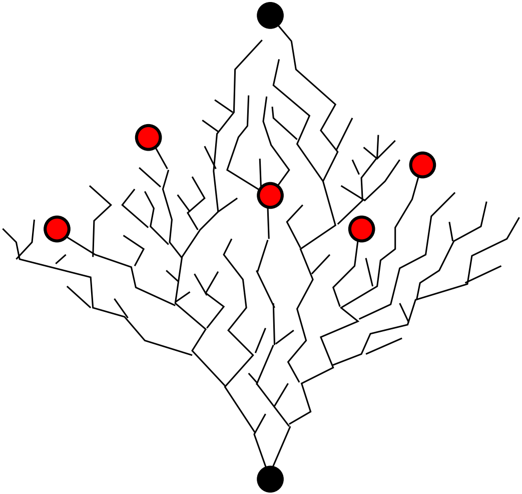
Search by Augmentations



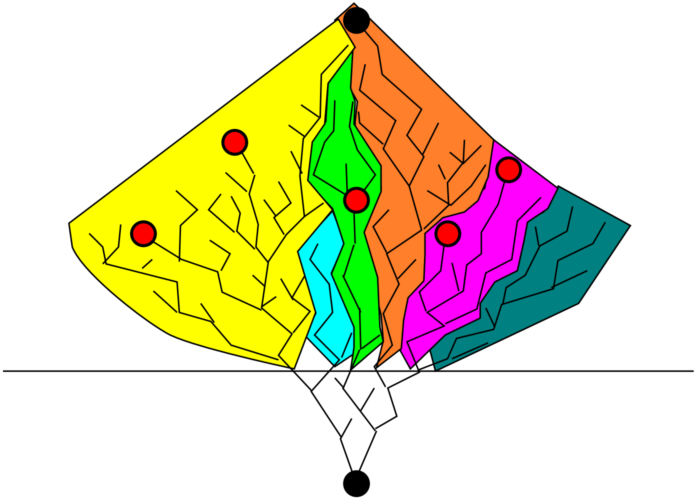
Search by Augmentations



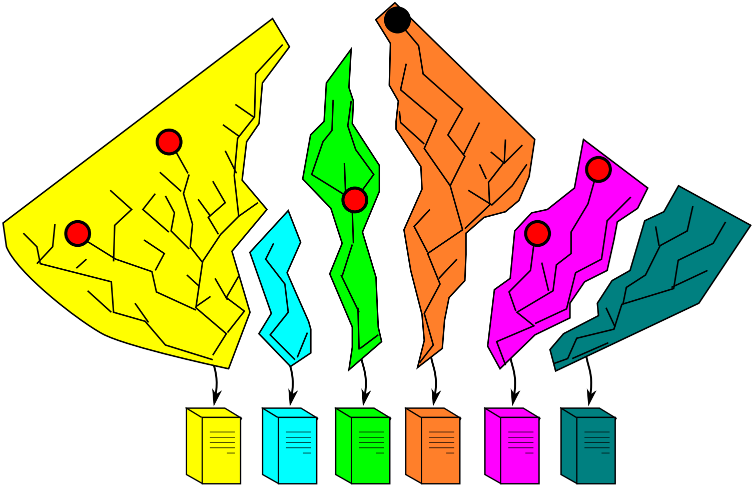
Search by Augmentations



Search by Augmentations



Search by Augmentations



Implementation

My **TreeSearch** library enables parallelization in the Condor scheduler.

Executes on the **Open Science Grid**, a collection of supercomputers around the country.

Open Science Grid

Problems Tackled in This Thesis

1. Which numbers are representable as the number of chains in a width-two poset?
   (with Kupin, Reiniger) Chapter 4
2. Which colorings of $\{1, \ldots, n\}$ avoid monochromatic progressions?
   (with Jobson, Kézdy) Chapter 5
3. How many edges can exist in a graph with p perfect matchings?
   (with Hartke, West, Yancey) Chapter 9
4. What graphs are uniquely K_r-saturated?
   (with Hartke) Chapter 11

Problems Tackled in This Thesis

1. Which numbers are representable as the number of chains in a width-two poset?
   (with Kupin, Reiniger) Chapter 4

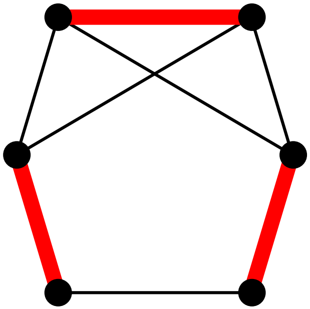
2. Which colorings of $\{1, \ldots, n\}$ avoid monochromatic progressions?
   (with Jobson, Kézdy) Chapter 5

3. How many edges can exist in a graph with p perfect matchings?
   (with Hartke, West, Yancey) Chapter 9

4. What graphs are uniquely K_r-saturated?
   (with Hartke) Chapter 11

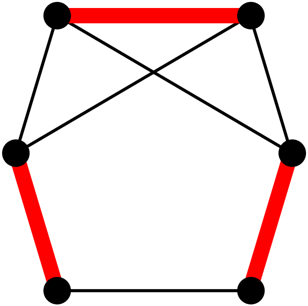
Perfect Matchings

A **perfect matching** is a set of edges which cover each vertex exactly once.



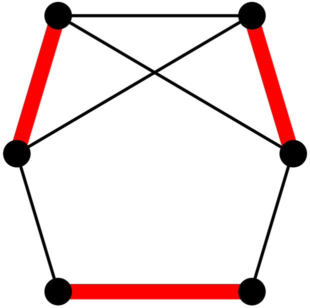
Perfect Matchings

A **perfect matching** is a set of edges which cover each vertex exactly once.
$\Phi(G)$ is the number of perfect matchings in the graph G.



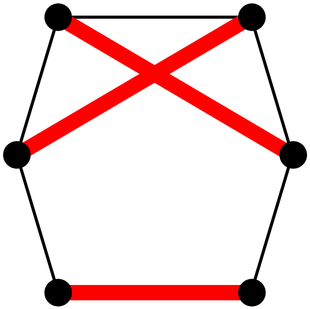
Perfect Matchings

A **perfect matching** is a set of edges which cover each vertex exactly once.
$\Phi(G)$ is the number of perfect matchings in the graph G.



Perfect Matchings

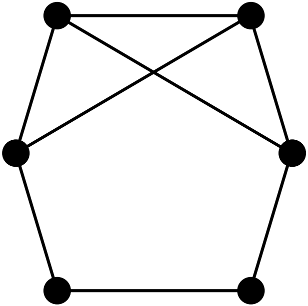
A **perfect matching** is a set of edges which cover each vertex exactly once.
$\Phi(G)$ is the number of perfect matchings in the graph G.



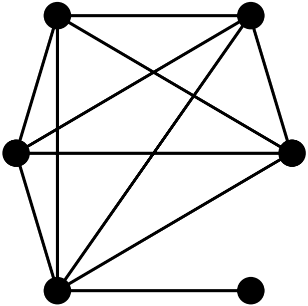
$\Phi(G) = 3$
8 edges

Perfect Matchings

A **perfect matching** is a set of edges which cover each vertex exactly once.



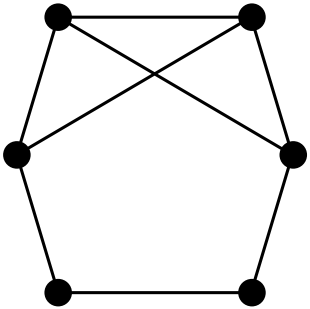
$\Phi(G) = 3$
8 edges



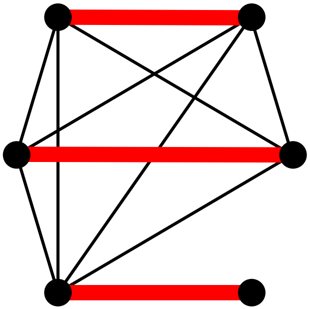
11 edges

Perfect Matchings

A **perfect matching** is a set of edges which cover each vertex exactly once.



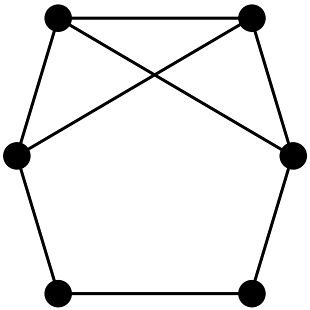
$\Phi(G) = 3$
8 edges



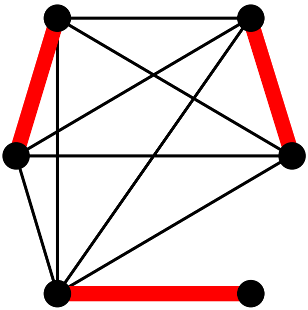
11 edges

Perfect Matchings

A **perfect matching** is a set of edges which cover each vertex exactly once.



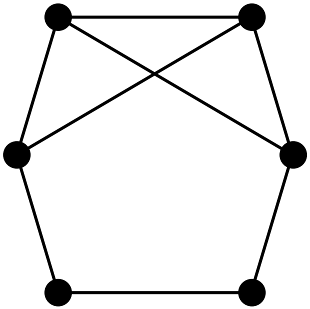
$\Phi(G) = 3$
8 edges



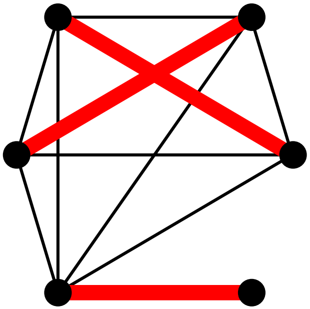
11 edges

Perfect Matchings

A **perfect matching** is a set of edges which cover each vertex exactly once.



$\Phi(G) = 3$
8 edges



$\Phi(G) = 3$
11 edges

Perfect Matchings

A **perfect matching** is a set of edges which cover each vertex exactly once.

Question (Dudek, Schmitt, 2010) What is the maximum number of edges in a graph with exactly n vertices and p perfect matchings?

Definition Let n be an even number and fix $p \geq 1$.

$$f(n,p) = \max\{|E(G)| : |V(G)| = n, \Phi(G) = p\}.$$

Graphs attaining this number of edges are ***p*-extremal**.

Hetyei's Theorem

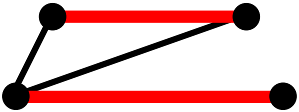
Theorem (Hetyei's Theorem, 1986) For all even $n \geq 2$,

$$f(n, 1) = \frac{n^2}{4}.$$

Hetyei's Theorem

Theorem (Hetyei's Theorem, 1986) For all even $n \geq 2$,

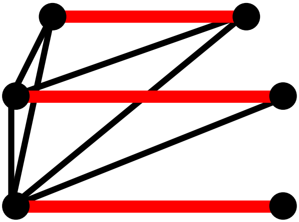
$$f(n, 1) = \frac{n^2}{4}.$$



Hetyei's Theorem

Theorem (Hetyei's Theorem, 1986) For all even $n \geq 2$,

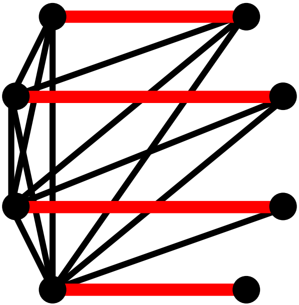
$$f(n, 1) = \frac{n^2}{4}.$$



Hetyei's Theorem

Theorem (Hetyei's Theorem, 1986) For all even $n \geq 2$,

$$f(n, 1) = \frac{n^2}{4}.$$



The Form of $f(n,p)$

Theorem (Dudek, Schmitt, 2010) For each p, there exist constants n_p, c_p so that for all $n \geq n_p$,

$$f(n,p) = \frac{n^2}{4} + c_p.$$

The Form of $f(n, p)$

Theorem (Dudek, Schmitt, 2010) For each p, there exist constants n_p, c_p so that for all $n \geq n_p$,

$$f(n, p) = \frac{n^2}{4} + c_p.$$

p	1	2	3	4	5	6
c_p	0	1	2	2	2	3
	H	Dudek, Schmitt, 2010				

Structure Theorem

Theorem (Hartke, Stolee, West, Yancey, 2011) For a fixed p, every graph G with n vertices, p perfect matchings, and $f(n, p) = \frac{n^2}{4} + c_p$ edges is composed of a finite list of **fundamental graphs** combined in specified ways.

