Graphs: Isomorphism, Cycles, Randomness and Reconstruction

Honors Thesis by Grady Ward ‘16
Supervised by Professor James Storer

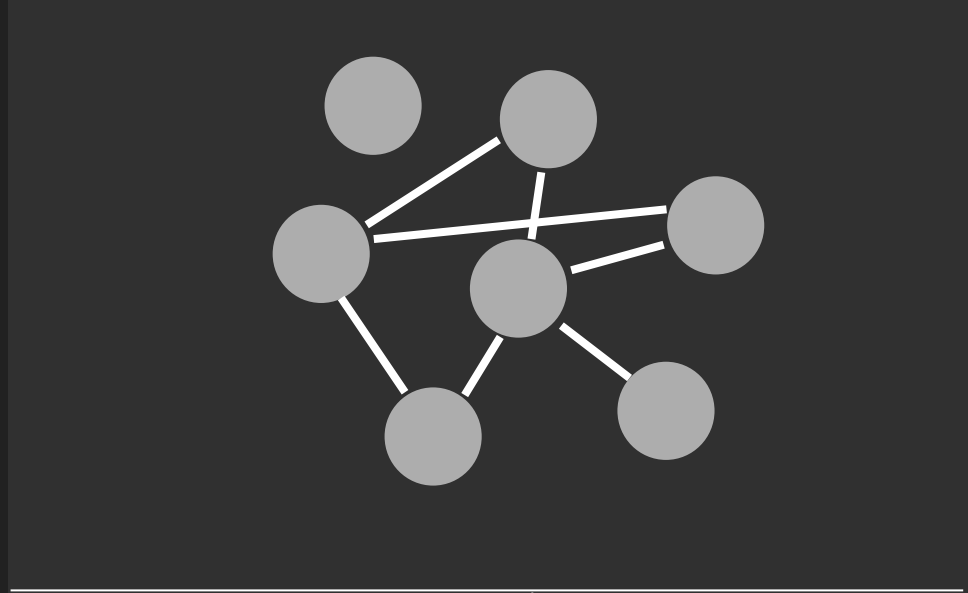
1) Graphs + Groups
2) Random Graph Generators
3) Invariants, Cycles
4) Reconstruction Hypothesis

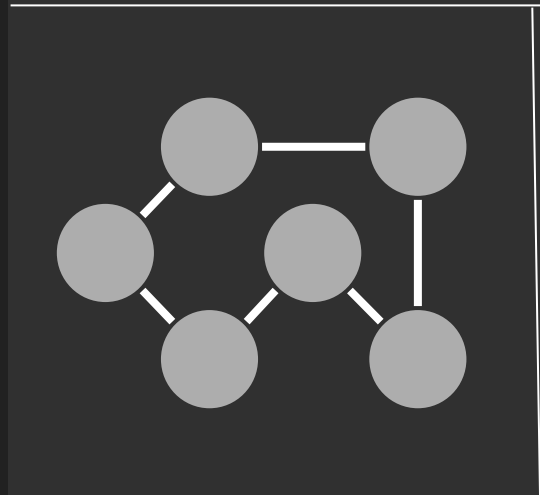
Prerequisites:

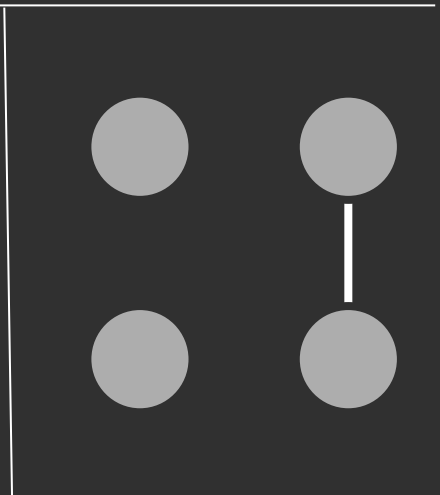
Group / Graph Theory In 10 minutes

Graphs

- Vertices + Edges
- Undirected Edges
- No Multi-edges (0 or 1)
- No Self-Loops
- No Labels
- Not Necessarily Fully Connected
- Algebraic Objects

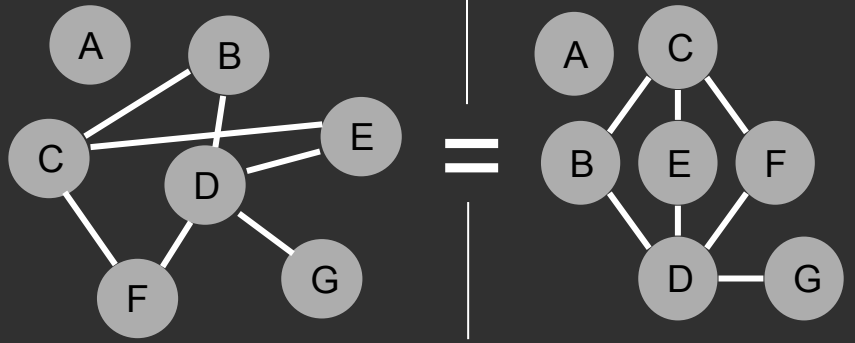






Representation

- Don’t let diagrams fool you!
- Adjacency Matrix:
  - Binary,
  - Zero-diagonaled
  - Requires Labeling
  - Unique to Labeling of Graph



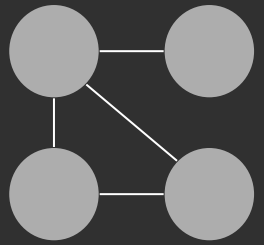
	A	B	C	D	E	F	G
A	0	0	0	0	0	0	0
B	0	0	1	1	0	0	0
C	0	1	0	0	1	1	0
D	0	1	0	0	1	1	1
E	0	0	1	1	0	0	0
F	0	0	1	1	0	0	0
G	0	0	0	1	0	0	0

Graph vs. Graph Instance

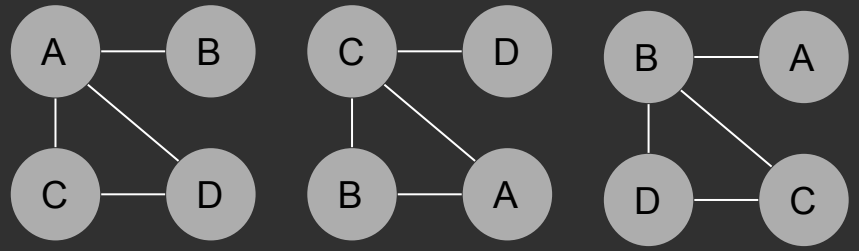
Graph = A structure (corresponding to the algebraic group), regardless of representation

Graph Instance = A **labeled** structure, which is one (of possibly multiple) lablelings of its graph.

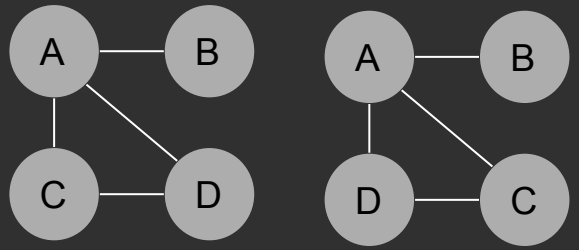
Graph *G*



Instances of Graph *G* (3 / 12 of them)



NOT different instances! why? Same Adj. Mat.



0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0

Isomorphism

Two graph instances are isomorphic if one can be relabeled via a mapping to equal the other.

If two graph instances are isomorphic, they represent the same graph.

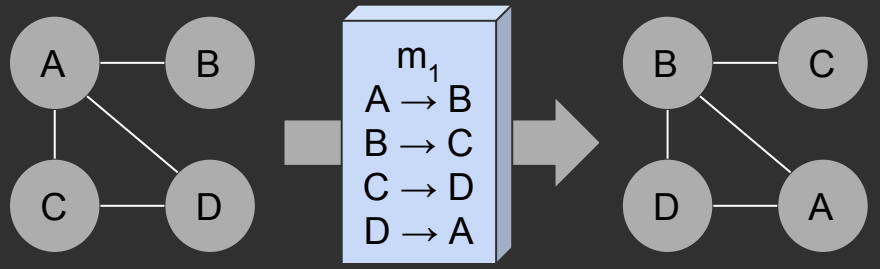
What is a mapping, you ask?

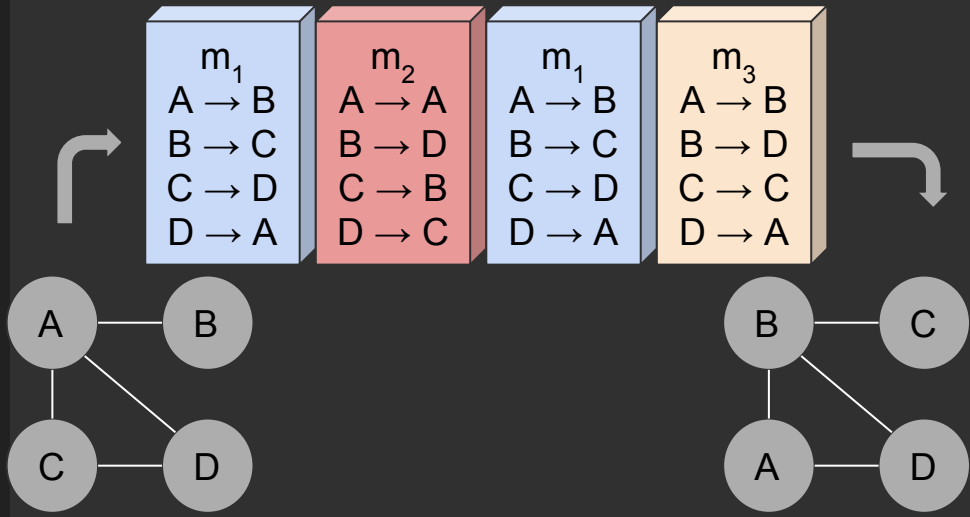
Mappings

One to One permutation

Map full set of graph vertices to full set of graph vertices.

Mappings are chained together by the “followed by” operation



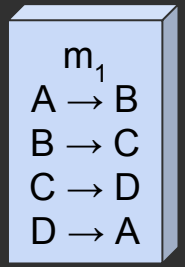


Mappings

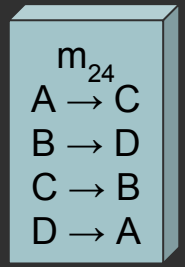
There are N! Different one to one Mappings

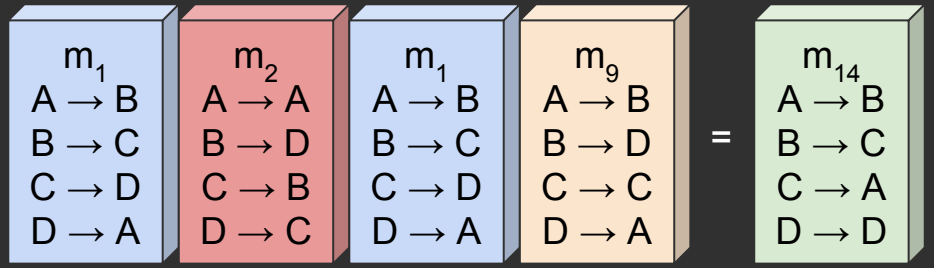
Math People: Mappings are actually members of a group (closed under the followed by operation)

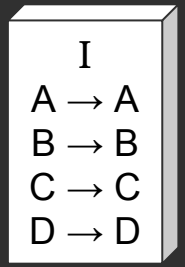
The Identity Element of this group is the mapping that maps each element to itself.



...





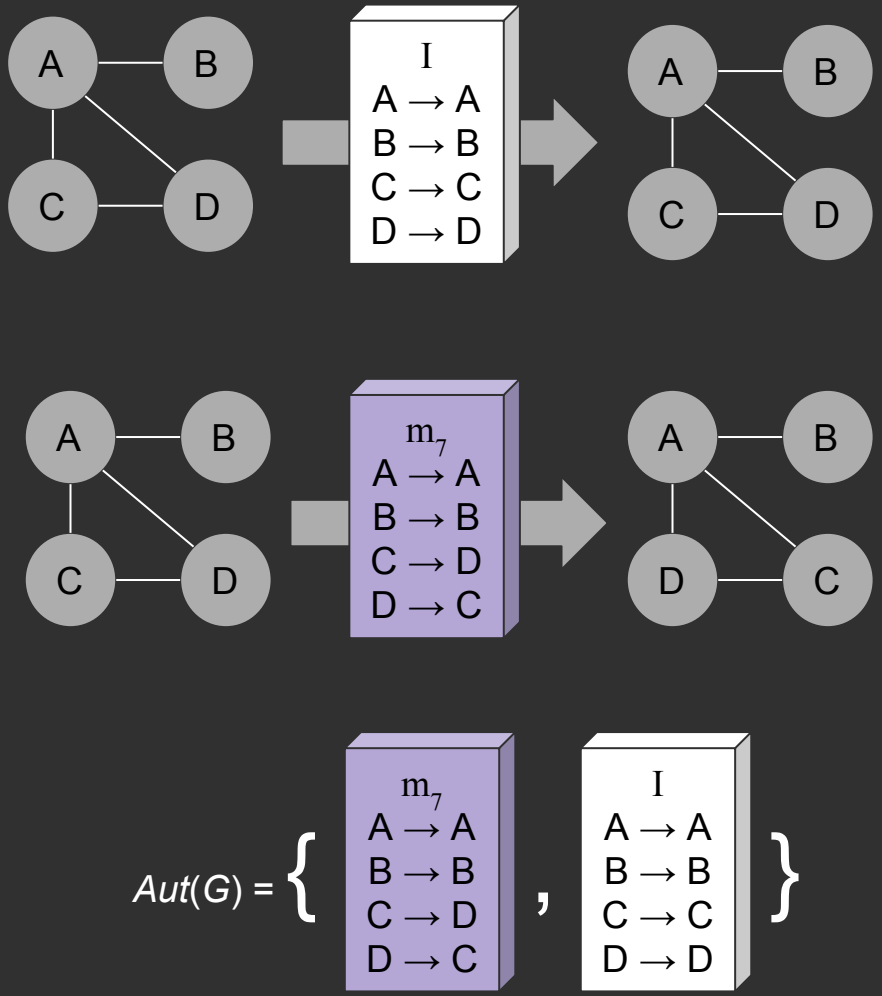


Automorphism

An automorphism is a mapping that preserves adjacency and non-adjacency.

