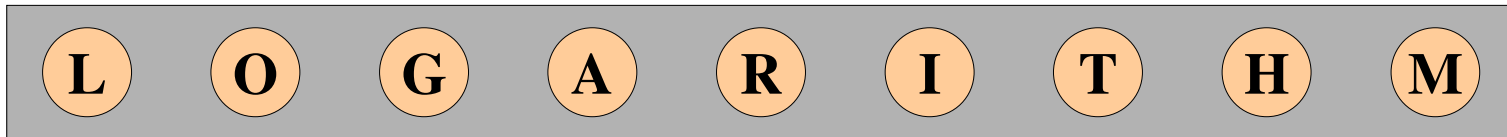
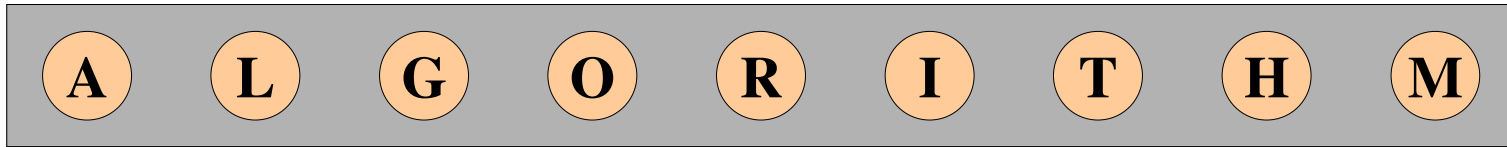


# A Warm-Up Problem

**Q:** Is “algorithm” a permutation of “logarithm”?

**A:** \_\_\_\_\_



Don't turn the page...

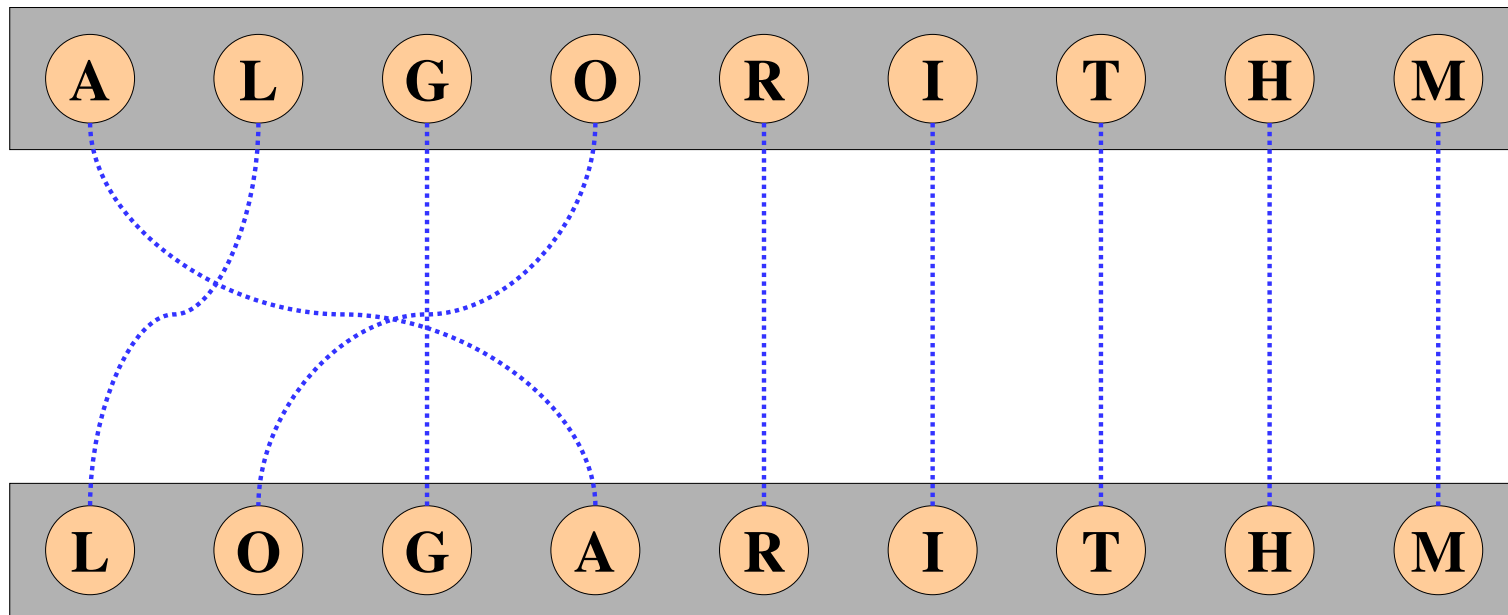
instead, consider what books you'd  
take with you...to an island...