Proof involves several classic structure theorems from matching theory in an extremal setting.

Structure Theorem

Theorem (Hartke, Stolee, West, Yancey, 2011) For a fixed p, every graph G with n vertices, p perfect matchings, and $f(n, p) = \frac{n^2}{4} + c_p$ edges is composed of a finite list of **fundamental graphs** combined in specified ways.

Proof involves several classic structure theorems from matching theory in an extremal setting.

For $p \leq 10$, the graphs have order at most 12.

Structure Theorem

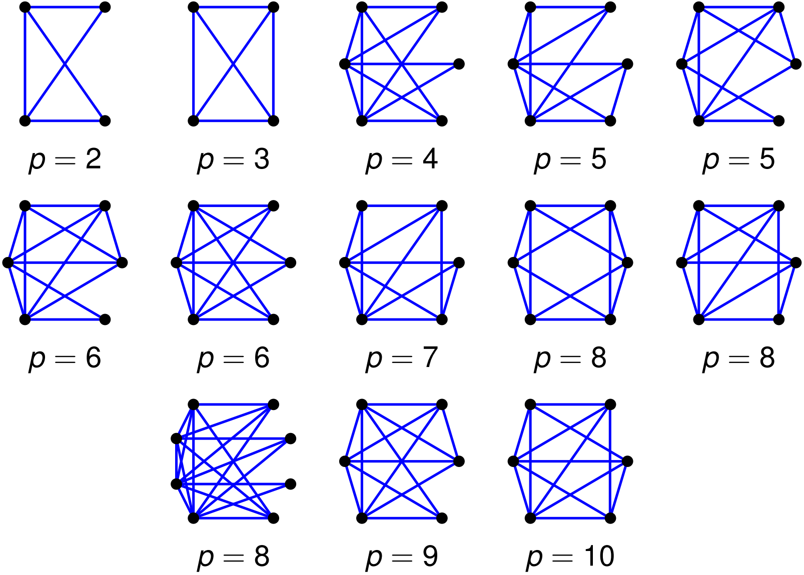
Theorem (Hartke, Stolee, West, Yancey, 2011) For a fixed p, every graph G with n vertices, p perfect matchings, and $f(n,p) = \frac{n^2}{4} + c_p$ edges is composed of a finite list of **fundamental graphs** combined in specified ways.

Proof involves several classic structure theorems from matching theory in an extremal setting.

For $p \leq 10$, the graphs have order at most 12.

Using standard software (McKay's *geng*) we found the graphs and computed c_p.

Fundamental Graphs for $2 \leq p \leq 10$



c_p for small p

p	1	2	3	4	5	6	7	8	9	10
c_p	0	1	2	2	2	3	3	3	4	4
	H	Dudek, Schmitt 2010					HSWY 2011			

c_p for small p

p	1	2	3	4	5	6	7	8	9	10
c_p	0	1	2	2	2	3	3	3	4	4
	H	Dudek, Schmitt 2010					HSWY 2011			

Q: Is c_p monotone in p?

Structural Theorem, Redux

Without more involved computational methods, brute force methods (such as *geng*) cannot go farther.

Structural Theorem, Redux

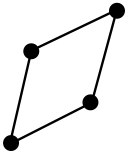
Without more involved computational methods, brute force methods (such as *geng*) cannot go farther.

The **Lovász Two Ear Theorem (1983)** provides a way to build fundamental graphs using **ear augmentations**.

Structural Theorem, Redux

Without more involved computational methods, brute force methods (such as *geng*) cannot go farther.

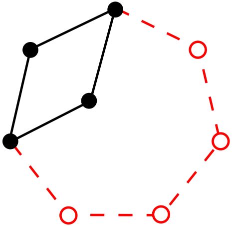
The **Lovász Two Ear Theorem (1983)** provides a way to build fundamental graphs using **ear augmentations**.



Structural Theorem, Redux

Without more involved computational methods, brute force methods (such as *geng*) cannot go farther.

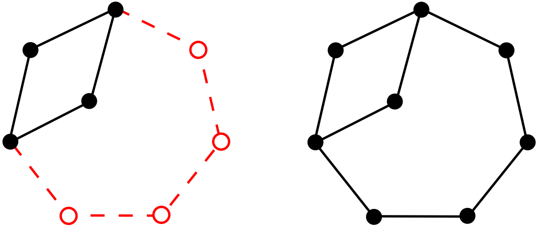
The **Lovász Two Ear Theorem (1983)** provides a way to build fundamental graphs using **ear augmentations**.



Structural Theorem, Redux

Without more involved computational methods, brute force methods (such as *geng*) cannot go farther.

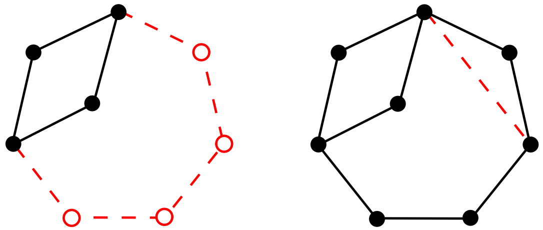
The **Lovász Two Ear Theorem (1983)** provides a way to build fundamental graphs using **ear augmentations**.



Structural Theorem, Redux

Without more involved computational methods, brute force methods (such as *geng*) cannot go farther.

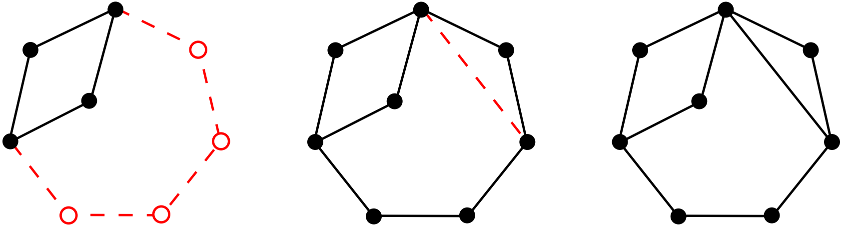
The **Lovász Two Ear Theorem (1983)** provides a way to build fundamental graphs using **ear augmentations**.



Structural Theorem, Redux

Without more involved computational methods, brute force methods (such as *geng*) cannot go farther.

The **Lovász Two Ear Theorem (1983)** provides a way to build fundamental graphs using **ear augmentations**.



Computational Method

Developed a computational method from:

1. **Augmentations:** Lovász Two Ear Theorem.
2. **Isomorphs:** Canonical Deletion. McKay
3. **Pruning:** Developed new structural and extremal theorems.

Computational Method

Developed a computational method from:

1. **Augmentations:** Lovász Two Ear Theorem.
2. **Isomorphs:** Canonical Deletion. McKay
3. **Pruning:** Developed new structural and extremal theorems.

Before: Stuck at $p \leq 10$ when searching on most 12 vertices.

Computational Method

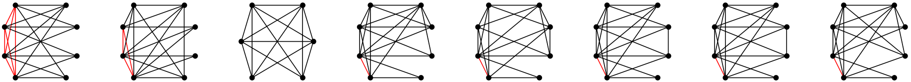
Developed a computational method from:

1. **Augmentations:** Lovász Two Ear Theorem.
2. **Isomorphs:** Canonical Deletion. McKay
3. **Pruning:** Developed new structural and extremal theorems.

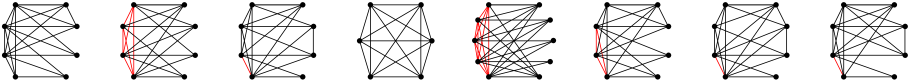
Before: Stuck at $p \leq 10$ when searching on most 12 vertices.

Now: Found graphs for all $p \leq 27$ on up to 22 vertices.

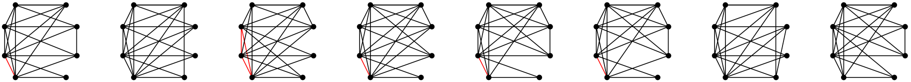
Fundamental Graphs for $11 \leq p \leq 27$



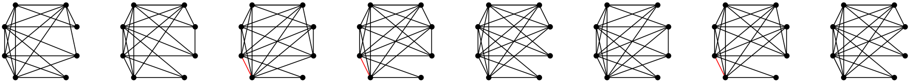
$p = 11$ $p = 11$ $p = 12$ $p = 13$ $p = 13$ $p = 13$ $p = 13$ $p = 13$



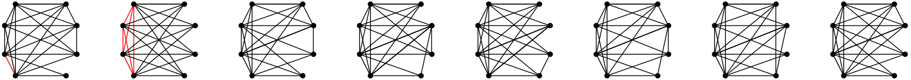
$p = 13$ $p = 14$ $p = 14$ $p = 15$ $p = 16$ $p = 16$ $p = 16$ $p = 16$



$p = 17$ $p = 17$ $p = 18$ $p = 18$ $p = 19$ $p = 19$ $p = 19$ $p = 19$



$p = 19$ $p = 19$ $p = 20$ $p = 21$ $p = 21$ $p = 21$ $p = 22$ $p = 23$



$p = 24$ $p = 24$ $p = 25$ $p = 25$ $p = 26$ $p = 26$ $p = 26$ $p = 27$

c_p for small p

p	1	2	3	4	5	6	7	8	9	10
c_p	0	1	2	2	2	3	3	3	4	4
	H	Dudek, Schmitt, 2010					HSWY, 2011			

p	11	12	13	14	15	16	17	18	19	20
c_p	3	5	3	4	6	4	4	5	4	5
	Stolee, 2011									

p	21	22	23	24	25	26	27
c_p	5	5	5	6	5	5	6
	Stolee, 2011						

c_p for small p

p	1	2	3	4	5	6	7	8	9	10
c_p	**0**	**1**	**2**	**2**	**2**	**3**	**3**	**3**	**4**	**4**
	H	Dudek, Schmitt, 2010					HSWY, 2011			

p	11	12	13	14	15	16	17	18	19	20
c_p	3	**5**	3	4	**6**	4	4	5	4	5
	Stolee, 2011									

p	21	22	23	24	25	26	27
c_p	5	5	5	**6**	5	5	**6**
	Stolee, 2011						

c_p not monotonic in p !

Blue numbers match conjectured upper bound.