In other words, putting the graph through the mapping does not change the graph.

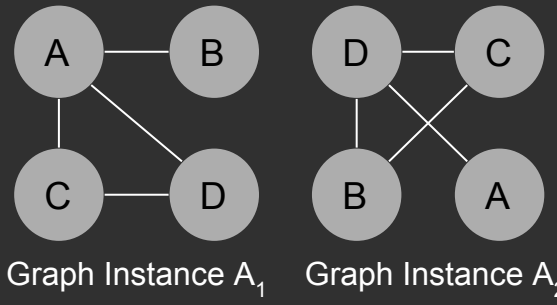
The Automorphism Group is the subgroup of mappings that are all automorphisms. This graph has one of size 2.



Isomorphism (≅)

Two graph instances are isomorphic if one can be relabeled via a mapping to equal the other.

If two graph instances are isomorphic, they represent the same graph.



$A_1 \cong A_2$ because $\exists$ m_1

such that

$m_1(A_2) = A_1$

Number of Graph Instances

A Function of the number of automorphisms of the Graph. (i.e. number of isomorphic graph instances)

Note that the complete graph has only one representation, though most graphs have N!

$M_{reps}(G)$ = Number of Matrices that are Representations of Graph G

$Aut(G)$ = Automorphism Group of G

$N(G)$ = Number of Vertices of G

$$M_{reps}(G) = \frac{N(G)!}{|Aut(G)|}$$

$M_{reps}(G)$ can be 1 to N!

There are a lot of graphs.

There are even more graph instances.

4 Vertices

11 Graphs, 64 Instances

8 Vertices

12,346 Graphs, 268 million instances

12 Vertices

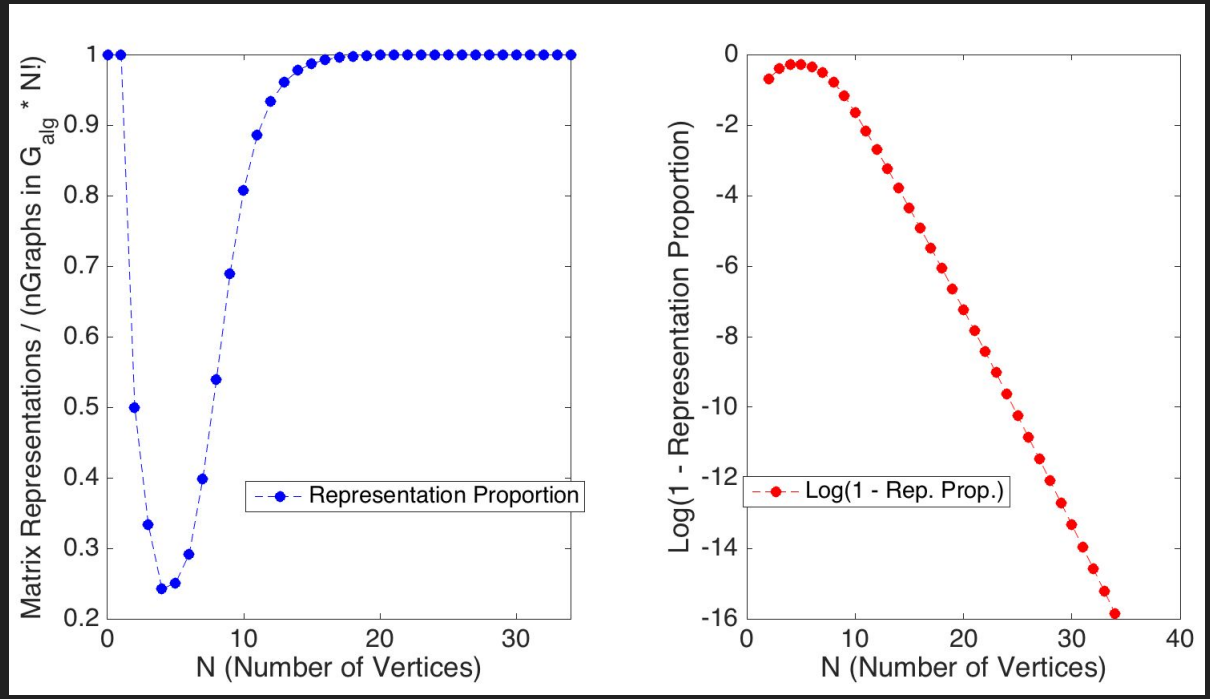
165 Billion Graphs, 7 Quintillion* Instances

*Quintillion = a billion billion

Most Graphs Have N! Instances, 1 Aut.

NMats / (NGraphs * N!) =
If we assume all graphs have N! Matrix Representations, How good is this assumption?
(Strictly Less than 1!)

Logarithm (1 - Blue)



10:00

First Section:

Random Graph Theory

Random Graph Generators

Mechanisms to Evaluate Theory

Ways of simulating real processes (like networks, chemical interactions and linked data)

Asymptotic Running Time of Slow (NPC) Algorithms

Field Established By Erdos + Reyni (1959)

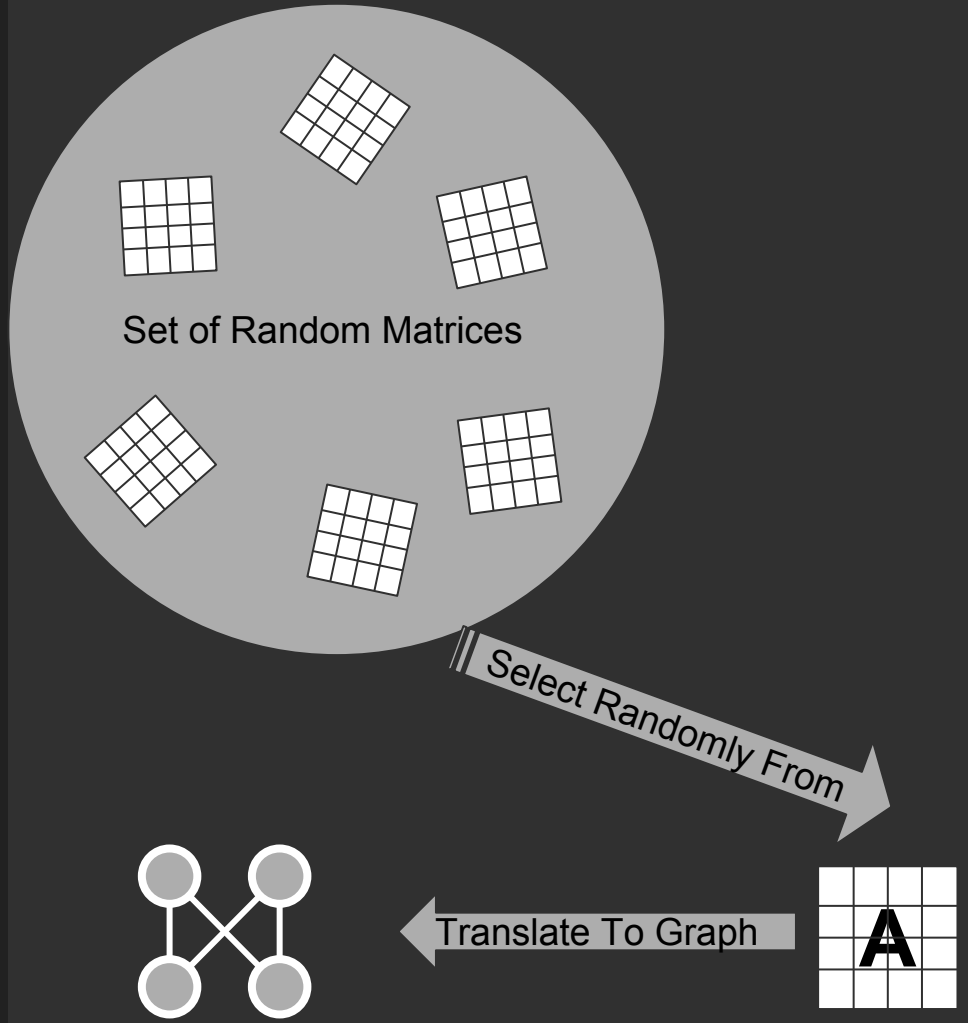
Functionally Equivalent to the Gilbert RGM (1959)

Disjoint Edge Probability =

Flipping a coin for each edge to determine if it exists (allows us to do lots of calculations easily)

Random Graph Generators

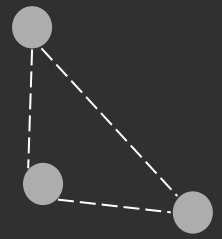
Equivalent to selection from the set of random matrices (when p=.5)



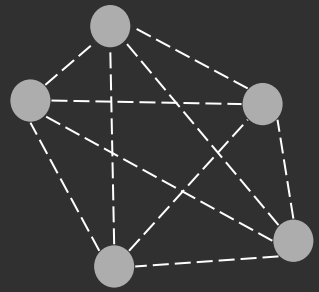
Why is Erdos-Reyni + Gilbert Dominant?

- Disjoint edge probabilities make it easy to prove … pretty much anything
- Easily computable at any size
- Models Real world Scenarios
- (networks, chemicals, etc)
- However, it is applied to claims about random graphs as structures

Equal, independent probability = 0.5



P(Triangle)?



P(Clique)?

Ideal World

We would have a random graph generator which selects all graphs with equal probability.

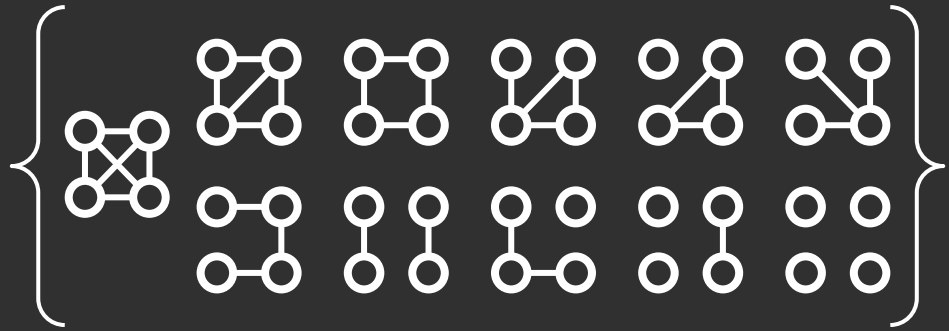
The RGG would randomly select from the set of all graphs, not graph instances.

For small-ish values of N we can do this by enumeration.
We call this an Ideal RGG.

$$\frac{P(G_1)}{P(G_2)} = 1$$

$$\forall \; G_1 \; G_2 \quad \in \quad G_{alg\ 4}$$

$$=$$



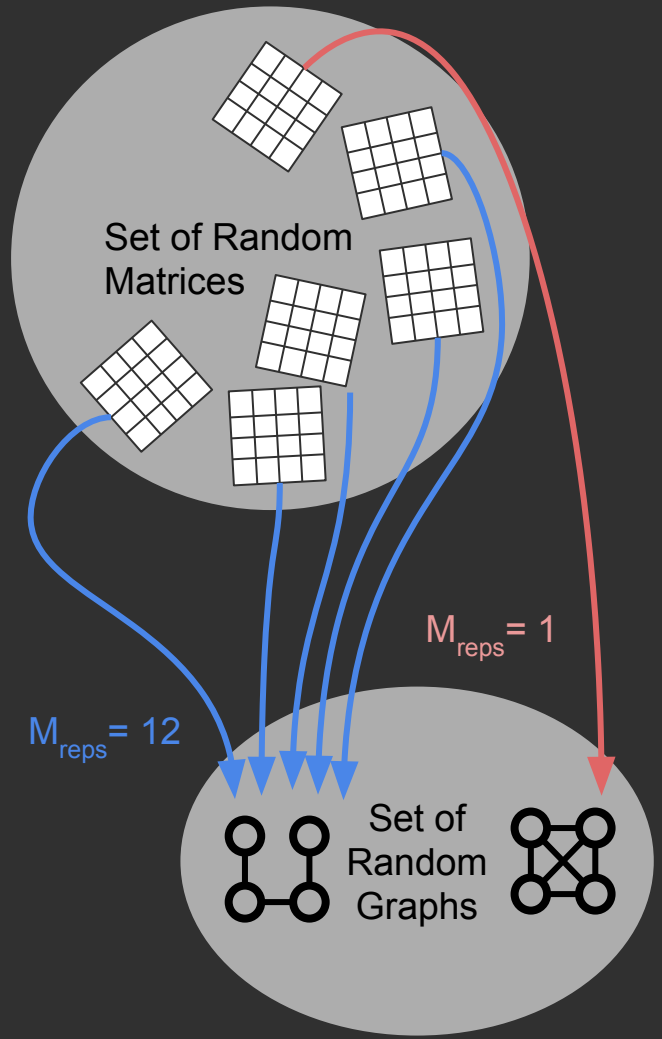
The Set of all Graphs over 4 vertices

Problem

Number of representations is not the same for all graphs

This leads to certain graphs being more likely than others
All graph instances are equally likely