...or on an airplane...across the ocean,  
but, please, don't turn the page...

# A Possible Solution

**Q:** Is “algorithm” a permutation of “logarithm”?

**A:** *Yes*, they share characters of equal frequency.



Complexity is  $O(n^2)$

# An Alternative Solution

**Q:** Is “algorithm” a permutation of “logarithm”?

**A:** *Yes*,  $\text{sort}(\text{“algorithm”}) = \text{sort}(\text{“logarithm”}) = \text{“aghilmort”}$ !

A L G O R I T H M

L O G A R I T H M

**IDEA:** Use *lexicographic sorting* to obtain a *minimum canonical isomorph*

A G H I L M O R T

**QUESTION:** How can we apply this idea to an unlabeled graph with edges?

Complexity is  $\Theta(n \cdot \log n)$

# Logarithmic Coloring: How Hard is it to Determine Isomorphism?

Chris Augeri<sup>1\*</sup>, Barry Mullins<sup>1</sup>, Rusty Baldwin<sup>1</sup>,  
Dursun Bulutoglu<sup>2</sup>, and Leemon Baird<sup>3</sup>

Department of Electrical and Computer Engineering<sup>1</sup>

Department of Mathematics and Statistics<sup>2</sup>

Air Force Institute of Technology, Wright-Patterson AFB, Dayton, OH  
{chris.augeri, barry.mullins, rusty.baldwin, dursun.bulutoglu}@afit.edu

Department of Computer Science<sup>3</sup>

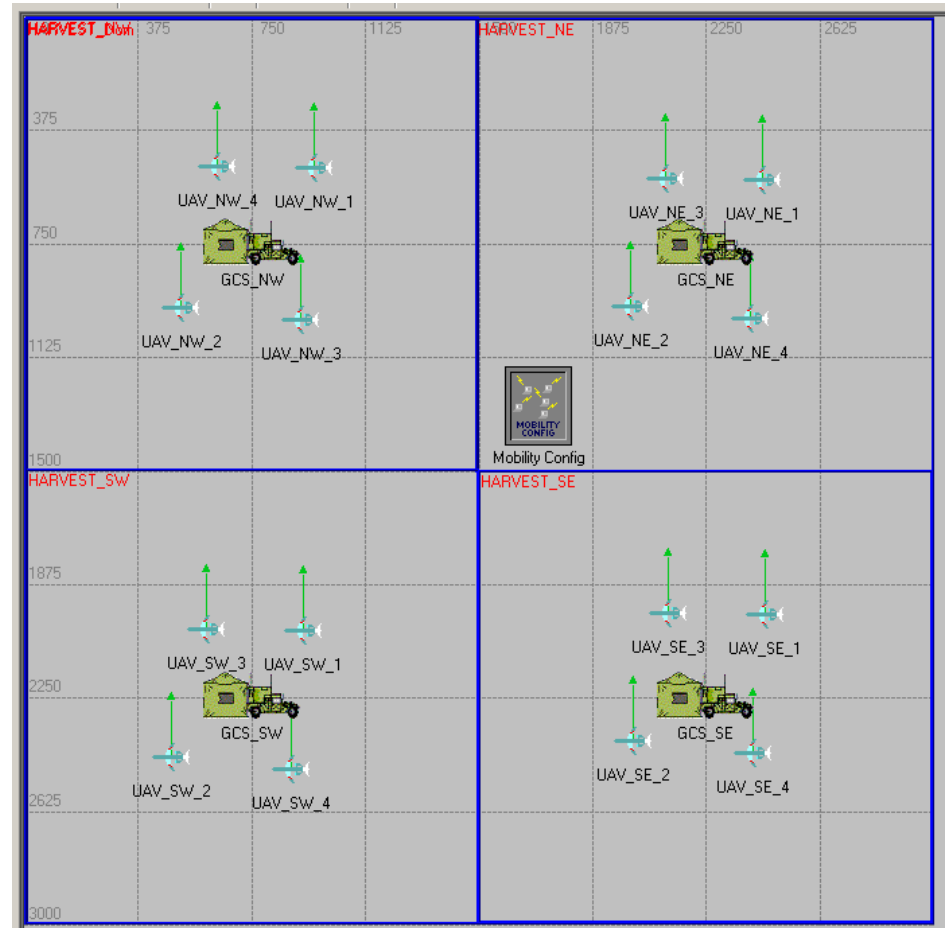
United States Air Force Academy, Colorado Springs, CO  
leemon.baird@usafa.edu