Problems Tackled in This Thesis

1. Which numbers are representable as the number of chains in a width-two poset?
   (with Kupin, Reiniger) Chapter 4
2. Which colorings of $\{1, \ldots, n\}$ avoid monochromatic progressions?
   (with Jobson, Kézdy) Chapter 5
3. How many edges can exist in a graph with p perfect matchings?
   (with Hartke, West, Yancey) Chapter 9
4. What graphs are uniquely K_r-saturated?
   (with Hartke) Chapter 11

Problems Tackled in This Thesis

1. Which numbers are representable as the number of chains in a width-two poset?
   (with Kupin, Reiniger) Chapter 4

2. Which colorings of $\{1, \ldots, n\}$ avoid monochromatic progressions?
   (with Jobson, Kézdy) Chapter 5

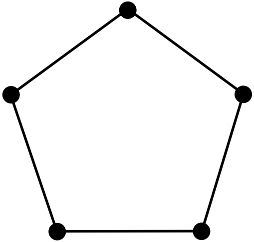
3. How many edges can exist in a graph with p perfect matchings?
   (with Hartke, West, Yancey) Chapter 9

4. What graphs are uniquely K_r-saturated?
   (with Hartke) Chapter 11

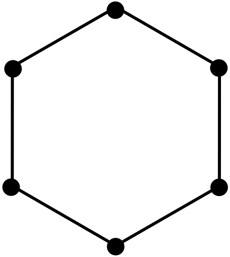
H-Saturated Graphs

Definition A graph G is ***H*-saturated** if

- G does not contain H as a subgraph. (***H*-free**)
- For every $e \in E(\overline{G})$, $G + e$ contains H as a subgraph.



5-cycle



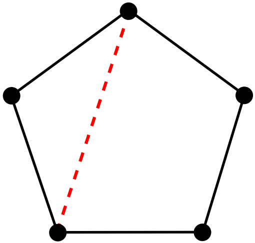
6-cycle

Example: $H = K_3$ where K_r is the **complete graph** on r vertices.

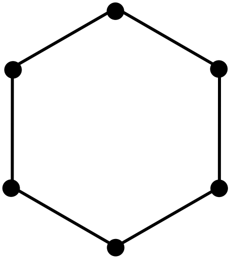
H-Saturated Graphs

Definition A graph G is ***H*-saturated** if

- G does not contain H as a subgraph. (***H*-free**)
- For every $e \in E(\overline{G})$, $G + e$ contains H as a subgraph.



5-cycle
is K_3-saturated



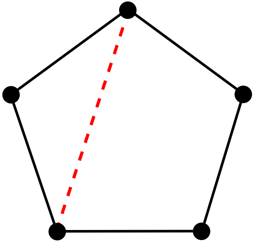
6-cycle

Example: $H = K_3$ where K_r is the **complete graph** on r vertices.

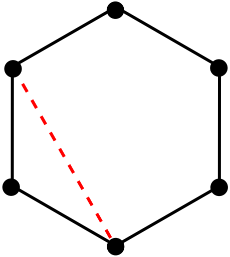
H-Saturated Graphs

Definition A graph G is ***H*-saturated** if

- G does not contain H as a subgraph. (***H*-free**)
- For every $e \in E(\overline{G})$, $G + e$ contains H as a subgraph.



5-cycle
is K_3-saturated



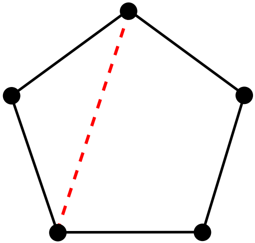
6-cycle

Example: $H = K_3$ where K_r is the **complete graph** on r vertices.

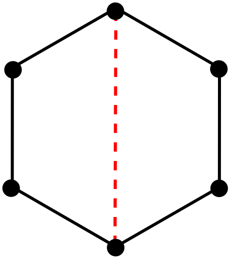
H-Saturated Graphs

Definition A graph G is ***H*-saturated** if

- G does not contain H as a subgraph. (***H*-free**)
- For every $e \in E(\overline{G})$, $G + e$ contains H as a subgraph.



5-cycle
is K_3-saturated



6-cycle
is **not** K_3-saturated

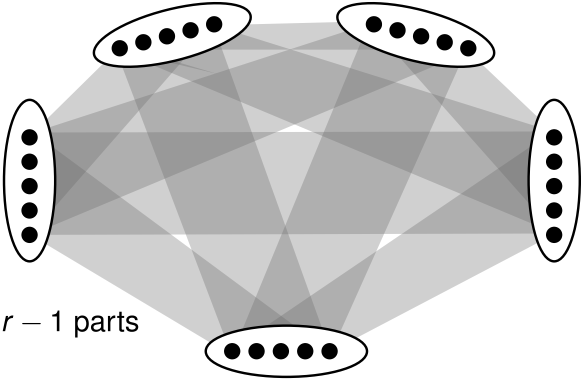
Example: $H = K_3$ where K_r is the **complete graph** on r vertices.

Turán's Theorem

Theorem (Turán, 1941) Let $r \geq 3$. If G is K_r-saturated on n vertices, then G has **at most** $\left(1 - \frac{1}{r-1}\right)\frac{n^2}{2}$ edges (asymptotically).

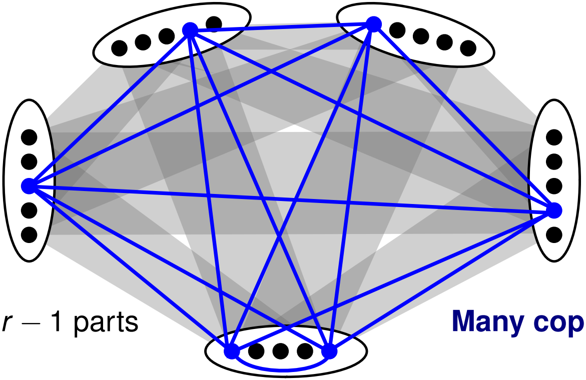
Turán's Theorem

Theorem (Turán, 1941) Let $r \geq 3$. If G is K_r-saturated on n vertices, then G has **at most** $\left(1 - \frac{1}{r-1}\right)\frac{n^2}{2}$ edges (asymptotically).



Turán's Theorem

Theorem (Turán, 1941) Let $r \geq 3$. If G is K_r-saturated on n vertices, then G has **at most** $\left(1 - \frac{1}{r-1}\right)\frac{n^2}{2}$ edges (asymptotically).

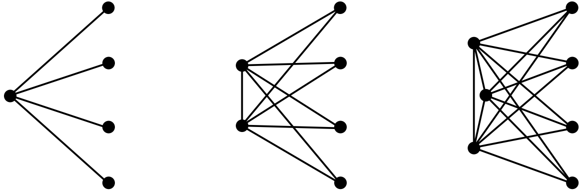


Erdős, Hajnal, and Moon

Theorem (Erdős, Hajnal, Moon, 1964) Let $r \geq 3$. If G is K_r-saturated on n vertices, then G has **at least** $\binom{r-2}{2} + (r-2)(n-r+2)$ edges.

Erdős, Hajnal, and Moon

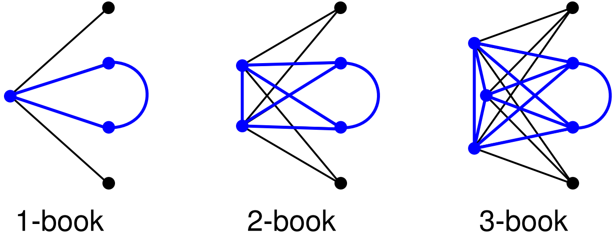
Theorem (Erdős, Hajnal, Moon, 1964) Let $r \geq 3$. If G is K_r-saturated on n vertices, then G has **at least** $\binom{r-2}{2} + (r-2)(n-r+2)$ edges.



1-book   2-book   3-book

Erdős, Hajnal, and Moon

Theorem (Erdős, Hajnal, Moon, 1964) Let $r \geq 3$. If G is K_r-saturated on n vertices, then G has **at least** $\binom{r-2}{2} + (r-2)(n-r+2)$ edges.



Exactly one copy of K_r!

Uniquely H-Saturated Graphs

The Turán graph has **many** copies of K_r when an edge is added.

The books have **exactly one** copy of K_r when an edge is added.

Uniquely H-Saturated Graphs

The Turán graph has **many** copies of K_r when an edge is added.

The books have **exactly one** copy of K_r when an edge is added.

Definition A graph G is **uniquely H-saturated** if G does not contain H as a subgraph and for every edge $e \in \overline{G}$ admits **exactly one** copy of H in $G + e$.

We consider the case where $H = K_r$ (an **r-clique**).

Uniquely K_3-Saturated Graphs

Lemma (Cooper, Lenz, LeSaulnier, Wenger, West, 2011)

The uniquely K_3-saturated graphs are either **stars** or **Moore graphs** of diameter 2 and girth 5.

Uniquely K_3-Saturated Graphs

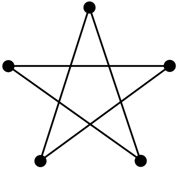
Lemma (Cooper, Lenz, LeSaulnier, Wenger, West, 2011)
The uniquely K_3-saturated graphs are either **stars** or **Moore graphs** of diameter 2 and girth 5.

Theorem (Hoffman, Singleton, 1964) There are a **finite number** of Moore graphs of diameter 2 and girth 5.

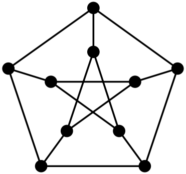
Uniquely K_3-Saturated Graphs

Lemma (Cooper, Lenz, LeSaulnier, Wenger, West, 2011)
The uniquely K_3-saturated graphs are either **stars** or **Moore graphs** of diameter 2 and girth 5.

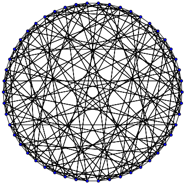
Theorem (Hoffman, Singleton, 1964) There are a **finite number** of Moore graphs of diameter 2 and girth 5.



C_5



Petersen



Hoffman–Singleton

?

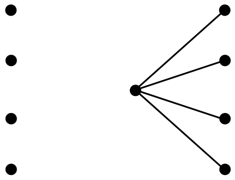
57-Regular
Order 3250

Dominating Vertices

Adding a dominating vertex to a uniquely K_r-saturated graph creates a uniquely K_{r+1}-saturated graph.

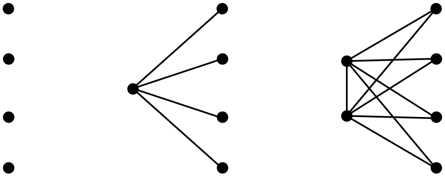
Dominating Vertices

Adding a dominating vertex to a uniquely K_r-saturated graph creates a uniquely K_{r+1}-saturated graph.



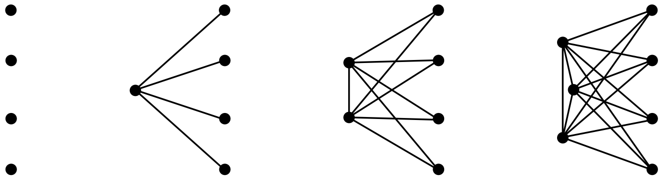
Dominating Vertices

Adding a dominating vertex to a uniquely K_r-saturated graph creates a uniquely K_{r+1}-saturated graph.



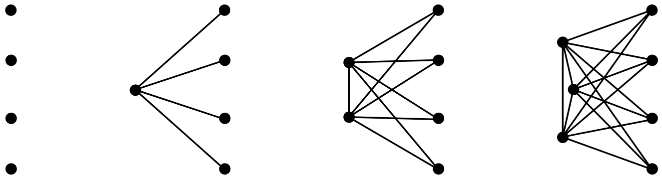
Dominating Vertices

Adding a dominating vertex to a uniquely K_r-saturated graph creates a uniquely K_{r+1}-saturated graph.