$$\frac{P(\text{graph})}{P(\text{graph})} = 12$$



This difference is important and noticeable

In Properties,
in Theory, and
in Application

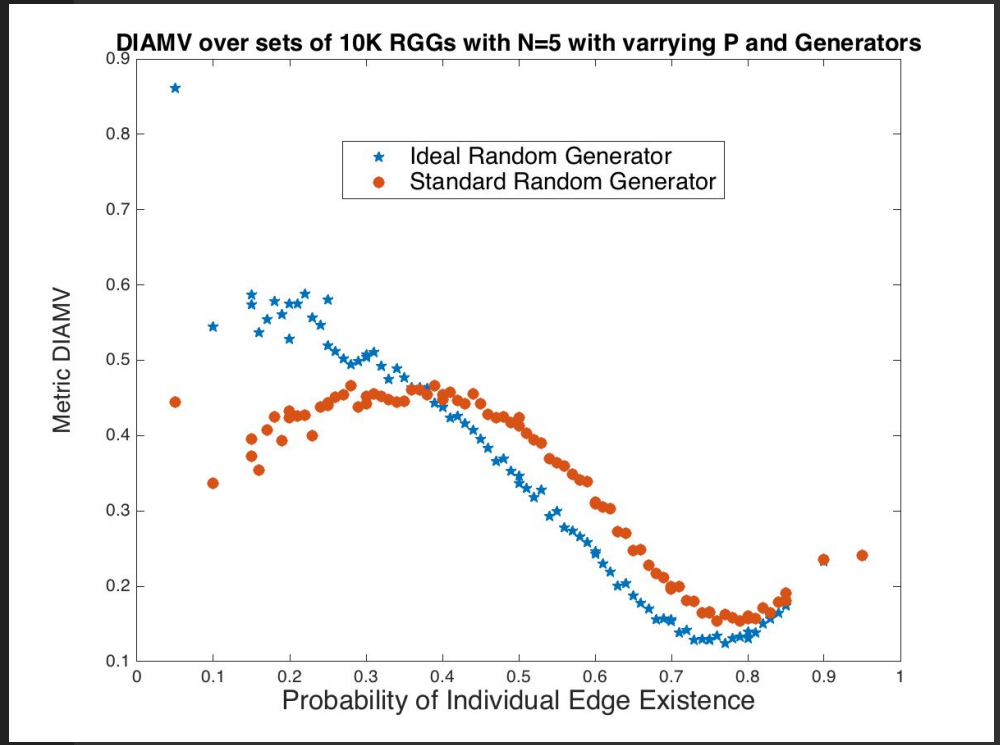
15:00

Appreciable Differences **PROPERTY**

Gilbert Model
Vs.
Idealized Model

Variance in the diameter* of graphs modeled over varied *p*

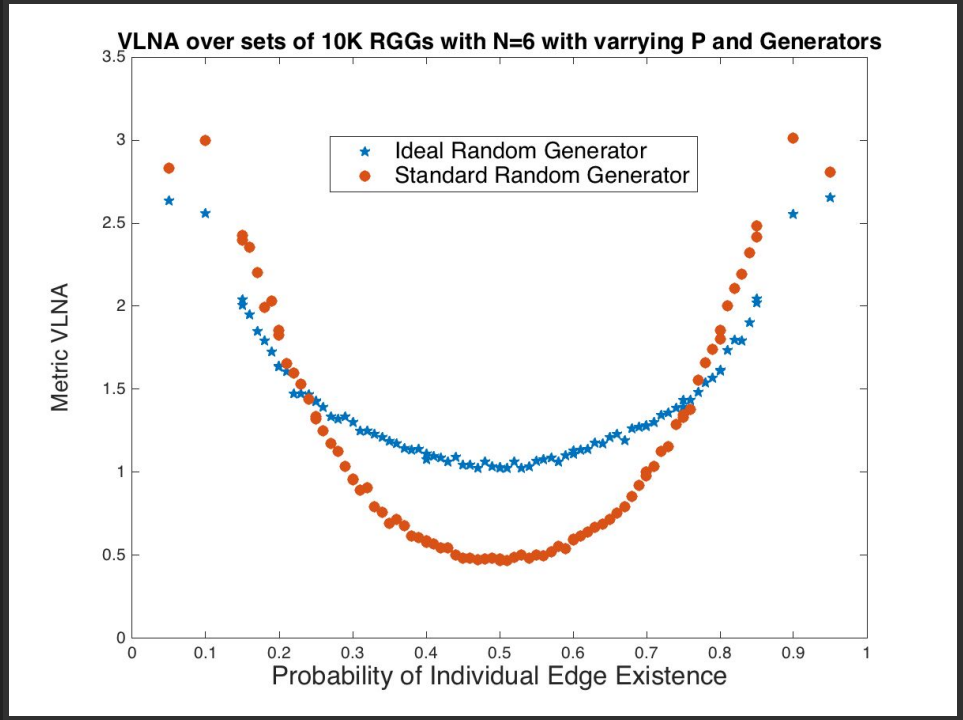
Diameter is the maximum distance between two nodes



Appreciable Differences **PROPERTY**

Gilbert Model
Vs.
Idealized Model

Variance in the logarithm of the number of automorphisms in each graph set over varying values of *p*



Appreciable Differences **THEORY**

Non Uniform Random Selection Yields Misdirected and Incorrect theoretical Results

Example:

A Canonical Labeling Algorithm's Average Running Time*

**Canonical Labeling = Consistent Labeling*

Over Set of Graph Instances

$$\bar{T}_{Gilbert} = \frac{\sum_{g \in G_{inst}} O(T(g))}{|G_{inst}|}$$

$$\bar{T}_{Gilbert} = \frac{\sum_{g \in G_{alg}} O(T(g)) * M_{reps}(g)}{|G_{inst}|}$$

$$\bar{T}_{Gilbert} = \frac{\sum_{g \in G_{alg}} |Aut(g)| * \frac{N!}{|Aut(g)|}}{|G_{inst}|}$$

$$\bar{T}_{Gilbert} = \frac{\sum_{g \in G_{alg}} N!}{2^{.5(N^2-N)}}$$

$$\bar{T}_{Gilbert} = \frac{|G_{alg}| * N!}{2^{.5(N^2-N)}}$$

Over Set of Graph Objects

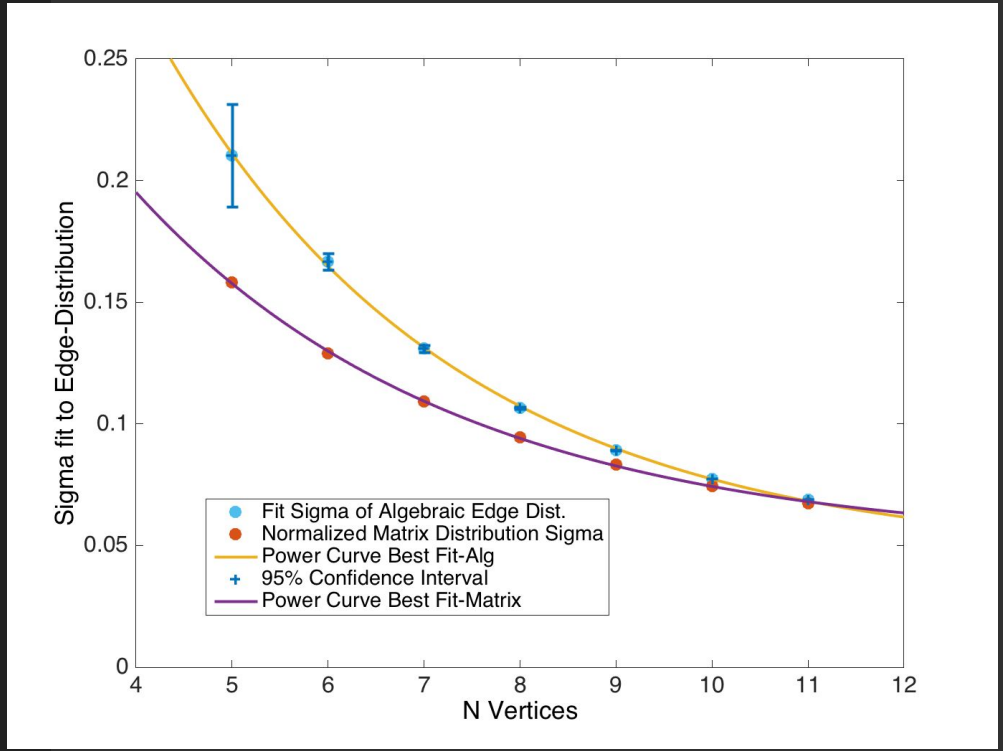
$$\bar{T}_{Ideal} = \frac{\sum_{g \in G_{alg}} O(T(g))}{|G_{alg}|}$$

$$\bar{T}_{Ideal} = \frac{\sum_{g \in G_{alg}} |Aut(g)|}{|G_{alg}|}$$

Appreciable Differences **THEORY**

Gilbert Model
Vs.
Idealized Model

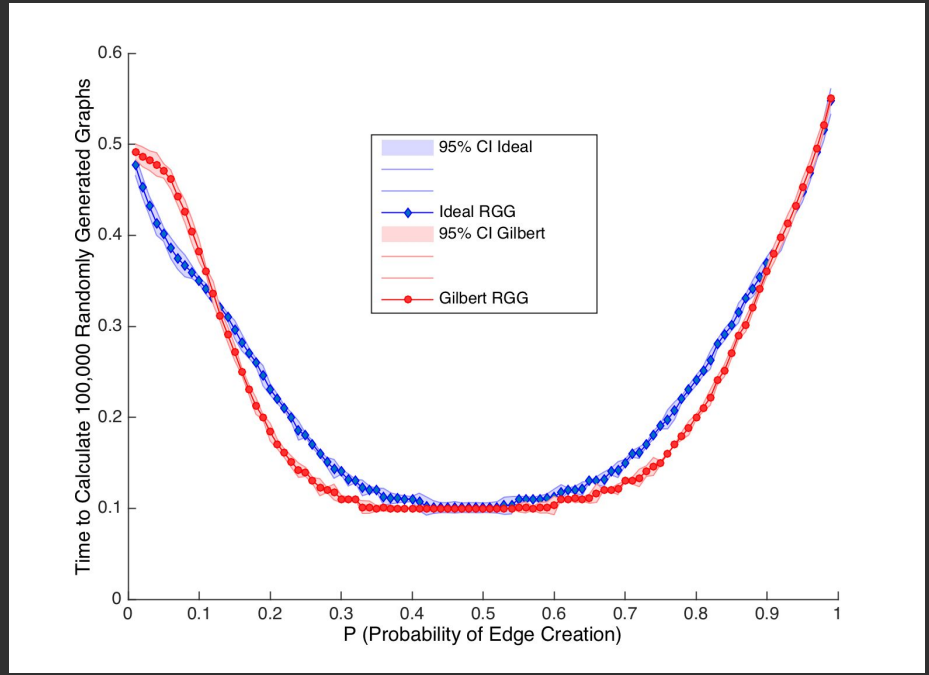
Sigma of Normal Distribution when fit to graph connectivity over each set (set of graphs, set of instances)



Appreciable Differences **APPLICATION**

Gilbert Model
Vs.
Idealized Model

Running time over standard canonical labeling algorithm (NAUTY)



What can we do that is better?

Methodology

(1) Establish Numerical Properties to Measure

(2) Establish Parameters for Evaluation

(3) Establish Baselines for these Properties

(4) Establish Scoring Mechanism...

36 Numerical Properties of Graph Sets, many including fundamental properties graph, others as derivative values (i.e. variance of diameter of graphs in the set)

N = 4, 5, 6, 7, 8, 9, 10
P = .1, .2, .3 … .8, .9
N_{graphs} = 1000
Trials = 30

Get mean and std for all metrics + configurations Over generators {Gilbert, Ideal}

Note that Ideal RGG's are limited to what we can enumerate (i.e. 12 is out of the question)

Example Mechanism: MinusOneA

Input: N (a number of vertices) and p (a probability of edge connection

Step 1: Generate a random graph of size N+1

Step 2: Generate all minus-one subgraphs of G

Step 3: Choose randomly from this set, weighting graphs by their number of automorphisms (1 to N!)

Scoring Mechanism

$$T(A,B) = T(\overline{x_A}, \overline{x_B}, s_A, s_B, n_A, n_B) = \frac{|\overline{x_A} - \overline{x_B}|}{\sqrt{(s_A^2/n_A + s_B^2/n_B}}$$

$$\text{Score for Metric M} = -1 * \min\left[\ln\left(\frac{T_M(I,N)}{T_M(I,G)} + e^{-10}\right), 10\right]$$

$$\text{Score for Generator} = \frac{1}{N_{Metrics}} \times \sum_{m}^{Metrics} Score_m$$

Example Performance: MinusOneA

MinusOneA Score Report: N x Metric, Averaged Across P								
	N							
Metric	4	5	6	7	8	9	10	Mean
ADSM	-2.18	-2.40	-1.71	-2.06	-1.36	-1.18	-1.66	-1.79
DSMV	-2.24	-1.22	-1.73	-0.39	-1.63	-1.24	-1.50	-1.42
ADSMD	-1.81	-0.54	-0.99	-0.42	0.55	0.74	0.44	-0.29
DSMDV	-0.31	-0.10	1.22	1.08	1.07	0.59	0.88	0.63
ADSMN	0.81	-0.83	-0.60	-0.40	-0.19	-0.20	0.12	-0.18
DSMNV	0.88	0.90	0.72	0.67	0.54	0.04	0.01	0.54
ADSMX	0.73	-1.34	-0.83	-0.19	-0.45	-0.11	-0.57	-0.40
DSMXV	0.73	0.89	0.83	0.72	0.32	0.09	0.10	0.53
ADSV	1.94	-1.47	-1.05	-0.14	-0.13	-0.05	-0.06	-0.14
DSVV	0.50	0.11	0.14	0.17	0.16	0.18	0.10	0.19
ADSR	1.27	-0.97	0.30	-0.99	-0.86	-0.27	-0.22	-0.25
PQRA	2.78	2.77	1.68	2.05	1.87	1.05	1.23	1.92
PQRB	0.00	-0.21	-2.06	-0.97	-1.42	-1.05	-0.80	-0.93
PQRC	0.00	-0.21	-2.06	-0.97	-1.42	-0.23	-0.14	-0.72
ODSPL	-2.18	-2.40	-1.71	-2.06	-1.36	-1.18	-1.66	-1.79
ANA	-0.07	-0.45	-0.38	-0.15	0.08	-0.60	-0.09	-0.24
NCP	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NR	0.69	-0.74	0.42	-0.03	0.85	-0.35	0.13	0.14
NE	-2.18	-2.40	-1.71	-2.06	-1.36	-1.18	-1.66	-1.79
PCONN	-0.09	0.88	0.55	0.43	0.53	0.40	1.52	0.60