*presented to the*

44th Midwestern Graph Theory Conference (MIGHTY XLIV)  
Wright State University, Dayton, OH, Saturday, 12 May 2007

# Information Management in UAV Swarms

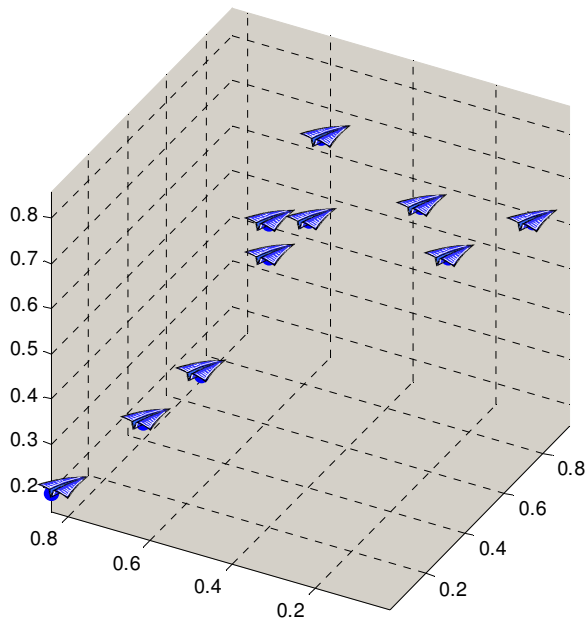
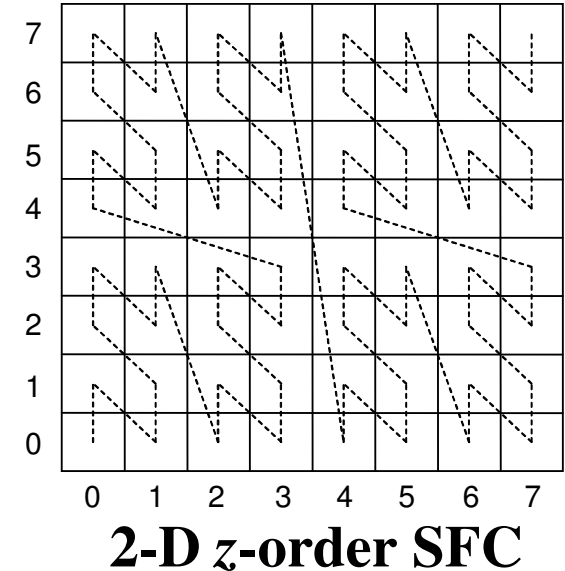
- Protocols & Pedagogy
  - Computer Architecture
  - Software Engineering
- Modeling Frameworks
  - HARVEST
  - JOCosim
- Data Management
  - XML Compression
  - Inverted Skip Graphs
  - IsoCanon (today's talk)\*
    - Deciding Isomorphism
    - PageRank & IsoCanon
    - Logarithmic Coloring



*This research is supported in part by the Air Force Communications Agency. The views expressed are those of the authors and do not reflect the official policy or position of the U.S. Air Force, Dept. of Defense, or U.S. Gov't.*