Dominating Vertices

Adding a dominating vertex to a uniquely K_r-saturated graph creates a uniquely K_{r+1}-saturated graph.



Call uniquely K_r-saturated graphs without a dominating vertex

***r*-primitive**.

r-Primitive Graphs

A uniquely K_r-saturated graph with no dominating vertex is ***r*-primitive**.

r-Primitive Graphs

A uniquely K_r-saturated graph with no dominating vertex is **r-primitive**.

2**-primitive** graphs are **empty graphs**.

r-Primitive Graphs

A uniquely K_r-saturated graph with no dominating vertex is ***r*-primitive**.

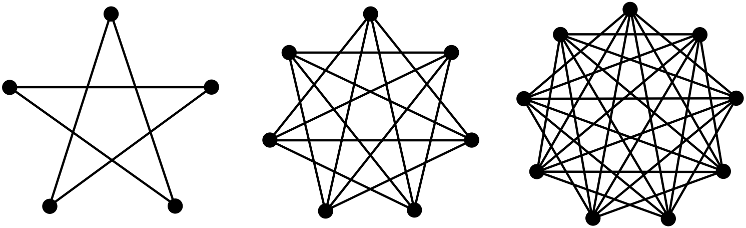
2**-primitive** graphs are **empty graphs**.

3**-primitive** graphs are **Moore graphs** of diameter 2 and girth 5.

r-Primitive Graphs

A uniquely K_r-saturated graph with no dominating vertex is **r-primitive**.

For $r \geq 1$, $\overline{C_{2r-1}}$ is r-primitive.



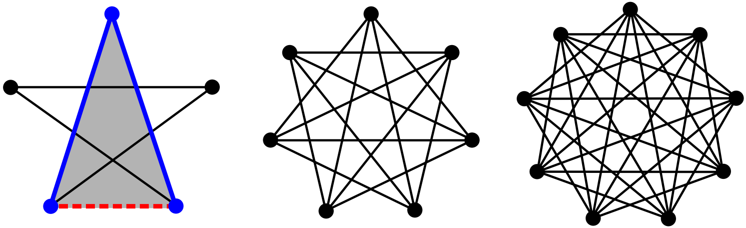
$\overline{C_5}$ $\overline{C_7}$ $\overline{C_9}$

(Collins, Cooper, Kay, Wenger, 2010)

r-Primitive Graphs

A uniquely K_r-saturated graph with no dominating vertex is ***r*-primitive**.

For $r \geq 1$, $\overline{C_{2r-1}}$ is r-primitive.



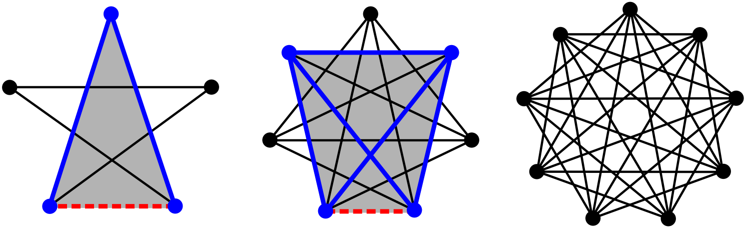
$\overline{C_5}$ $\overline{C_7}$ $\overline{C_9}$

(Collins, Cooper, Kay, Wenger, 2010)

r-Primitive Graphs

A uniquely K_r-saturated graph with no dominating vertex is **r-primitive**.

For $r \geq 1$, $\overline{C_{2r-1}}$ is r-primitive.



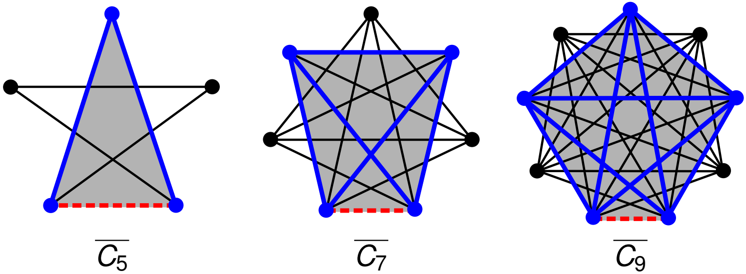
$\overline{C_5}$ $\overline{C_7}$ $\overline{C_9}$

(Collins, Cooper, Kay, Wenger, 2010)

r-Primitive Graphs

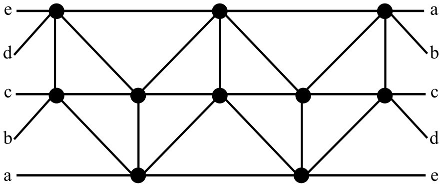
A uniquely K_r-saturated graph with no dominating vertex is ***r*-primitive**.

For $r \geq 1$, $\overline{C_{2r-1}}$ is r-primitive.

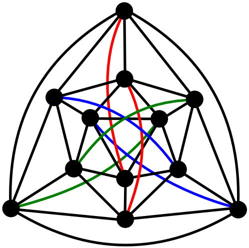


(Collins, Cooper, Kay, Wenger, 2010)

Uniquely K_4-Saturated Graphs



10 vertices



12 vertices

Previously known 4-primitive graphs (Collins, Cooper, Kay, 2010)

Two Questions

Two Questions

1. Fix $r \geq 3$. Are there a **finite number** of r-primitive graphs?

Two Questions

1. Fix $r \geq 3$. Are there a **finite number** of r-primitive graphs?

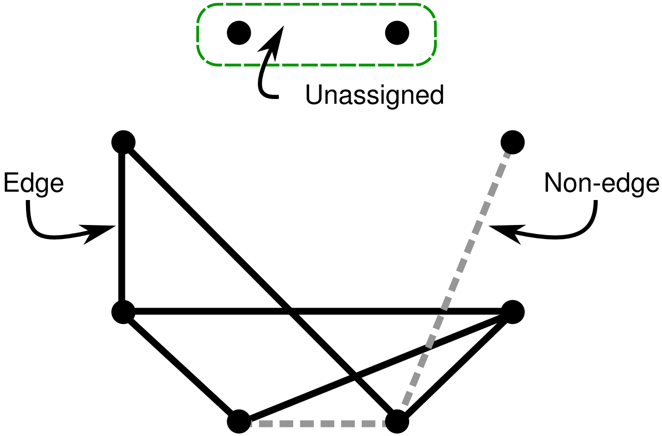
2. Is every r-primitive graph **regular**?

Edges and Non-Edges

Non-edges are crucial to the structure of r-primitive graphs.

Edges and Non-Edges

Non-edges are crucial to the structure of r-primitive graphs.

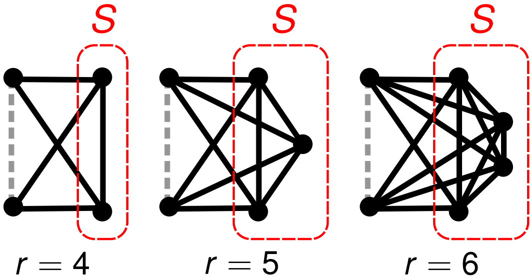


Trigraph

K_r-Completions

For every non-edge we add, we add a K_r-**completion**:

ij a non-edge **if and only if** there exists a set $S \subset [n]$, $|S| = r - 2$, so that ia, ja, and ab are edges for all $a, b \in S$.



Computational Method

Developed a computational method from:

1. **Augmentations:** K_r-Completions.
2. **Isomorphs:** Orbital Branching. Ostrowsky *et al.*
3. **Pruning:** Contains K_r or double-completion.

Exhaustive Search Times

n	$r = 4$	$r = 5$	$r = 6$	$r = 7$	$r = 8$
10	0.10 s	0.37 s	0.13 s	0.01 s	0.01 s
11	0.68 s	5.25 s	1.91 s	0.28 s	0.09 s
12	4.58 s	1.60 m	25.39 s	1.97 s	1.12 s
13	34.66 s	34.54 m	6.53 m	59.94 s	20.03 s
14	4.93 m	10.39 h	5.13 h	20.66 m	2.71 m
15	40.59 m	23.49 d	10.08 d	12.28 h	1.22 h
16	6.34 h	1.58 y	1.74 y	34.53 d	1.88 d
17	3.44 d			8.76 y	115.69 d
18	53.01 d				
19	2.01 y				
20	45.11 y				

Total CPU times using Open Science Grid.

⟵ clique size ⟶

⟵ vertices ⟶

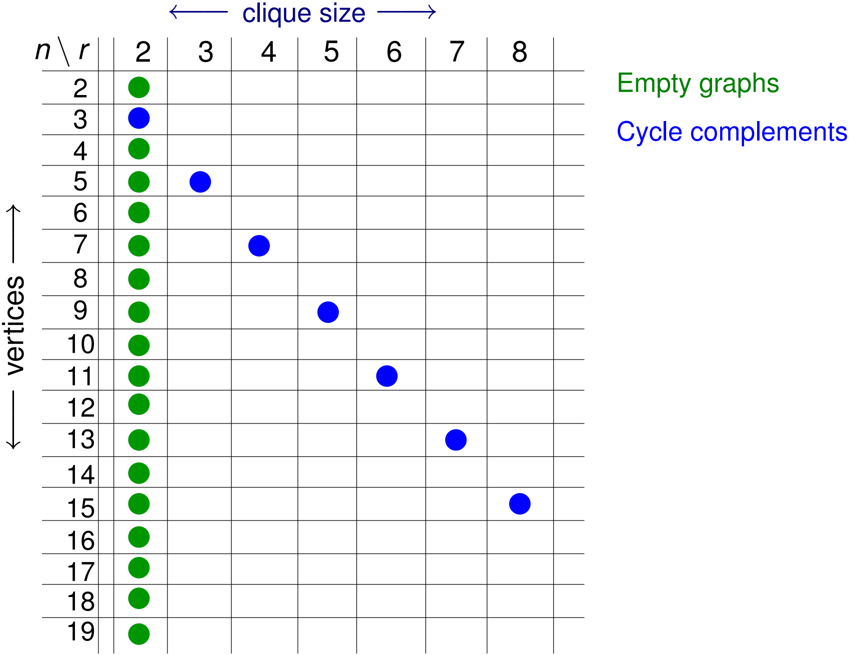
$n \setminus r$		2	3	4	5	6	7	8	
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									