Example Performance: MinusOneA

MinusOneA Score Report: P x Metric, Averaged Across N										
	P									
Metric	.1	.2	.3	.4	.5	.6	.7	.8	.9	Mean
PQRB	-0.39	-1.91	-1.09	-0.83	-0.78	-0.75	-1.07	-1.13	-0.44	-0.93
PQRC	-0.30	-1.66	-0.89	-0.59	-0.48	-0.54	-0.74	-0.94	-0.35	-0.72
ODSPL	-1.97	-2.15	-1.97	-1.99	-0.11	-1.31	-1.76	-2.82	-2.05	-1.79
ANA	-2.02	-0.74	0.66	0.38	0.41	0.39	0.59	0.28	-2.11	-0.24
NCP	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NR	-0.46	-0.02	0.55	0.47	0.53	0.63	0.47	0.20	-1.12	0.14
NE	-1.97	-2.15	-1.97	-1.99	-0.11	-1.31	-1.76	-2.82	-2.05	-1.79
PCONN	0.06	1.89	1.46	0.80	0.39	0.35	0.21	0.02	0.24	0.60
ANCC	-0.10	1.05	0.98	1.25	0.48	0.40	0.22	0.02	0.25	0.51
NCCV	0.35	0.12	0.06	0.92	0.12	0.75	0.33	-0.01	0.27	0.32
ADIAM	-1.31	0.15	0.21	0.18	0.34	-0.74	0.86	-0.55	-2.43	-0.36
DIAMV	-0.97	0.22	-0.26	-0.59	-0.87	-0.33	-0.16	-0.40	-1.10	-0.49
PTRIL	0.08	-0.25	-0.57	-1.05	2.27	0.63	0.28	0.07	0.08	0.17
ANTRI	0.16	-0.21	-0.18	-0.18	0.24	1.00	-0.20	-1.40	-1.23	-0.22
ANQUAD	-0.74	-0.13	-0.33	0.05	0.49	2.65	0.69	-0.59	-1.52	0.06
ALNA	0.42	0.85	0.61	0.46	0.49	0.44	0.64	0.82	0.24	0.55
MLNA	-4.13	1.57	-0.00	6.03	2.86	4.29	-0.05	0.11	-2.38	0.92
VLNA	0.29	-0.45	-0.64	-0.10	0.09	-0.23	-0.28	-0.30	0.44	-0.13
NUG	1.23	1.56	-0.07	-0.84	0.21	-0.70	-1.75	0.22	-0.22	-0.04
ANR	-0.24	0.06	0.72	-0.65	-1.20	0.14	0.28	0.04	-0.39	-0.14
MNR	-1.03	-0.32	-0.32	0.73	0.57	0.25	-0.05	-0.75	-1.04	-0.22
SQNR	-0.80	0.30	0.53	0.64	0.40	0.21	0.35	0.28	-0.82	0.12
PDL	-0.15	0.04	0.74	-0.62	-1.19	0.14	0.25	-0.15	-0.43	-0.15
Mean	-0.53	-0.14	-0.04	0.09	0.28	0.34	-0.08	-0.37	-0.68	-0.13

Example Performance: MinusOneA

MinusOneA Score Report: N x P, Averaged Across Metrics								
	N							
P	4	5	6	7	8	9	10	Mean
0.1	-0.41	-0.92	-0.63	-0.69	-0.15	-0.81	-0.13	-0.53
0.2	-0.33	0.13	-0.20	-0.24	-0.17	-0.08	-0.10	-0.14
0.3	-0.29	-0.21	-0.19	0.35	-0.06	-0.18	0.27	-0.04
0.4	-0.61	0.19	0.30	-0.25	0.36	0.29	0.35	0.09
0.5	0.37	0.30	0.52	0.14	0.75	-0.21	0.05	0.28
0.6	0.51	0.33	0.14	0.25	0.71	0.14	0.27	0.34
0.7	-0.52	-0.27	0.21	0.25	-0.13	-0.33	0.20	-0.08
0.8	-0.81	-0.23	-0.35	-0.11	-0.18	-0.33	-0.58	-0.37
0.9	-0.21	-1.33	-1.08	-1.01	-0.35	-0.57	-0.22	-0.68
Mean	-0.26	-0.22	-0.14	-0.15	0.08	-0.23	0.01	-0.13

Performance

Score = 0 means comparable deviations to Gilbert RGG

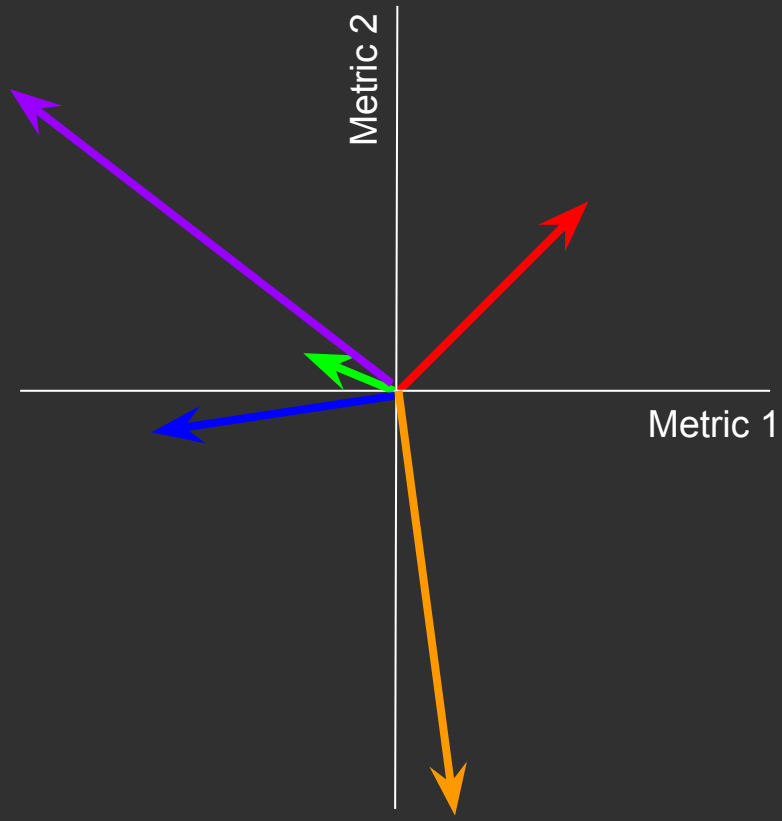
Wrote 13 random graph generators

Average score was bad… significantly worse than RGG (-1.2)

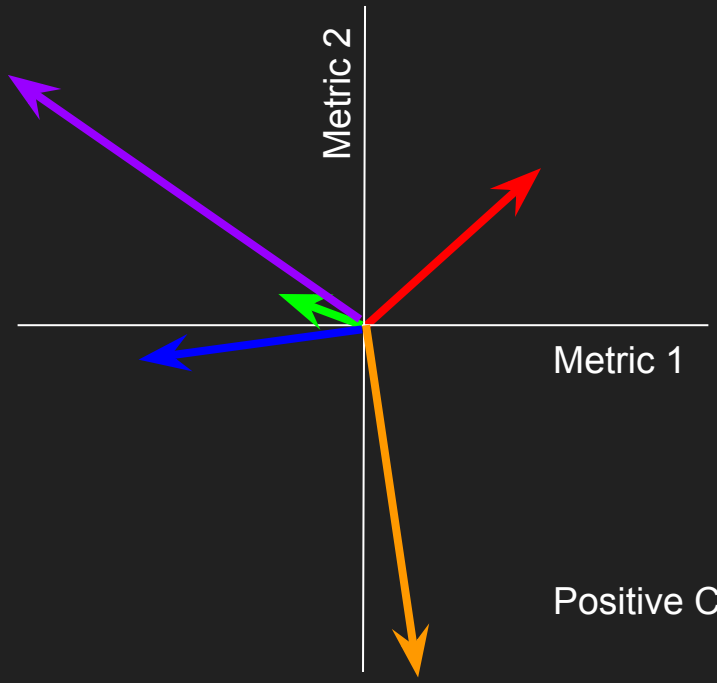
But DIRECTIONALLY BAD!

24:00

Deviation from Ideal, normed by Gilbert Distance
(Just a Metaphorical Diagram, not real numbers)

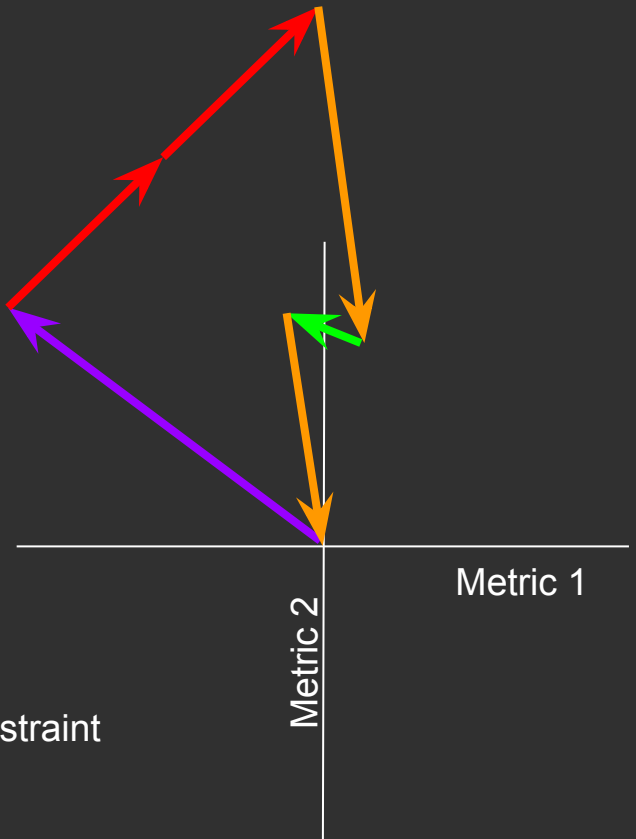


Linear Algebra, Savior



Positive Constants are Only Constraint

Deviations from Ideal, normed by Gilbert Distance
(Just a Metaphorical Diagram, not real numbers)



~~What can we do that is better?~~

We did better!

Current Best Performance (Not Final)

+0.92

(i.e. about 1 standard deviations better than Gilbert)

Second Section:

Graph Isomorphism, Invariants + Cycles

Graph Isomorphism

As a computational question, it is unknown whether or not GI as a decision problem is in P or NPC, it is one of 7 remaining “intermediate” problems, of the 37 initially described.

Metascientific Analysis…?

Graph Invariants

- Number of Vertices
- Number of Edges
- Girth, Diameter
- Degree Sequence
- Eigenvalues (Chromatic Polynomial)

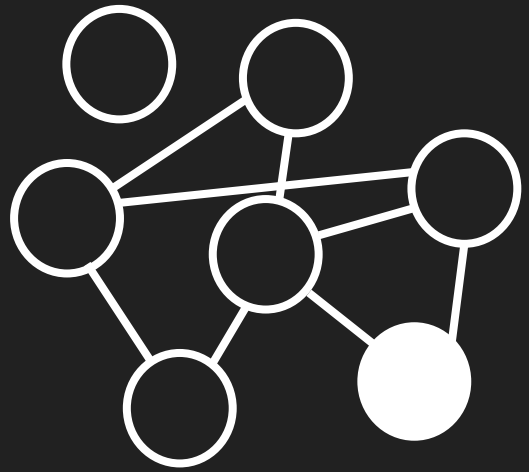
Vertex Invariants

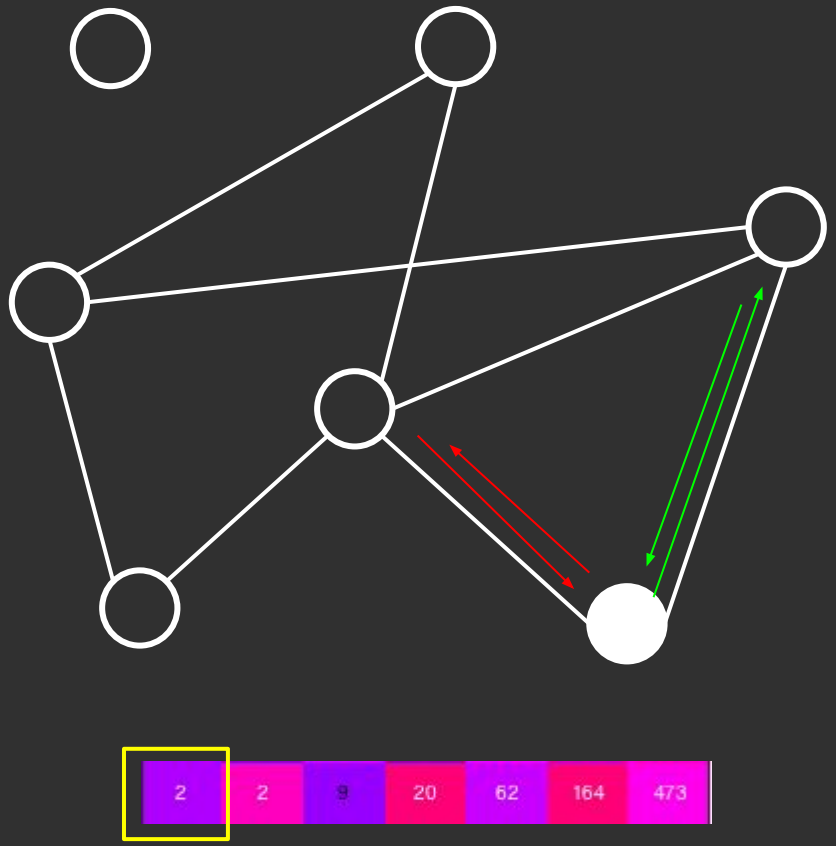
- Degree
- Number of Triangles
- Connectivity Sequence

Diagnostic tests with $\alpha > 0$ and $\beta = 0$

Cycles Invariant (VERTEX)

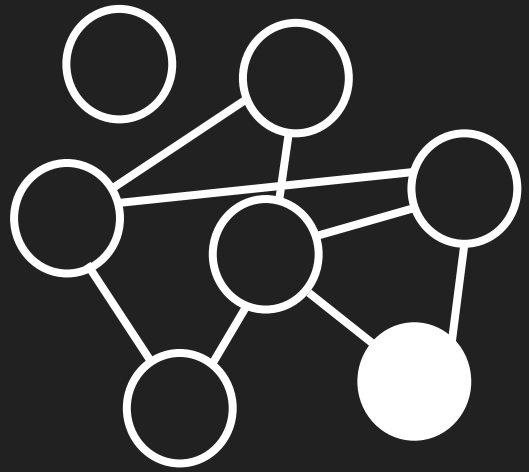
Counts the number of closed walks that start and end at a given vertex within the graph.