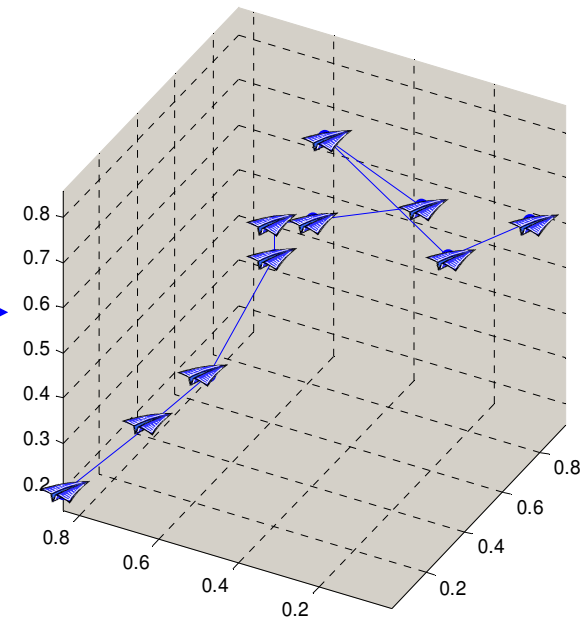
# Linearizing $k$ -Dimensional Data

- **Applications:** range queries, logistics, network security
- **Dimensions:** node coordinates, time, sensor readings
- **Input:** map  $k$ -D data to 2-D distance matrix
- **Methods**
  - **Static:** space-filling curves,  $k$ -D trees
  - **Dynamic:** SVD, multi-dimensional scaling (MDS)
  - **Canonical:** equivalent to determining isomorphism!



**UAV Swarm**

*Vertex  
Ordering*



**A Possible Linearization**

# Determining Matrix Isomorphism

- Equivalent:  $G_1 \cong G_2 \leftrightarrow \mathbf{A}_1 \cong \mathbf{A}_2$
- Convenient:  $\mathbf{A}_2 = \mathbf{P} \cdot \mathbf{A}_1 \cdot \mathbf{P}^{-1}, O(n^3)$
- Efficient:  $\mathbf{A}_2 = \mathbf{P} \cdot \mathbf{A}_1 \cdot \mathbf{P}^T, O(n^2)$

Assume  $G$  is

1. simple
2. connected

	u	v	w	x	y	z
u	0	1	0	1	0	0
v	1	0	0	0	1	1
w	0	0	0	1	1	0
x	1	0	1	0	1	0
y	0	1	1	1	0	1
z	0	1	0	0	1	0

=

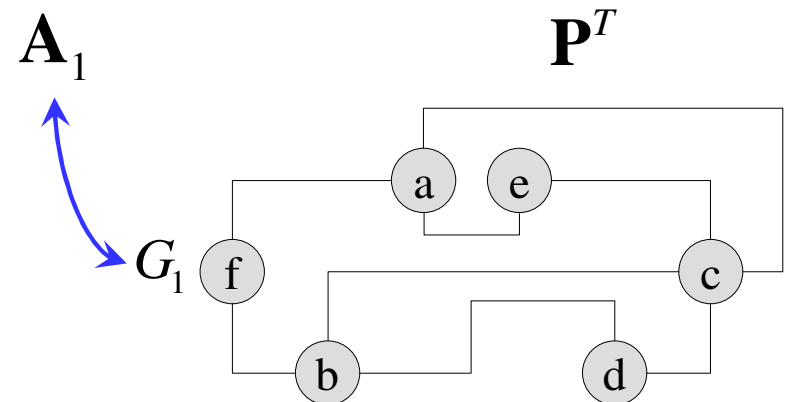
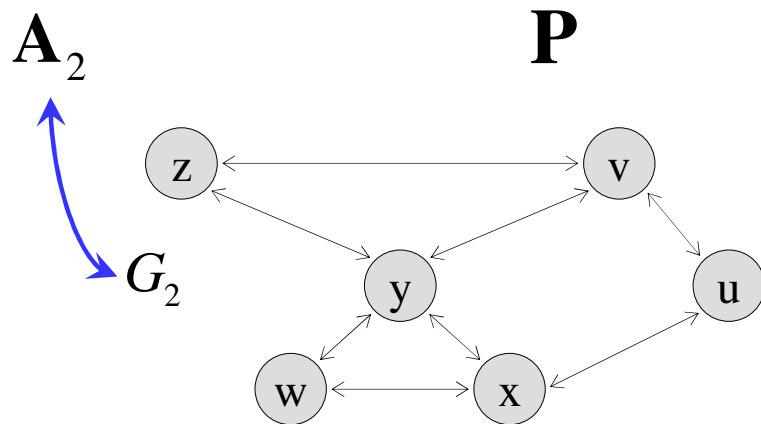
	1	2	3	4	5	6
1	0	0	0	0	0	1
2	1	0	0	0	0	0
3	0	0	0	1	0	0
4	0	1	0	0	0	0
5	0	0	1	0	0	0
6	0	0	0	0	1	0