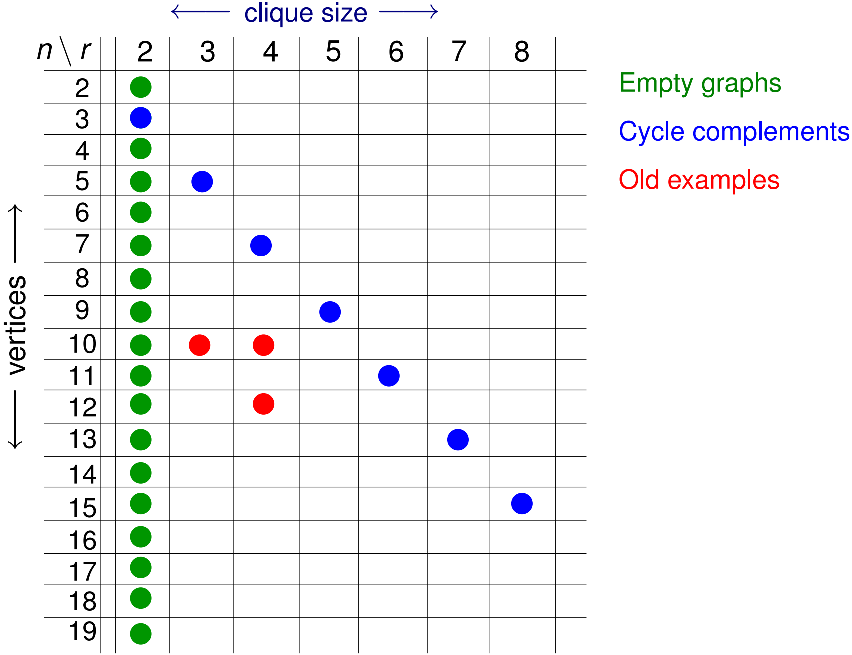
← clique size →

Empty graphs

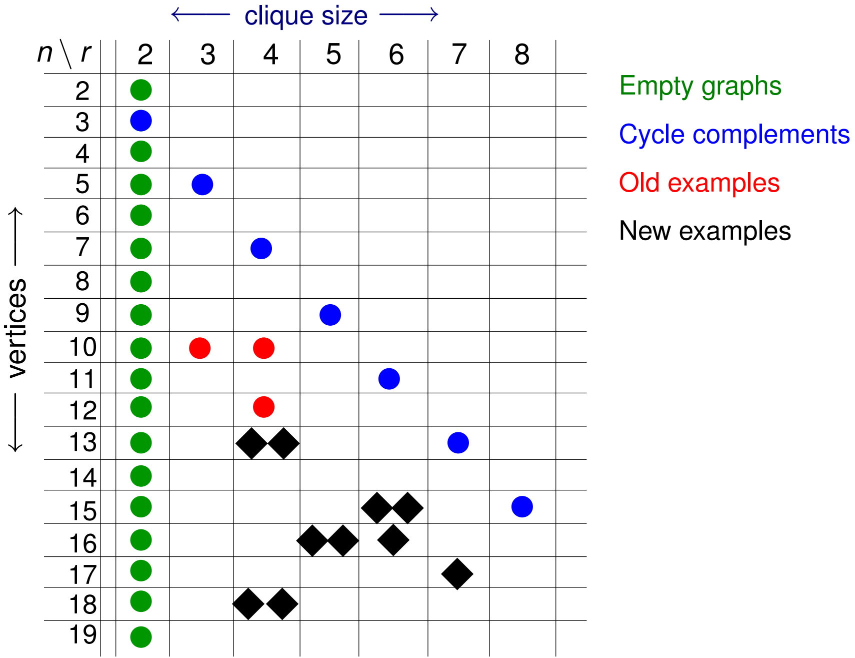
vertices

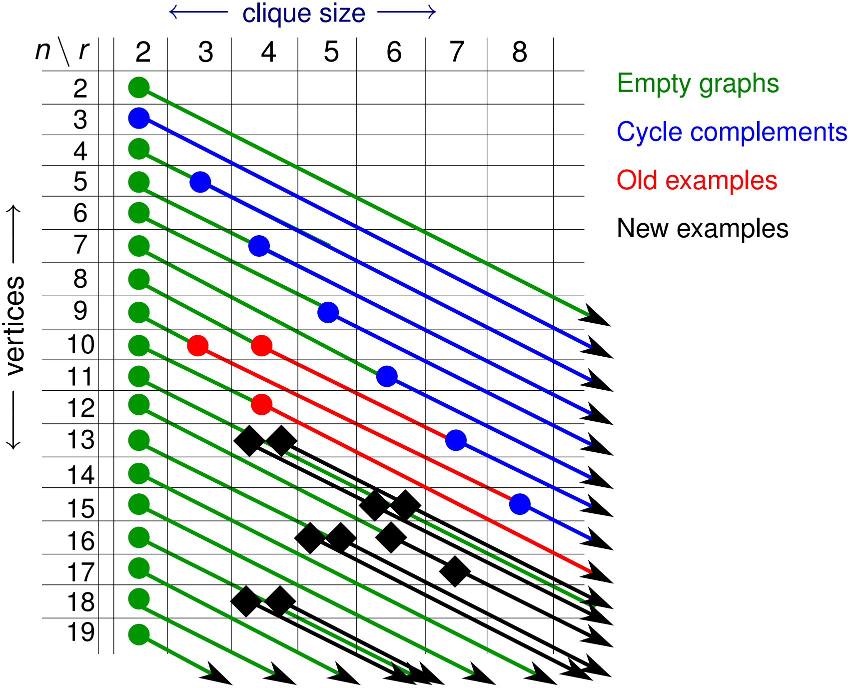
$n \setminus r$	2	3	4	5	6	7	8
2	●						
3	●						
4	●						
5	●						
6	●						
7	●						
8	●						
9	●						
10	●						
11	●						
12	●						
13	●						
14	●						
15	●						
16	●						
17	●						
18	●						
19	●						


clique size
n \ r
2
3
4
5
6
7
8
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
vertices
Empty graphs
Cycle complements



clique size
n \ r
2
3
4
5
6
7
8
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
vertices
Empty graphs
Cycle complements
Old examples

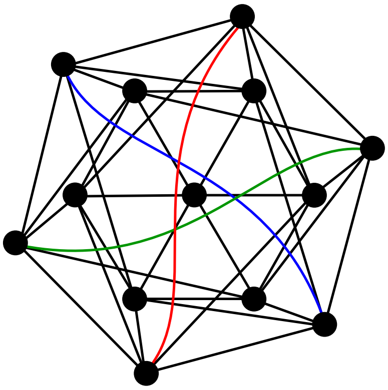

clique size
n \ r
2
3
4
5
6
7
8
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
vertices
Empty graphs
Cycle complements
Old examples
New examples



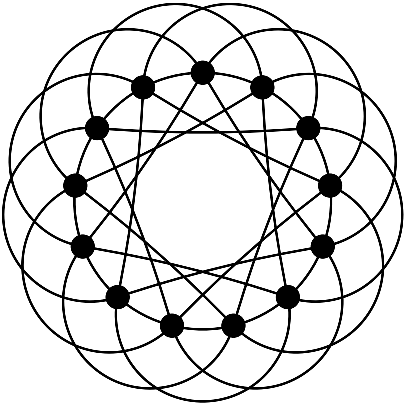
clique size
n \ r
2
3
4
5
6
7
8
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
vertices
Empty graphs
Cycle complements
Old examples
New examples

4-Primitive Graphs

$n = 13$



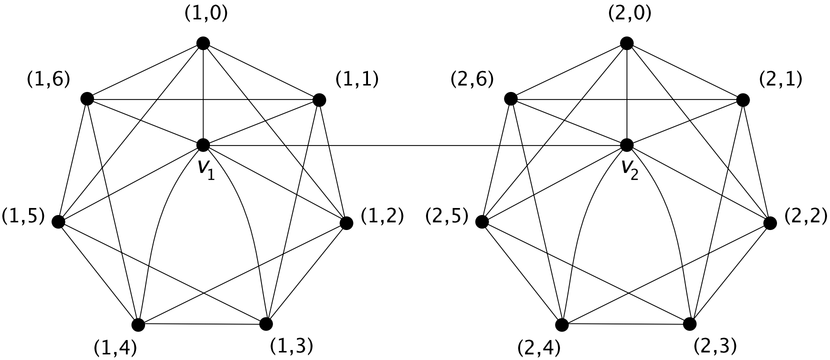
$G_{13}^{(A)}$



Paley(13)

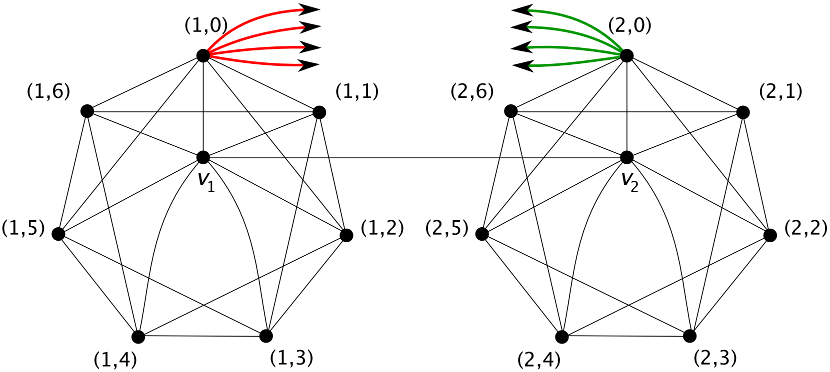
5-Primitive Graph

$n = 16 : G_{16}^{(A)}$



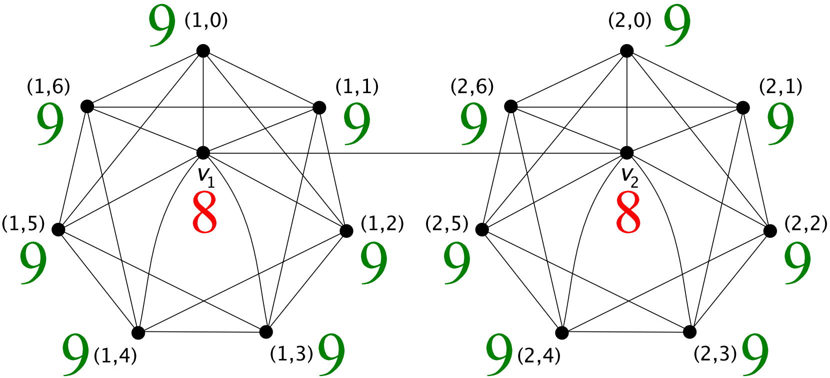
5-Primitive Graph

$n = 16 : G_{16}^{(A)}$



5-Primitive Graph

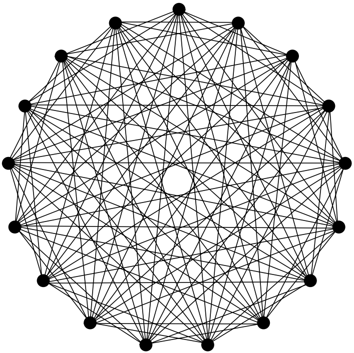
$n = 16 : G_{16}^{(A)}$



Not all *r*-primitive graphs are regular!

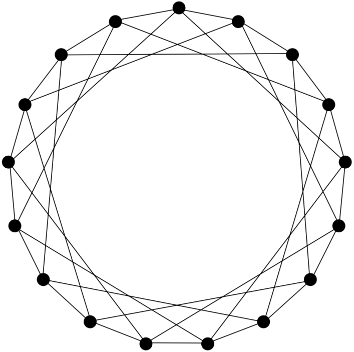
7-Primitive Graph

$n = 17 : G_{17}^{(A)}$



7-Primitive Graph

$n = 17 : G_{17}^{(A)}$



Let Γ be a group and $S \subseteq \Gamma$ a set of generators.

The undirected **Cayley graph** $C(\Gamma, S)$ has vertex set Γ and for all $a \in \Gamma$ and $b \in S$, there is an edge between a and ab.

Let Γ be a group and $S \subseteq \Gamma$ a set of generators.

The undirected **Cayley graph** $C(\Gamma, S)$ has vertex set Γ and for all $a \in \Gamma$ and $b \in S$, there is an edge between a and ab.