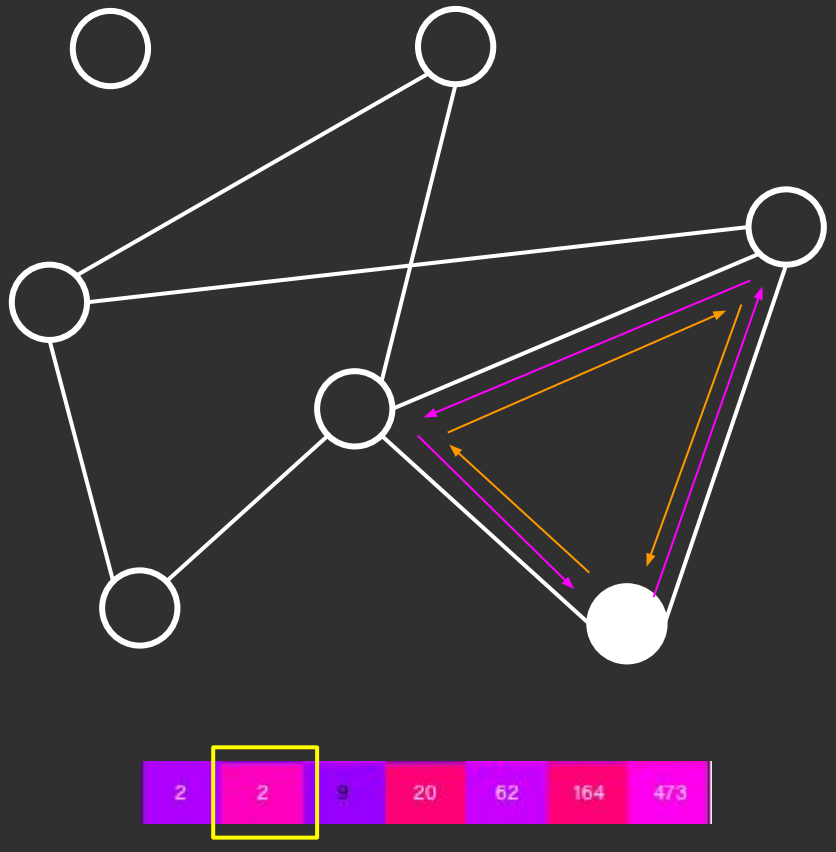




Cycles Invariant (VERTEX)

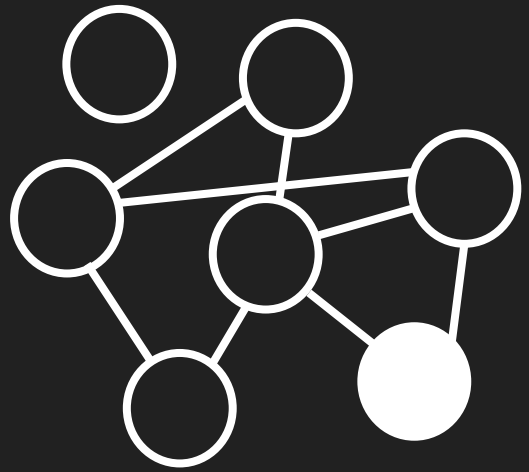
Counts the number of closed walks that start and end at a given vertex within the graph.

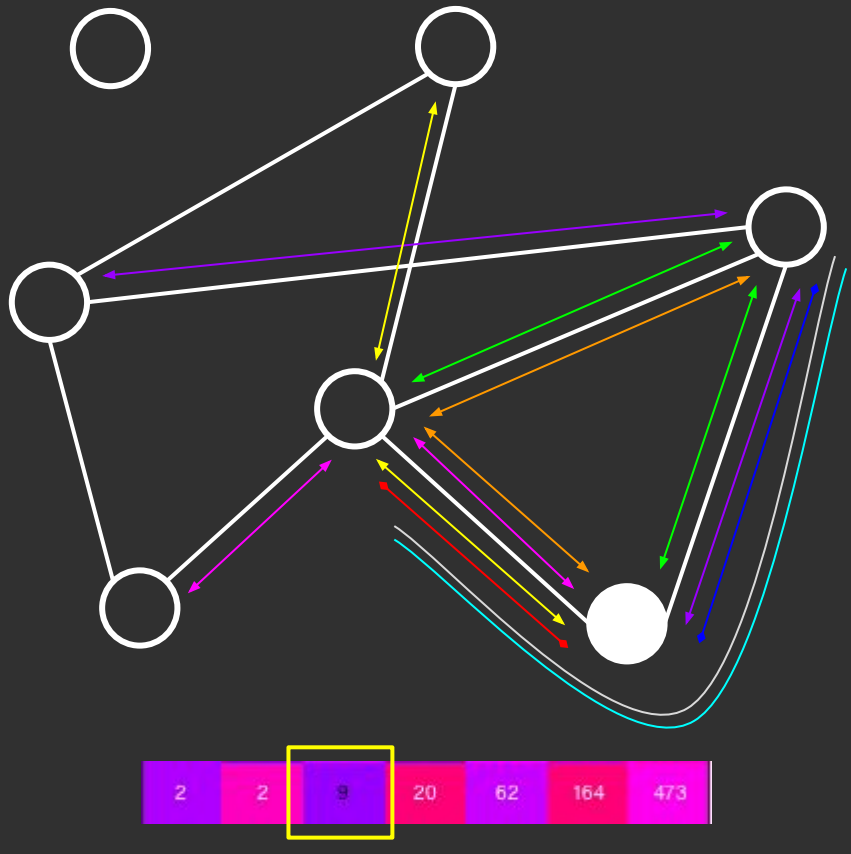




Cycles Invariant (VERTEX)

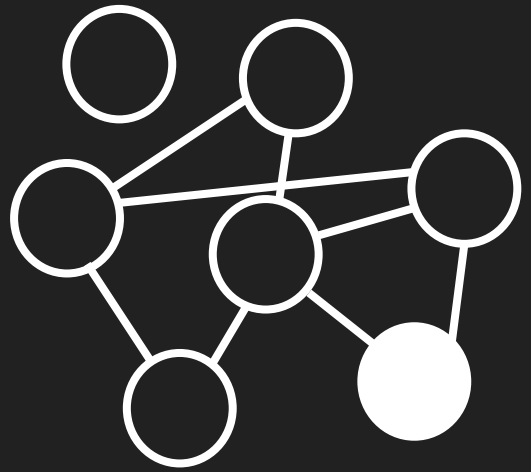
Counts the number of closed walks that start and end at a given vertex within the graph.





Cycles Invariant (VERTEX)

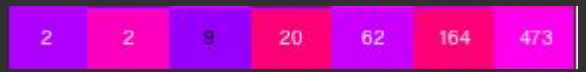
Counts the number of closed walks that start and end at a given vertex within the graph.

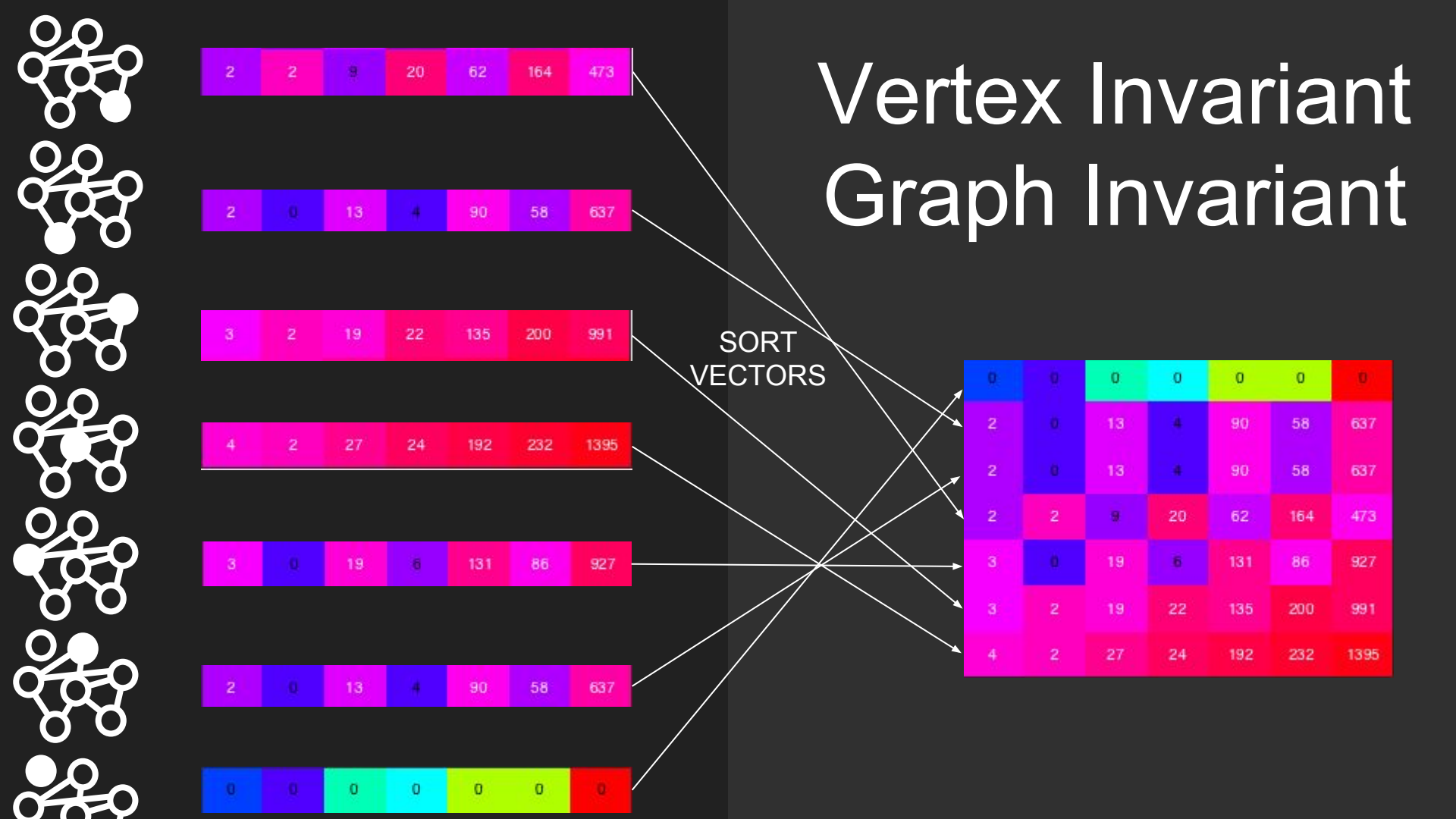


Easy to calculate!

$$A^p[v_i, v_i]$$

for $p > 1$

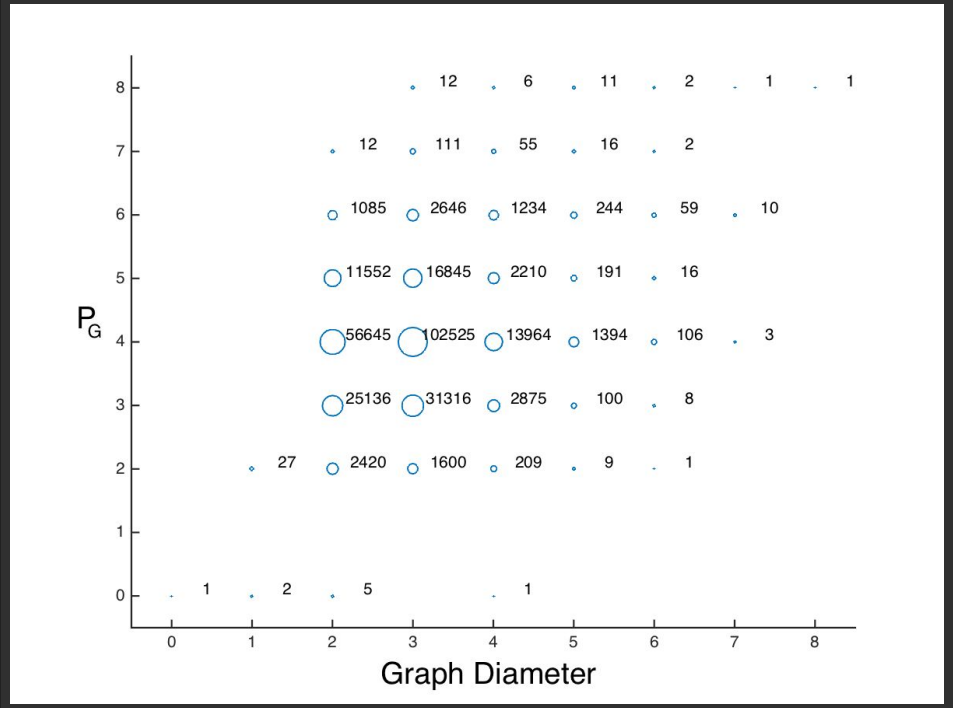



Vertex Invariant
Graph Invariant
SORT
VECTORS
2 2 9 20 62 164 473
2 0 13 4 90 58 637
3 2 19 22 135 200 991
4 2 27 24 192 232 1395
3 0 19 6 131 86 927
2 0 13 4 90 58 637
0 0 0 0 0 0 0
0 0 0 0 0 0 0
2 0 13 4 90 58 637
2 0 13 4 90 58 637
2 2 9 20 62 164 473
3 0 19 6 131 86 927
3 2 19 22 135 200 991
4 2 27 24 192 232 1395

How long a cycle to consider?