×

	a	b	c	d	e	f
a	0	0	1	0	1	1
b	0	0	1	1	0	1
c	1	1	0	1	1	0
d	0	1	1	0	0	0
e	1	0	1	0	0	0
f	1	1	0	0	0	0

×

	1	2	3	4	5	6
1	0	1	0	0	0	0
2	0	0	0	1	0	0
3	0	0	0	0	1	0
4	0	0	1	0	0	0
5	0	0	0	0	0	1
6	1	0	0	0	0	0



# A Template for Deciding Isomorphism

## 1. Compare invariants

$$\Psi = \left[ |V|, |E|, \text{sort}(\{\deg(v_i)\}), \text{eig}(\mathbf{A}), \dots \right]$$

	z	w	u	x	v	y
z	0	0	0	0	1	1
w	0	0	0	1	0	1
u	0	0	0	1	1	0
x	0	1	1	0	0	1
v	1	0	1	0	0	1
y	1	1	0	1	1	0

## 2. Compute canonical isomorph, e.g., $\text{MCI}(\mathbf{A}) \in \text{NP}$

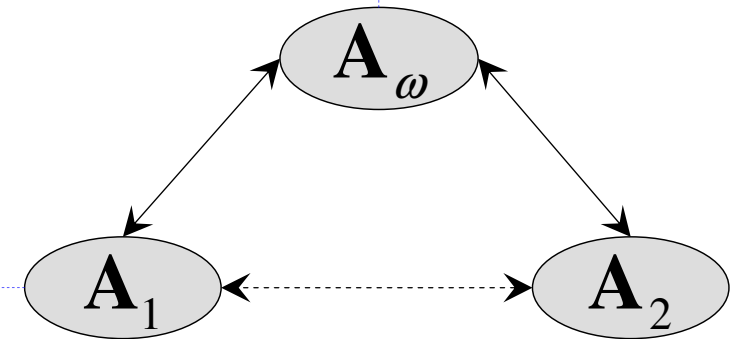
$$\begin{aligned} \text{MCI}(\mathbf{A}) &= \min(\text{num}(\mathbf{A}_i)) \\ &= 000011101011011_2 \end{aligned}$$

$$\mathbf{A}_\omega = \mathbf{P}_\omega \cdot \mathbf{A} \cdot \mathbf{P}_\omega^T$$

*The hard part!*

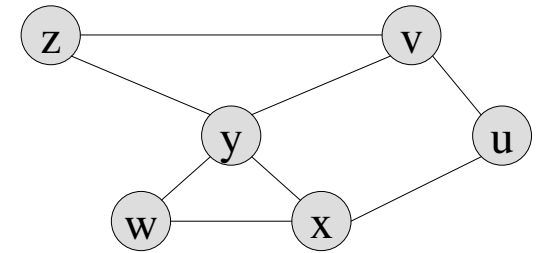
## 3. Compare canonical isomorphs

$$(\mathbf{A}_1)_\omega \equiv (\mathbf{A}_2)_\omega \iff \mathbf{A}_1 \cong \mathbf{A}_2$$



# The Motivating Approach: PageRank

- Heart of **Google**'s web page rankings
- Ranks pages by leading eigenvector [PBM+99]
  - perturbs matrix s.t. it is positive & row-stochastic
  - by Perron-Frobenius theorem, it exists & is unique
  - computable in  $O(n^3)$  time
- Yields canonical isomorph on ~75% of random graphs, e.g., WWW
- If duplicate entries exist, may not be canonical



	u	v	w	x	y	z
u	0	1	0	1	0	0
v	1	0	0	0	1	1
w	0	0	0	1	1	0
x	1	0	1	0	1	0
y	0	1	1	1	0	1
z	0	1	0	0	1	0
Σ	2	3	2	3	4	2

	u	v	w	x	y	z
u	0.025	0.308	0.025	0.308	0.025	0.025
v	0.450	0.025	0.025	0.025	0.238	0.450
w	0.025	0.025	0.025	0.308	0.238	0.025
x	0.450	0.025	0.450	0.025	0.238	0.025
y	0.025	0.308	0.450	0.308	0.025	0.450
z	0.025	0.308	0.025	0.025	0.238	0.025
Σ	1.000	1.000	1.000	1.000	1.000	1.000