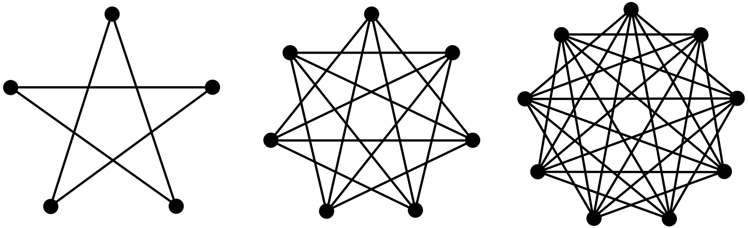
The **Cayley complement** $\overline{C}(\Gamma, S)$ is the complement of $C(\Gamma, S)$.

Let Γ be a group and $S \subseteq \Gamma$ a set of generators.

The undirected **Cayley graph** $C(\Gamma, S)$ has vertex set Γ and for all $a \in \Gamma$ and $b \in S$, there is an edge between a and ab.

The **Cayley complement** $\overline{C}(\Gamma, S)$ is the complement of $C(\Gamma, S)$.

For $r \geq 1$, $\overline{C}(\mathbb{Z}_{2r-1}, \{1\}) \cong \overline{C_{2r-1}}$ is r-primitive.



$\overline{C_5}$ $\overline{C_7}$ $\overline{C_9}$

Two or Three Generators

S	r	n
$\{1,4\}$	7	17
$\{1,6\}$	16	37
$\{1,8\}$	29	65
$\{1,10\}$	46	101
$\{1,12\}$	67	145

$g = 2$

S	r	n
$\{1,5,6\}$	9	31
$\{1,8,9\}$	22	73
$\{1,11,12\}$	41	133
$\{1,14,15\}$	66	211
$\{1,17,18\}$	97	307

$g = 3$

Infinite Families

Conjecture (Hartke, Stolee, 2012) Let $t \geq 1$,

$$n = 4t^2 + 1, \qquad \text{and} \qquad r = 2t^2 - t + 1.$$

The Cayley complement $\overline{C}(\mathbb{Z}_n, \{1, 2t\})$ is r-primitive.

Conjecture (Hartke, Stolee, 2012) Let $t \geq 1$,

$$n = 9t^2 - 3t + 1 \qquad \text{and} \qquad r = 3t^2 - 2t + 1.$$

The Cayley complement $\overline{C}(\mathbb{Z}_n, \{1, 3t-1, 3t\})$ is r-primitive.

Infinite Families

Theorem (Hartke, Stolee, 2012) Let $t \geq 1$,

$$n = 4t^2 + 1, \qquad \text{and} \qquad r = 2t^2 - t + 1.$$

The Cayley complement $\overline{C}(\mathbb{Z}_n, \{1, 2t\})$ is r-primitive.

Proof uses **counting** method.

Conjecture (Hartke, Stolee, 2012) Let $t \geq 1$,

$$n = 9t^2 - 3t + 1 \qquad \text{and} \qquad r = 3t^2 - 2t + 1.$$

The Cayley complement $\overline{C}(\mathbb{Z}_n, \{1, 3t - 1, 3t\})$ is r-primitive.

Infinite Families

Theorem (Hartke, Stolee, 2012) Let $t \geq 1$,

$$n = 4t^2 + 1, \qquad \text{and} \qquad r = 2t^2 - t + 1.$$

The Cayley complement $\overline{C}(\mathbb{Z}_n, \{1, 2t\})$ is r-primitive.

Proof uses **counting** method.

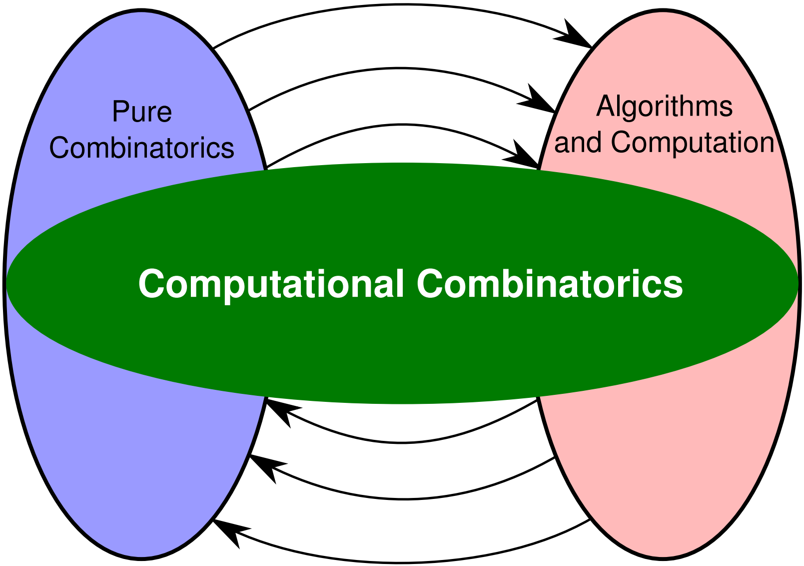
Theorem (Hartke, Stolee, 2012) Let $t \geq 1$,

$$n = 9t^2 - 3t + 1 \qquad \text{and} \qquad r = 3t^2 - 2t + 1.$$

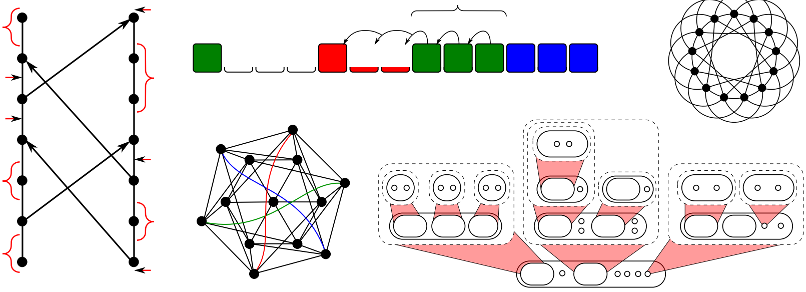
The Cayley complement $\overline{C}(\mathbb{Z}_n, \{1, 3t - 1, 3t\})$ is r-primitive.

Proof uses **discharging** method.

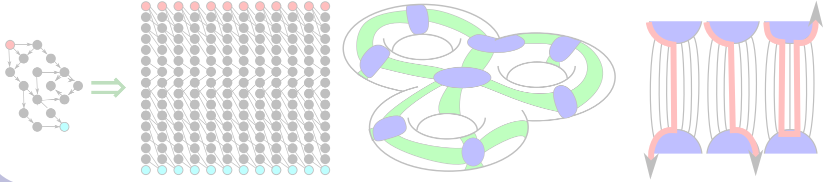
Computational Combinatorics



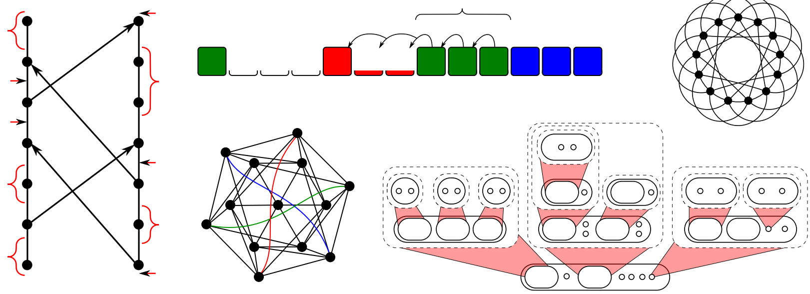
Computational Combinatorics



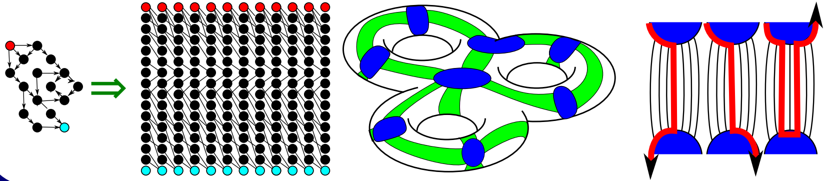
Computational Complexity



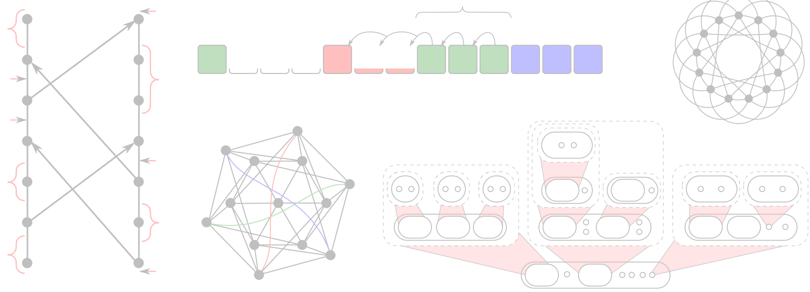
Computational Combinatorics



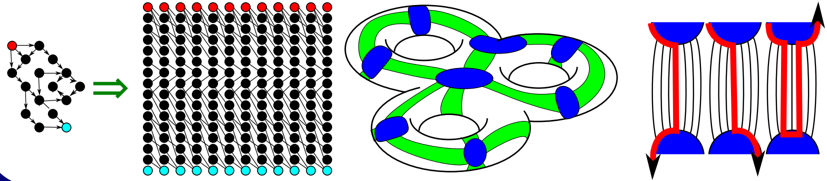
Computational Complexity



Computational Combinatorics



Computational Complexity



Complexity Results in This Thesis

1. ReachFewL $=$ ReachUL.
   (with Garvin, Tewari, Vinodchandran) Chapter 13
2. Reachability in surface-embedded acyclic digraphs.
   (with Vinodchandran) Chapter 14

Space-Bounded Complexity

A language is in L if there is a **deterministic** Turing machine that decides the language using at most $O(\log(n))$ work cells.

Space-Bounded Complexity

A language is in L if there is a **deterministic** Turing machine that decides the language using at most $O(\log(n))$ work cells.

A language is in NL if there is a **non-deterministic** Turing machine that decides the language using at most $O(\log(n))$ work cells.

$$\mathsf{L} \subseteq \mathsf{NL}$$

Configuration Graphs

If M is an $O(\log(n))$-space non-deterministic Turing machine and $\mathbf{x} \in \{0,1\}^*$, the **configuration graph** $G_{M,\mathbf{x}}$ has

Configuration Graphs

If M is an $O(\log(n))$-space non-deterministic Turing machine and $\mathbf{x} \in \{0,1\}^*$, the **configuration graph** $G_{M,\mathbf{x}}$ has

1. Vertices are **configurations**: assignments of state, work cell contents, and tape head positions.
   (Requires $O(\log n)$ bits to describe.)

Configuration Graphs

If M is an $O(\log(n))$-space non-deterministic Turing machine and $\mathbf{x} \in \{0,1\}^*$, the **configuration graph** $G_{M,\mathbf{x}}$ has

1. Vertices are **configurations**: assignments of state, work cell contents, and tape head positions.
   (Requires $O(\log n)$ bits to describe.)
2. An edge $C \to C'$ exists if there is a transition function of M whose operation on C results in C'.

Configuration Graphs

If M is an $O(\log(n))$-space non-deterministic Turing machine and $\mathbf{x} \in \{0,1\}^*$, the **configuration graph** $G_{M,\mathbf{x}}$ has

1. Vertices are **configurations**: assignments of state, work cell contents, and tape head positions.
   (Requires $O(\log n)$ bits to describe.)
2. An edge $C \to C'$ exists if there is a transition function of M whose operation on C results in C'.

M accepts $\mathbf{x}$ if and only if there is a path from C_{init} to C_{accept} in $G_{M,\mathbf{x}}$.