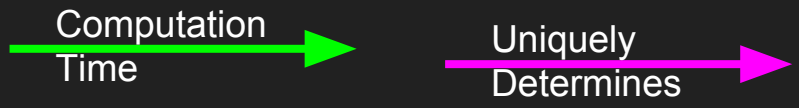
Long story…
Proof is complicated but cool!
Answer is

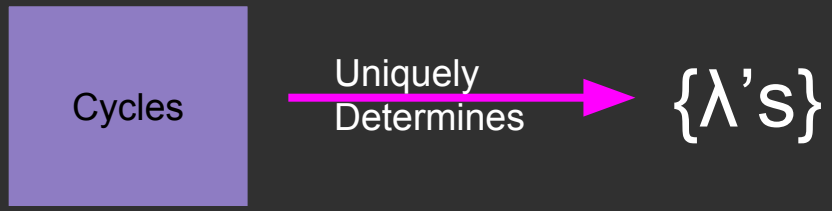
N-1 if N is odd
N-2 if N is even

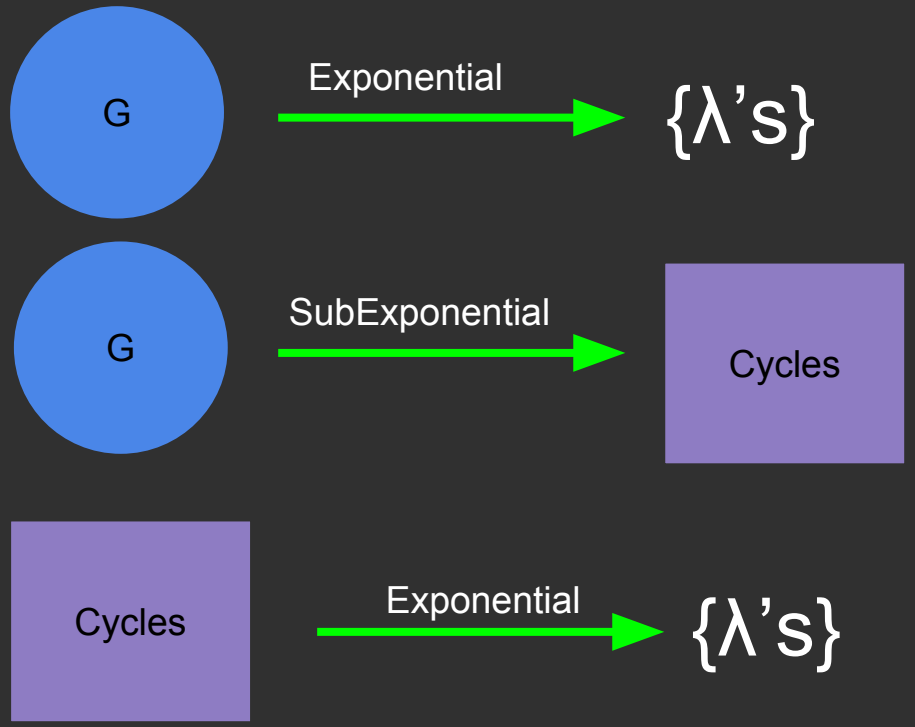


32:00

Significant Result: Cycles Encodes the Chromatic Polynomial







Co-Cycles Graphs Exist, but are very rare

Novel Result

None exist with N < 10

In the set of all graphs of size 10, there exist only 116

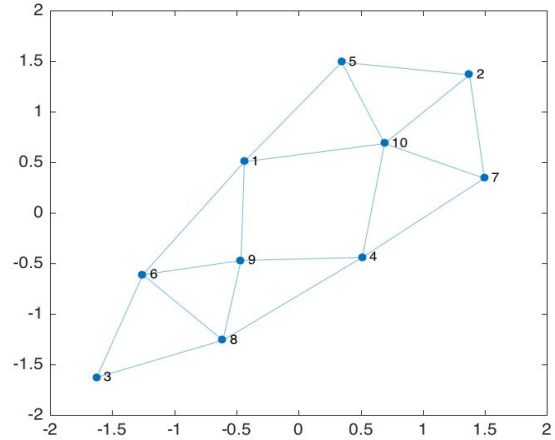
That is 1 / 100,000

Not Trivial to compute

Lazy Evaluation, Stream Processing, Memoization and Efficient Memory Allocation... Thank you Prf. Marison, all the concepts!

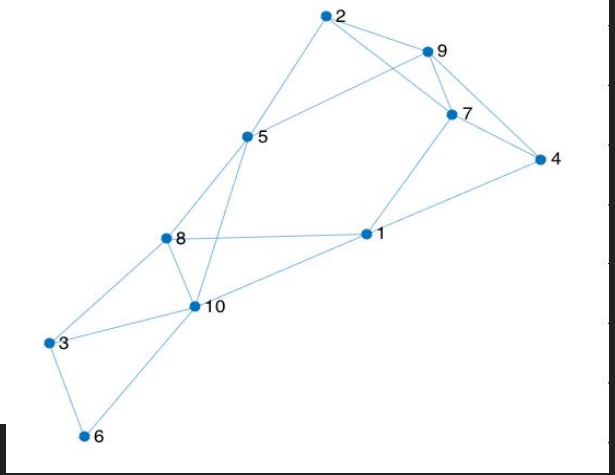
Copaths Graphs Example

v10e18-C [l?qa`iTu_]

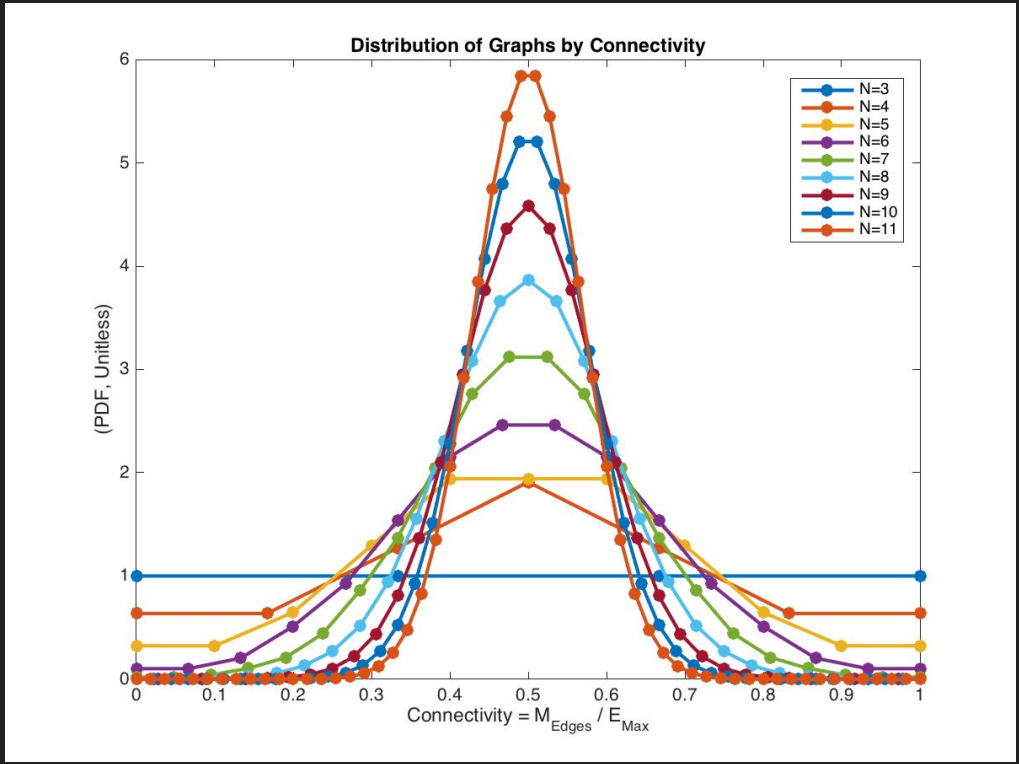


	2	3	4	5	6	7	8	9	10
3-C	-0.62	-0.79	-0.87	-0.97	-0.99	-1.04	-1.05	-1.08	-1.09
5-E	-0.11	-0.22	-0.20	-0.24	-0.22	-0.24	-0.23	-0.23	-0.23
7-G	-0.11	-0.22	-0.20	-0.24	-0.22	-0.24	-0.23	-0.23	-0.23
2-B	-0.11	-0.18	-0.26	-0.26	-0.33	-0.32	-0.35	-0.35	-0.37
1-A	-0.11	0.20	0.14	0.27	0.26	0.31	0.31	0.34	0.34
4-D	-0.11	0.20	0.14	0.27	0.26	0.31	0.31	0.34	0.34
6-F	0.23	0.14	0.20	0.15	0.17	0.15	0.16	0.15	0.15
8-H	0.23	0.14	0.20	0.15	0.17	0.15	0.16	0.15	0.15
9-I	0.23	0.26	0.31	0.33	0.34	0.35	0.36	0.37	0.37
10-J	0.48	0.47	0.54	0.54	0.55	0.55	0.56	0.56	0.56

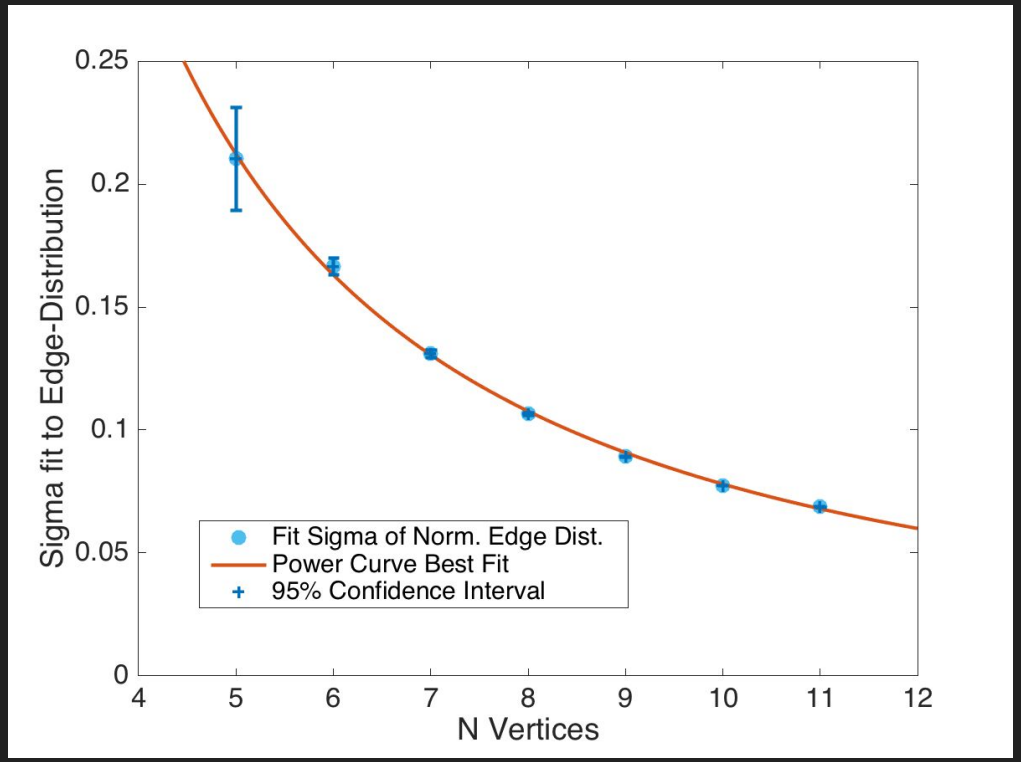
v10e18-D



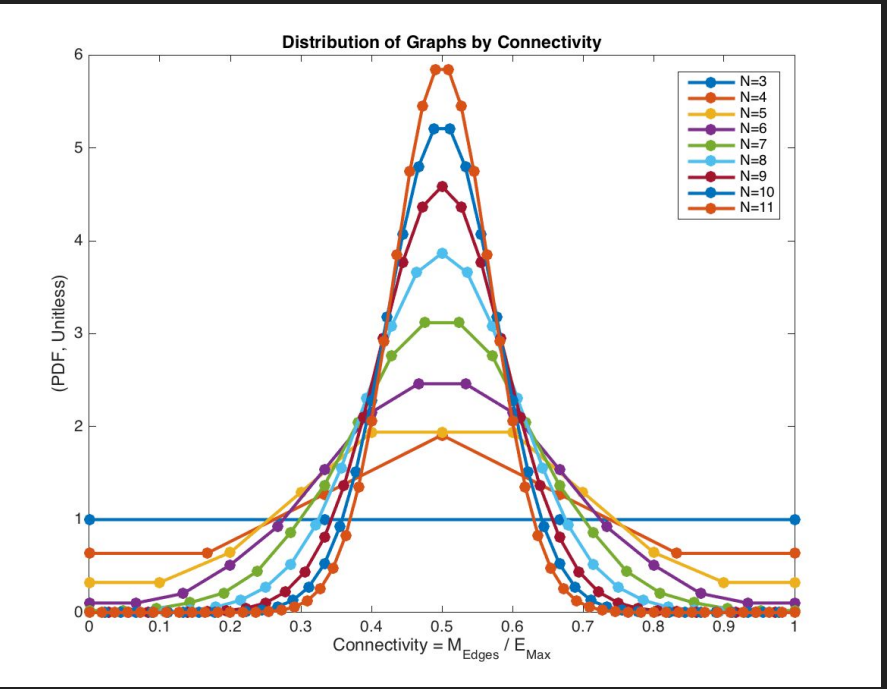
Unexpected Result: Half-Connected Graphs *not* the most difficult to discriminate against!

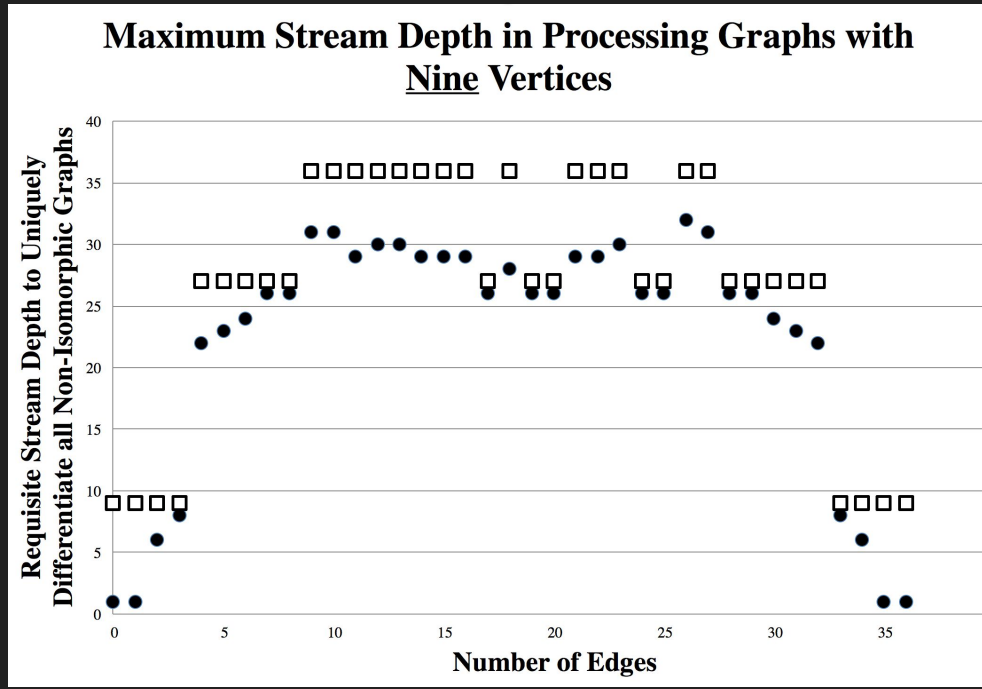


Unexpected Result: Half-Connected Graphs *not* the most difficult to discriminate against!

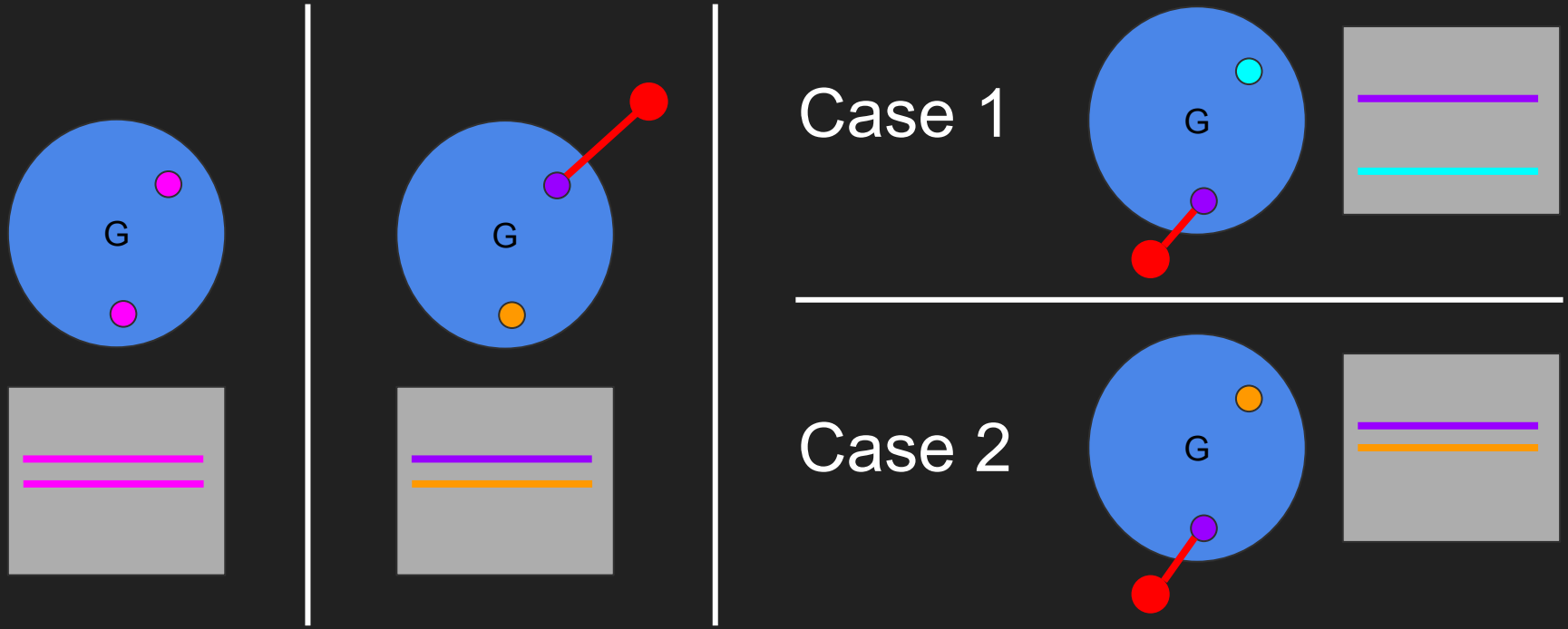


Unexpected Result: Half-Connected Graphs *not* the most difficult to discriminate against!

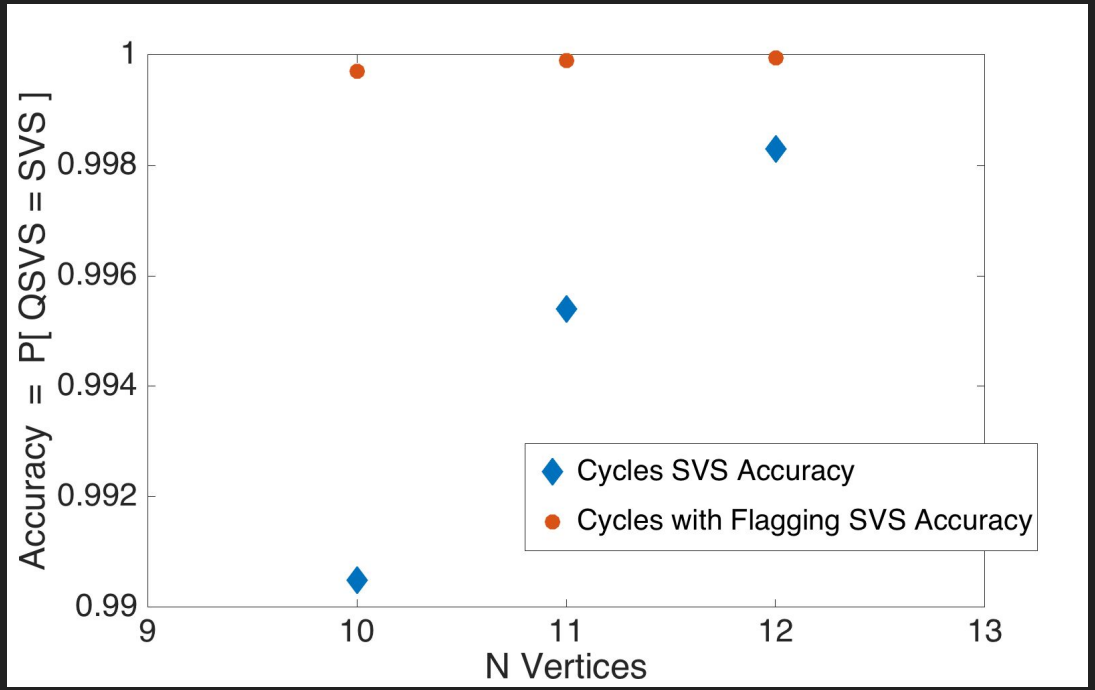




Augmented Paths
(Significantly better discrimination, marginal cost)

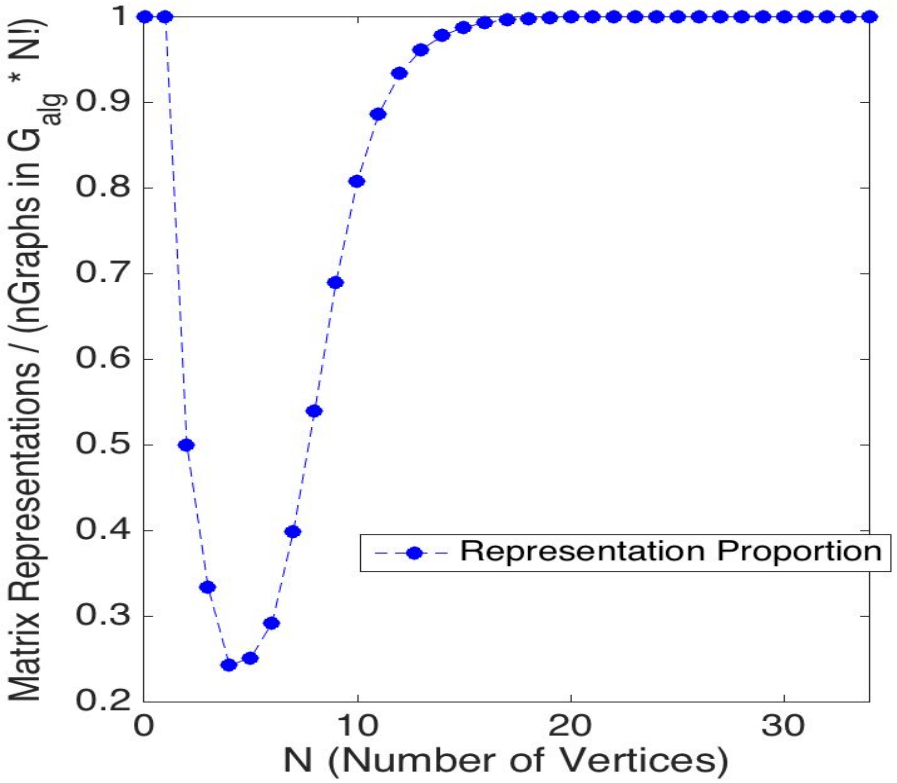


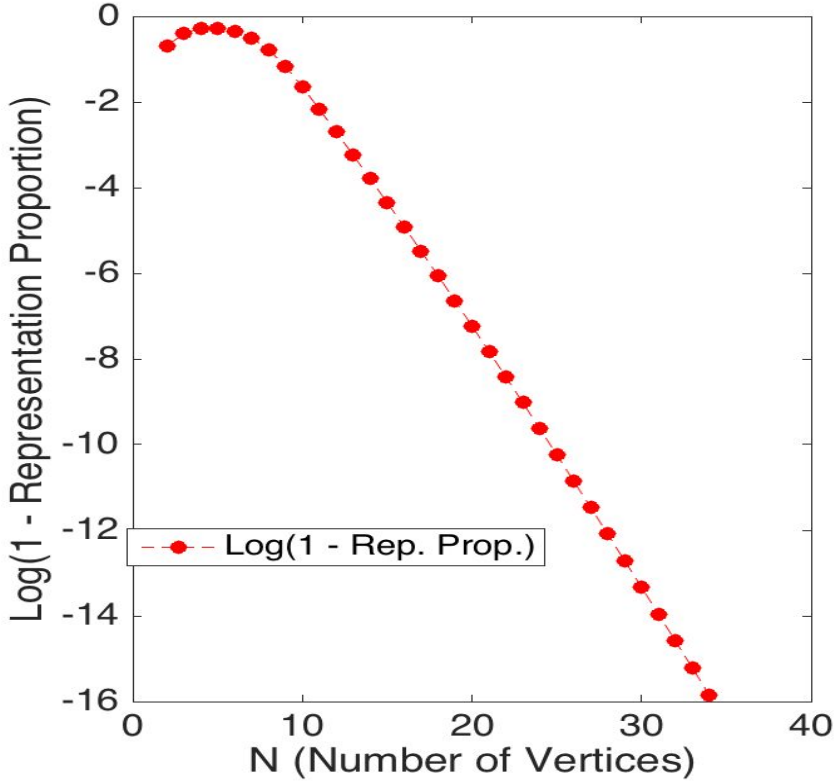
Augmented Paths Dramatically Improves Performance



Appears to asymptote… why?

Most graphs have one automorphism!