Configuration Graphs

If M is an $O(\log(n))$-space non-deterministic Turing machine and $\mathbf{x} \in \{0,1\}^*$, the **configuration graph** $G_{M,\mathbf{x}}$ has

1. Vertices are **configurations**: assignments of state, work cell contents, and tape head positions.
   (Requires $O(\log n)$ bits to describe.)
2. An edge $C \to C'$ exists if there is a transition function of M whose operation on C results in C'.

M accepts $\mathbf{x}$ if and only if there is a path from C_{init} to C_{accept} in $G_{M,\mathbf{x}}$.

$G_{M,\mathbf{x}}$ has poly-size and can be written using log-space.

Meta-Theory of Space-Bounded Complexity

Every space-bounded complexity problem can be reduced to some form of the **reachability problem** in digraphs.

Meta-Theory of Space-Bounded Complexity

Every space-bounded complexity problem can be reduced to some form of the **reachability problem** in digraphs.

$$\text{Reach} = \{\langle G, s, t\rangle : G \text{ is a directed graph with a path from } s \text{ to } t\}$$

$$\mathsf{L} \subseteq \mathsf{NL} \subseteq \mathsf{P}$$

Complexity Results in This Thesis

1. ReachFewL $=$ ReachUL.
   (with Garvin, Tewari, Vinodchandran) Chapter 13

2. Reachability in surface-embedded acyclic digraphs.
   (with Vinodchandran) Chapter 14

Complexity Results in This Thesis

1. ReachFewL = ReachUL.
   (with Garvin, Tewari, Vinodchandran) Chapter 13
2. Reachability in surface-embedded acyclic digraphs.
   (with Vinodchandran) Chapter 14

Log-space Classes and Reachability

L
Deterministic

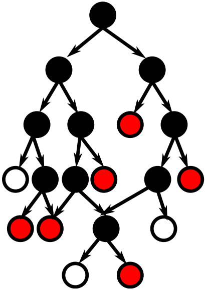
Complete:
Undirected Reach
(Reingold 08)

Log-space Classes and Reachability

L
Deterministic

Complete:
Undirected Reach
(Reingold 08)

NL
Nondeterministic



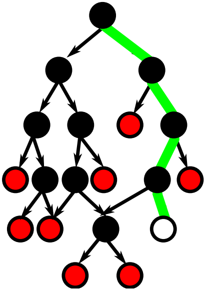
Complete:
Directed Reach

Log-space Classes and Reachability

L
Deterministic

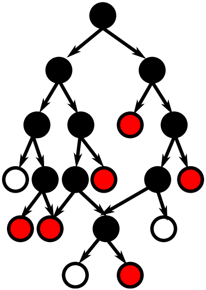
Complete:
Undirected Reach
(Reingold 08)

UL
Unambiguous



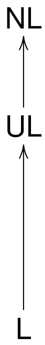
Contains:
Dir. Planar Reach
(Bourke, Tewari, Vinodchandran 09)

NL
Nondeterministic

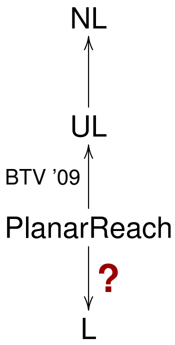


Complete:
Directed Reach

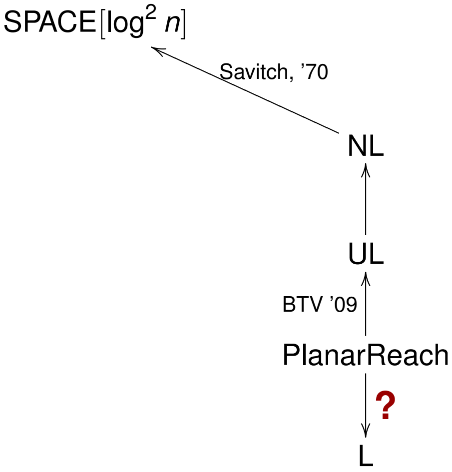
Other Perspectives



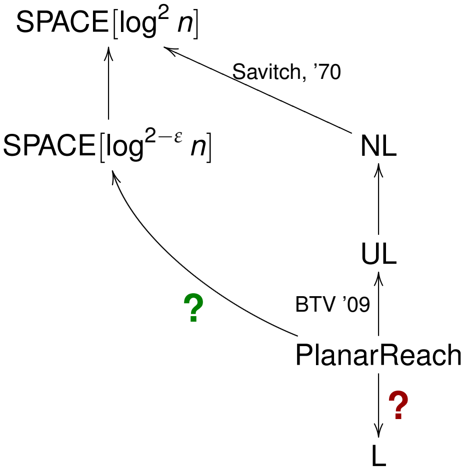
Other Perspectives



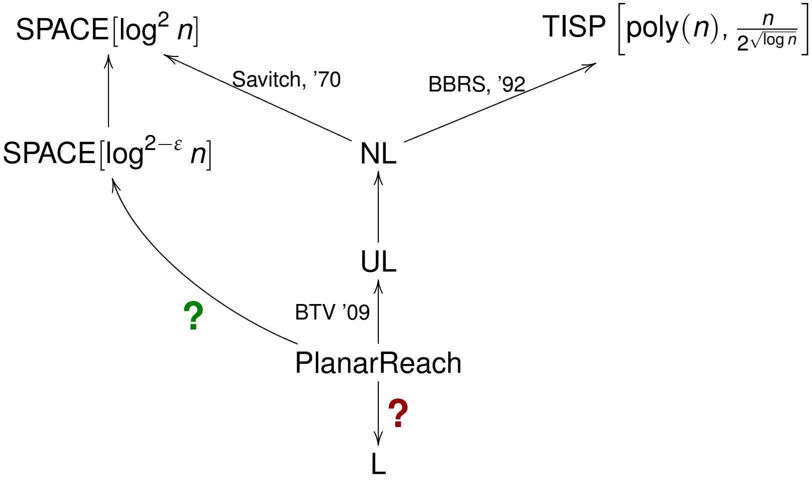
Other Perspectives



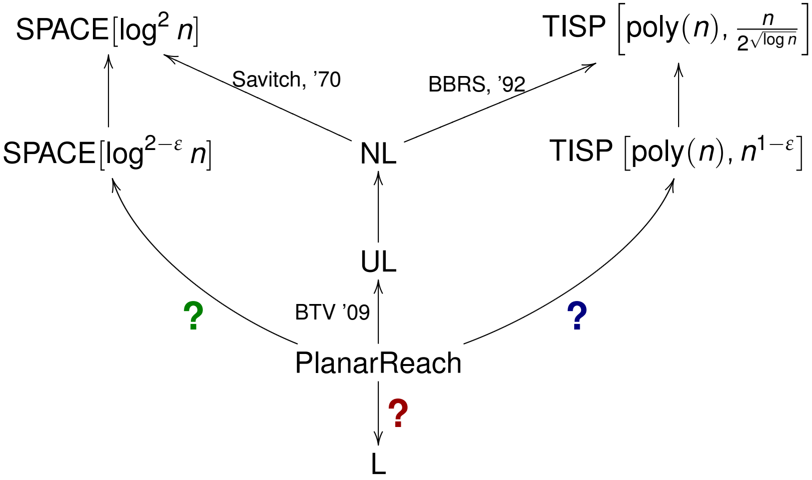
Other Perspectives



Other Perspectives



Other Perspectives



Planar and Acyclic Restrictions

1. Reach for **acyclic** digraphs is complete for NL.

Planar and Acyclic Restrictions

1. Reach for **acyclic** digraphs is complete for NL.
2. Reach for **planar** digraphs is in UL, but we believe UL $=$ NL.

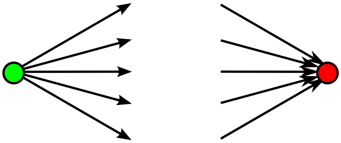
Planar and Acyclic Restrictions

1. Reach for **acyclic** digraphs is complete for NL.
2. Reach for **planar** digraphs is in UL, but we believe UL $=$ NL.
3. What if we combine **acyclic** and **planar**?

Planar and Acyclic Restrictions

1. Reach for **acyclic** digraphs is complete for NL.
2. Reach for **planar** digraphs is in UL, but we believe UL = NL.
3. What if we combine **acyclic** and **planar**?

We also bound number of



sources **sinks**

Planar + Acyclic Reachability in Log-Space

1. Series-parallel graphs
   (Jakoby, Liśkiewicz, Reischuk, Tantau, ’06/’07)

Planar + Acyclic Reachability in Log-Space

1. Series-parallel graphs

   (Jakoby, Liśkiewicz, Reischuk, Tantau, '06/'07)

2. Single-source Single-Sink Planar DAGs

   (Allender, Barrington, Chakraborty, Datta, Roy, '09)

Planar + Acyclic Reachability in Log-Space

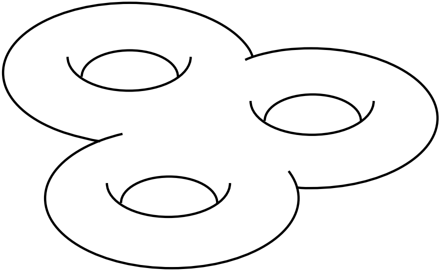
1. Series-parallel graphs
   (Jakoby, Liśkiewicz, Reischuk, Tantau, '06/'07)
2. Single-source Single-Sink Planar DAGs
   (Allender, Barrington, Chakraborty, Datta, Roy, '09)
3. Single-source Multiple-Sink Planar DAGs
   (Allender, Barrington, Chakraborty, Datta, Roy, '09)

Planar + Acyclic Reachability in Log-Space

1. Series-parallel graphs
   (Jakoby, Liśkiewicz, Reischuk, Tantau, '06/'07)
2. Single-source Single-Sink Planar DAGs
   (Allender, Barrington, Chakraborty, Datta, Roy, '09)
3. Single-source Multiple-Sink Planar DAGs
   (Allender, Barrington, Chakraborty, Datta, Roy, '09)
4. Log-source Multiple-Sink Planar DAGs
   (Stolee, Bourke, Vinodchandran, '10)

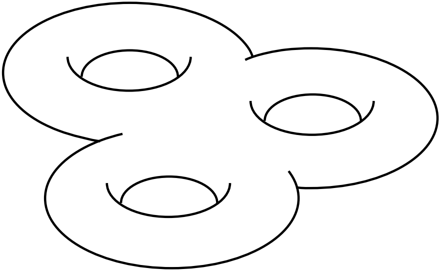
Surface-embedded graphs

We also extend to graphs embedded in *surfaces of low genus*.



Surface-embedded graphs

We also extend to graphs embedded in *surfaces of low genus*.