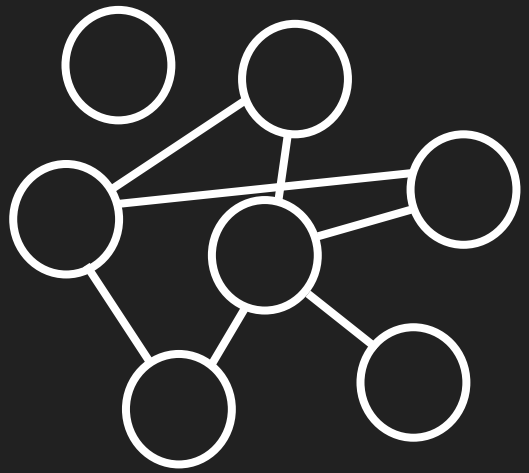




Third Section, Unfinished:

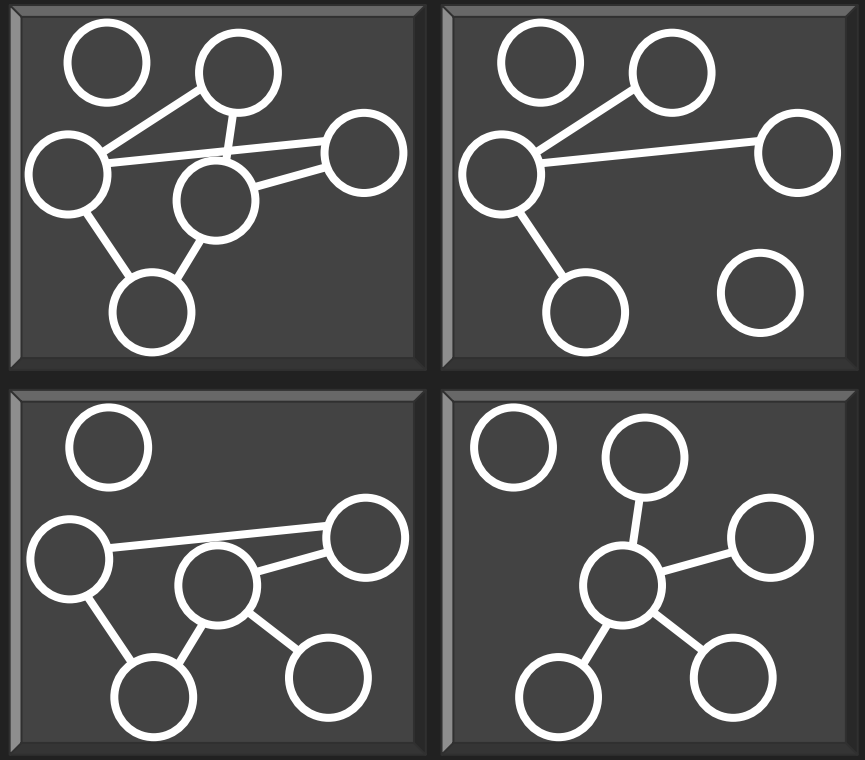
Applying Cycles to the Reconstruction Hypothesis

Original Graph *G*



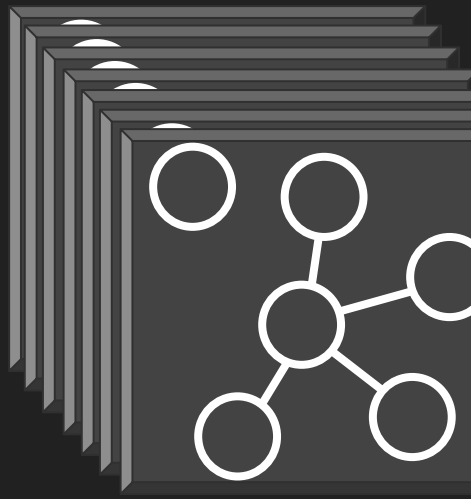
4/7 of the CARDS of *G*

Card = Vertex Deleted Subgraph

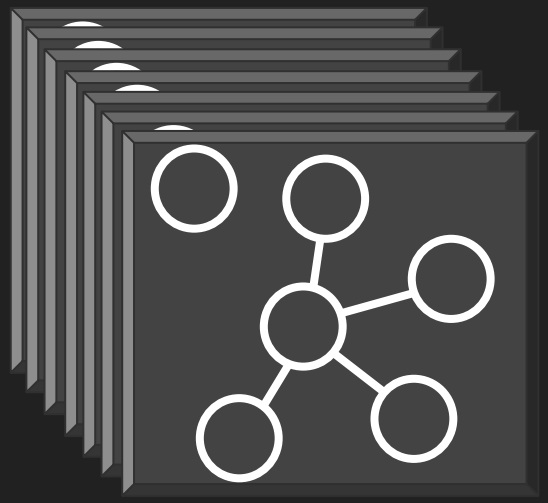


Deck(*G*)

= N Cards of G

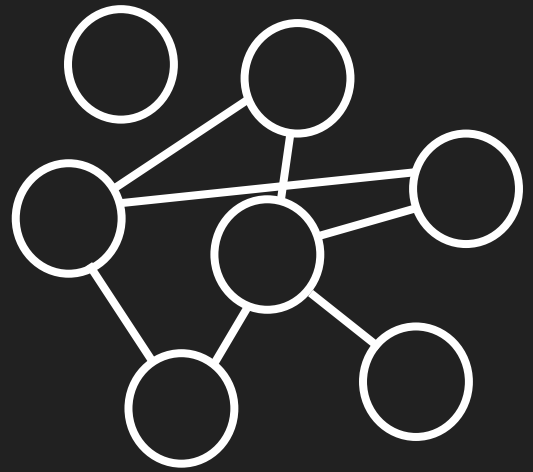


Reconstruction Hypothesis Claims that



Deck (*G*)

Uniquely Determines



Original Graph *G*

McKay verified up to 11 Vertices in 1999

He used a generation mechanism which guaranteed that hypothetical matches would occur within a certain distance, minimizing memory usage, but requiring large amounts of contiguous CPU time. Not directly distributable.

Which is equivalent to:

$\nexists$ G, H where

$\mathrm{Deck}(G) \cong \mathrm{Deck}(H)$ and $G \ncong H$

1,018,997,864	11
165,091,172,592	12

McKay’s method took about two years equivalent of CPU time.

The same methodology is not feasible for 12 Vertices.

We need to distribute!

Alternative: a (distributable) algorithm

Take a graph G of size 11

For nEdgesAdded from maxDegree(G) to 10

S = all supergraphs of G with nEdgesAdded additional edges to the new vertex

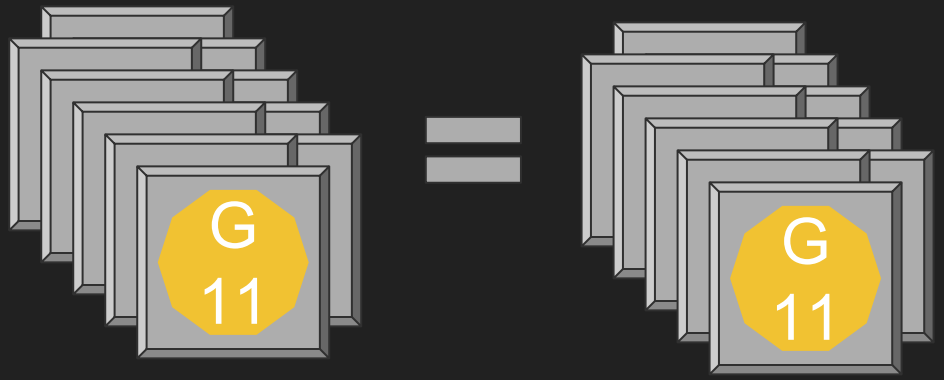
Canonically Label S, turn in to set

Run Heuristics (like Cycles) on each member of the set to test for deck similarity

Run canonical labeler on each deck of S, report back any failures

Running time in optimized C averaged over 10000 test graphs: 0.23 seconds

Must Exist 12 G11s, which do we choose?



Most Graphs have their maximum degree w/ incidence of 1… choose the max!

.23 seconds

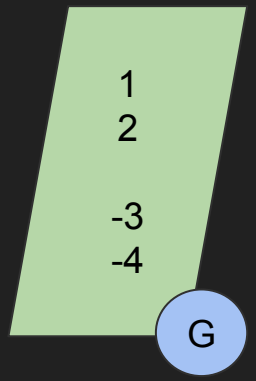
* 1,018,997,864 graphs of size 11

= 2712 days (1 Server)
= 90 days (30 Servers)
= 2 days (GPU Estimate)

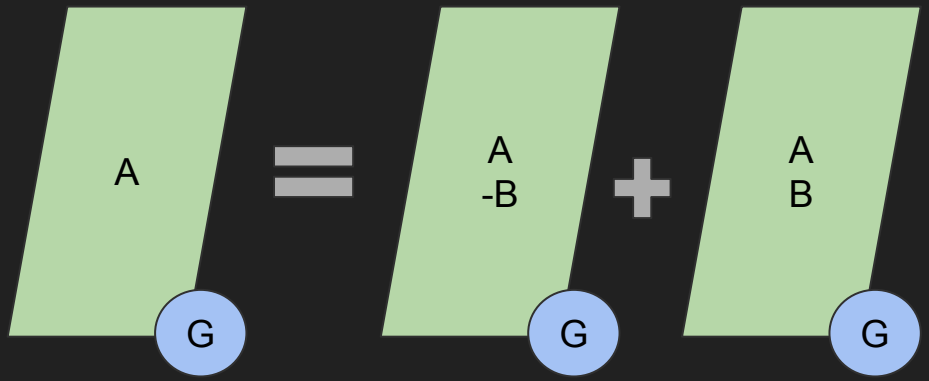
Next step:
Optimize C + run real world trial run

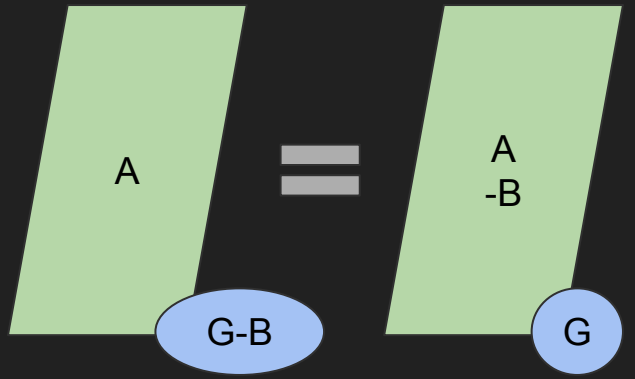
Triangle Inequality:

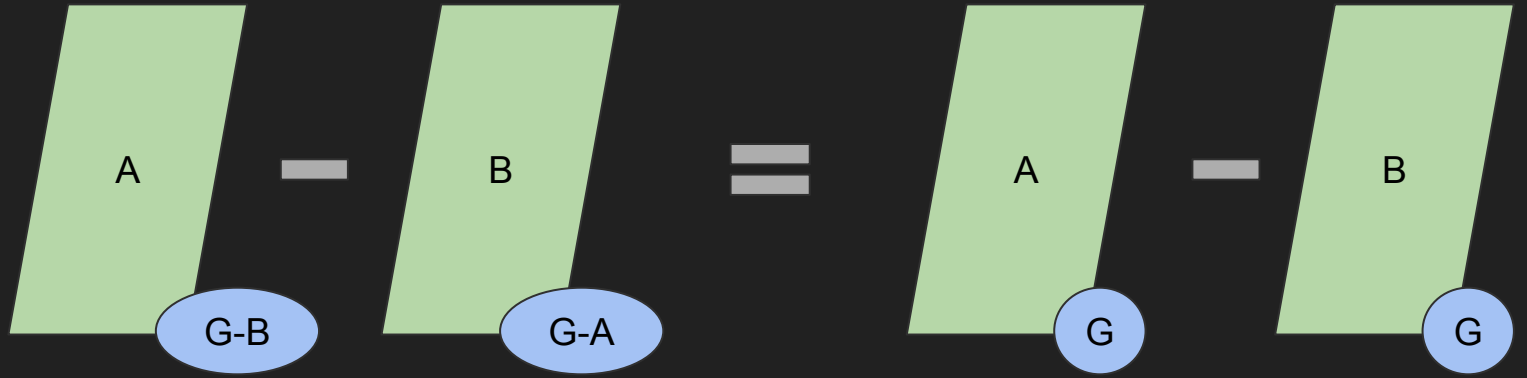
A step toward a proof of general correctness

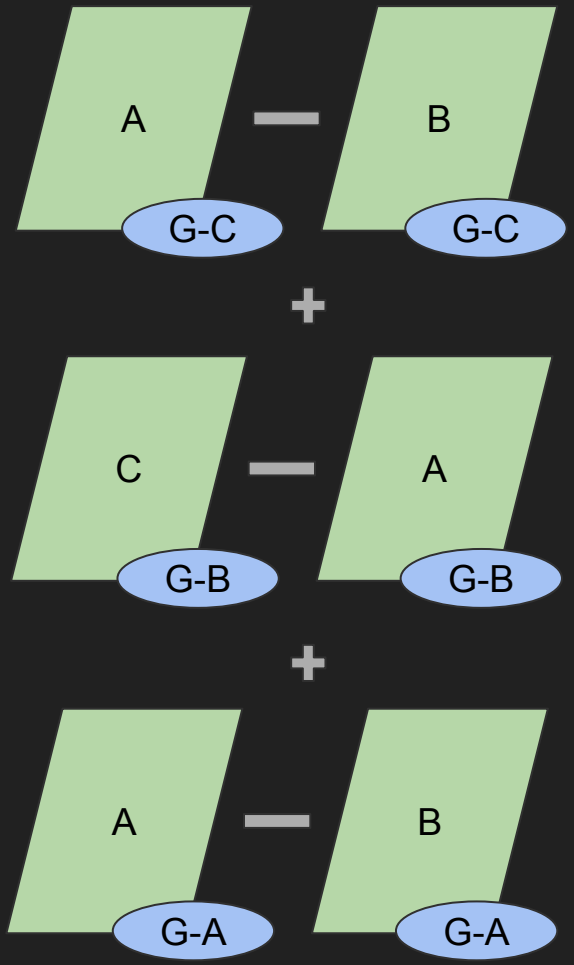


= Number of Cycles which pass through vertices 1 and 2 but not through 3 or 4 in graph G


A
G
A
-B
G
A
B
G


A
G-B
A
-B
G


A
G-B
B
G-A
A
G
B
G



A useful tool for ruling out potential mappings between the vertices on each card and the cards themselves. Particularly for Valid Deck Checking (an open Q)

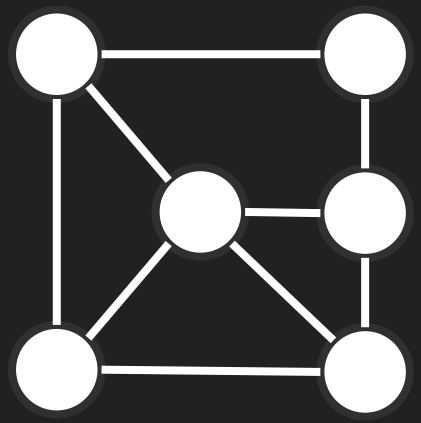
Thank You Professor Storer!

Also many thanks to
Advisors, Friends, Family and the Department

An Open Source Project

1) Version Control enables multi-machine work
2) Science should not just be the final result, process is important
3) Everyone works better in public

github.com/gbdubs/thesis



0	1	0	0	1	0
1	0	1	1	0	0
0	1	0	1	0	1
0	1	1	0	1	1
1	0	0	1	0	1
0	0	1	1	1	0

```
6 100101 100101 111000
E   j    C    w
```

0	0	0	1	0	1
0	0	1	0	1	1
0	1	0	1	1	0
1	0	1	0	1	0
0	1	1	1	0	1
1	1	0	0	1	0

```
6 001011 011110 101000
E   L    ^    G
```