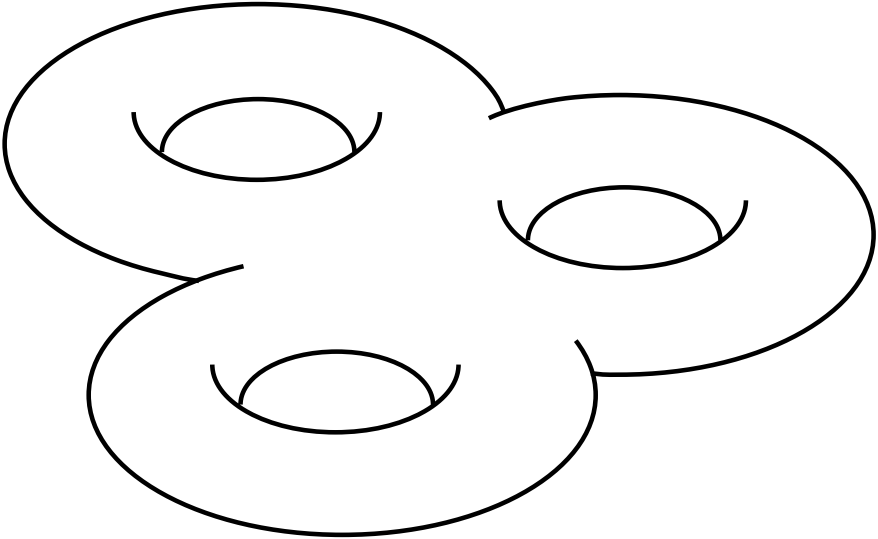
Let $\mathcal{G}(m, g)$ denote the **acyclic** digraphs with m **sources** and embedded in a **genus** g **surface**.

Reduction with Compression

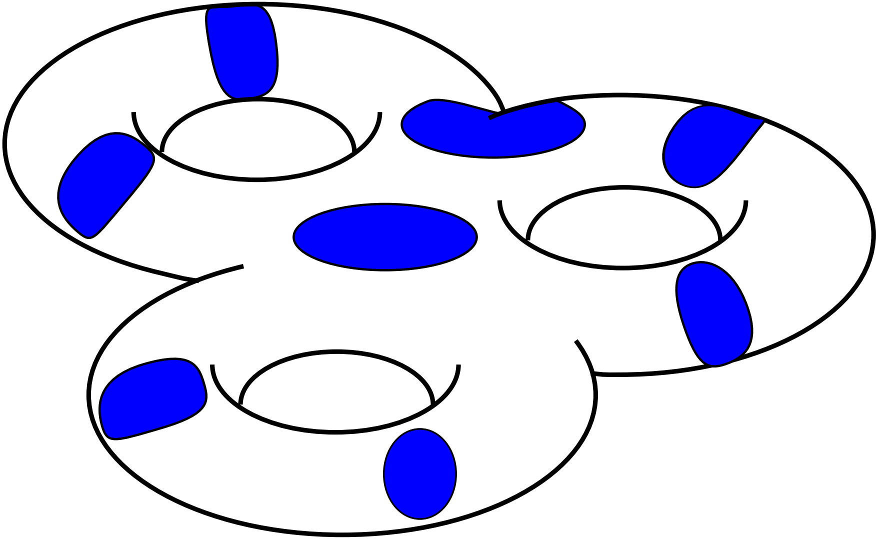
Theorem (Stolee, Vinodchandran, '12) Given a graph $G \in \mathcal{G}(m, g)$ and $s, t \in V(G)$, we can compute in **log-space** a graph G' with vertices s', t' so that

1. There is a path from s to t in G **if and only if** there is a path from s' to t' in G'.
2. G' has $O(m + g)$ vertices.

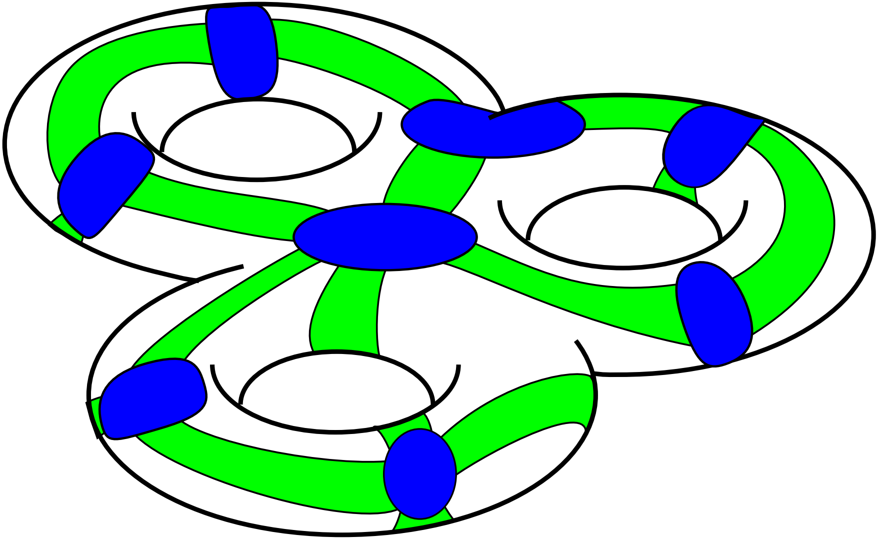
Topological Equivalence



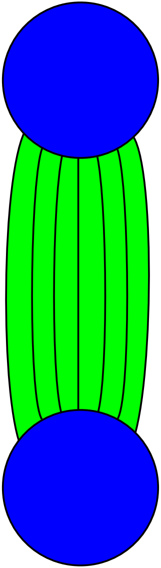
Topological Equivalence



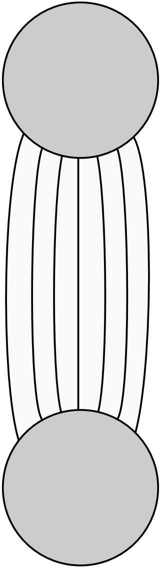
Topological Equivalence



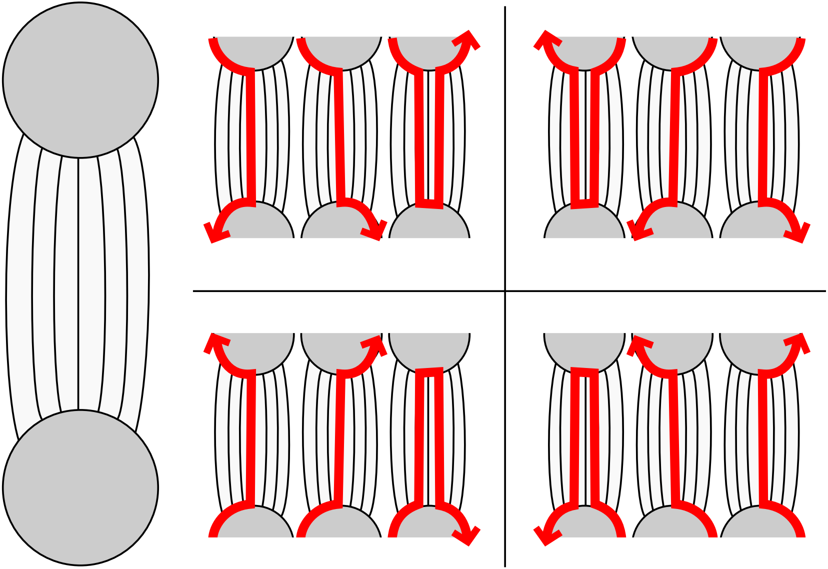
Topological Equivalence



Topological Equivalence



Topological Equivalence



Our Results (Stolee, Vinodchandran, '12)

Theorem (Sub-Savitch) Reachability for graphs of order n in $\mathcal{G}(m, g)$ is in $\mathsf{SPACE}[\log n + \log^2(m+g)]$.

Our Results (Stolee, Vinodchandran, '12)

Theorem (Sub-Savitch) Reachability for graphs of order n in $\mathcal{G}(m, g)$ is in $\mathsf{SPACE}[\log n + \log^2(m + g)]$.

Theorem (Log-Space) If $m = g = 2^{\sqrt{\log n}}$, reach for $\mathcal{G}(m, g)$ is in L.

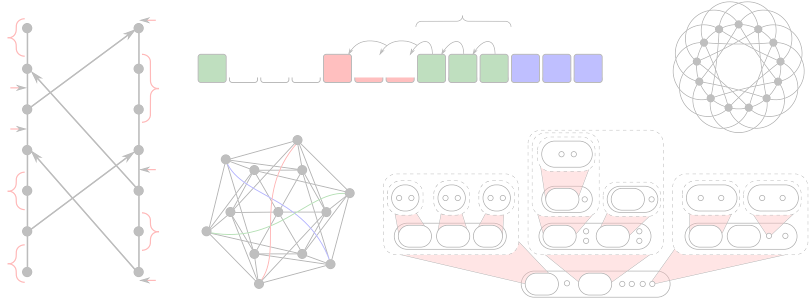
Our Results (Stolee, Vinodchandran, '12)

Theorem (Sub-Savitch) Reachability for graphs of order n in $\mathcal{G}(m, g)$ is in $\mathsf{SPACE}[\log n + \log^2(m+g)]$.

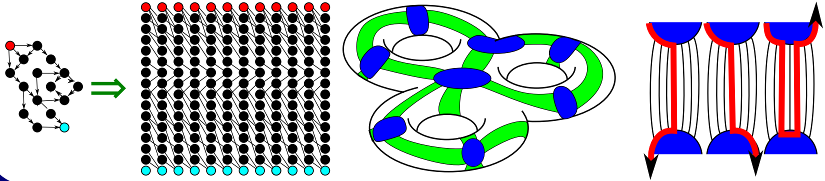
Theorem (Log-Space) If $m = g = 2^{\sqrt{\log n}}$, reach for $\mathcal{G}(m, g)$ is in L.

Theorem (Time-Space) Reachability for graphs of order n in $\mathcal{G}(m, g)$ is in $\mathsf{TISP}[\mathrm{poly}(n), \log n + m + g]$.

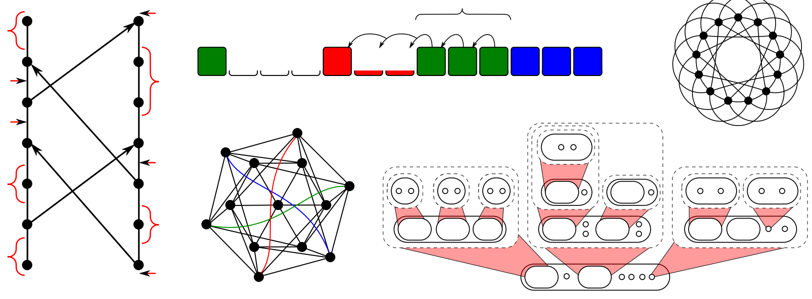
Computational Combinatorics



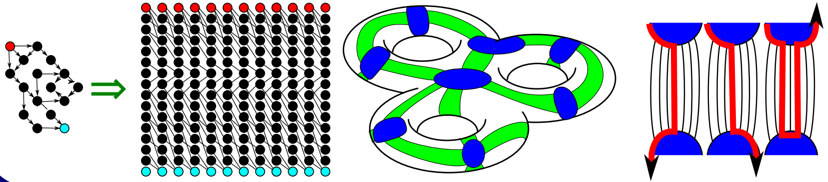
Computational Complexity



Computational Combinatorics



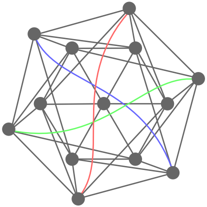
Computational Complexity

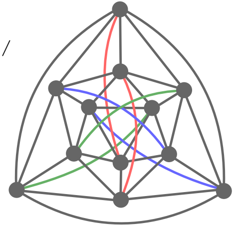


Combinatorics Using Computational Methods

Derrick Stolee

University of Nebraska–Lincoln
s-dstolee1@math.unl.edu
http://www.math.unl.edu/∼s-dstolee1/





March 13, 2012
Dissertation Defense

Supported by NSF grants DMS-0354008, DMS-0914815, and CCF-0916525,

and a University of Nebraska Presidential Fellowship.