Λ	1.00		y	v	x	u	w	z	
u	0.31		y	0	1	1	0	1	1
v	0.44		v	1	0	0	1	0	1
w	0.31		x	1	0	0	1	1	0
x	0.44		u	0	1	1	0	0	0
y	0.57		w	1	0	1	0	0	0
z	0.31		z	1	1	0	0	0	0

$$\begin{aligned}
 \alpha &= 0.85 \\
 \delta &= (1-\alpha)/n \\
 \mathbf{D} &= \mathbf{0}^{n,n} \\
 D_{i,i} &= \sum A_{i,:} \\
 \mathbf{A}' &= \alpha \cdot \mathbf{D}^{-1} \cdot \mathbf{A} + \delta
 \end{aligned}$$

$$\begin{aligned}
 (\mathbf{A} - \lambda \mathbf{I}) \cdot \mathbf{X} &= 0, \mathbf{X} \neq 0 \\
 \mathbf{A} &= \mathbf{X}_\lambda \cdot \Lambda \cdot \mathbf{X}_\lambda^{-1}
 \end{aligned}$$

$$\begin{aligned}
 \mathbf{P} &= \mathbf{I}^n (\phi(V), :) \\
 \mathbf{A}_\omega &= \mathbf{P} \cdot \mathbf{A} \cdot \mathbf{P}^T
 \end{aligned}$$

# Plan B: Other Information Matrices?

- Lexicographic sorting on eigenvectors yields a canonical isomorph on 75% of *random* graphs?
- Do other information matrices yield a canonical isomorph more often when sorted on lexicographically?
- It is likely such a matrix,  $\mathbf{X}$ , is *unique* up to isomorphism

$$\mathbf{P} \cdot \mathbf{A} \cdot \mathbf{P}^T \iff \mathbf{P} \cdot \mathbf{X} \cdot \mathbf{P}^T$$

- One possibility is the matrix inverse,  $\mathbf{X} = \mathbf{A}^{-1}$ , where  $\mathbf{A} \cdot \mathbf{A}^{-1} = \mathbf{I}$ ; however,  $\mathbf{A}^{-1}$  may not always exist.
- By applying two *isomorphism-preserving perturbations* based on the *signless* Laplacian,  $\mathbf{L} = \mathbf{D} + \mathbf{A}$ , we ensure the inverse of the perturbed matrix exists via  $\mathbf{L}^{+\varepsilon} = \mathbf{D} + \mathbf{A} + \vec{\mathcal{E}} \cdot \mathbf{I}$ .

Thanks, Manley!

# IsoCanon: Main Loop

## 1. Initialize Variables

$$z = 1$$

$$\mathbf{A}_\omega = \mathbf{A}_{\omega_1} = \mathbf{A}_{\omega_2} = \mathbf{A}$$

## 2. Run Iteration

`iso_canon_iter(A, n, t)`

## 3. Limit Cycle?

$$z \equiv \lceil \log_2(n_\beta) \rceil + 1$$

✓  $\mathbf{A}_\omega \equiv \mathbf{A}_{\omega_1}$

✓  $\mathbf{A}_\omega \equiv \mathbf{A}_{\omega_2}$

## 4. Update History

$$z = z + 1$$

$$\mathbf{A}_{\omega_2} = \mathbf{A}_{\omega_1}$$

$$\mathbf{A}_{\omega_1} = \mathbf{A}_\omega$$

## Key Ideas

1. Apply  
Isomorphism-Preserving Perturbations

$$\mathbf{A}_1 \cong \mathbf{A}_2 \iff \mathbf{A}'_1 \cong \mathbf{A}'_2$$

2. Construct  
Deterministic Information Matrix  
(*unique up to isomorphism*)

$$\mathbf{P} \cdot \mathbf{A} \cdot \mathbf{P}^T \iff \mathbf{P} \cdot \mathbf{A}^{-1} \cdot \mathbf{P}^T$$

3. Sort Lexicographically  
on Deterministic Information Matrix

$$\text{sort\_lex}(\mathbf{X} \quad \mathbf{I}) \rightarrow \mathbf{P}$$

# IsoCanon: One Iteration

*Perturb & Obtain Inverse*

**1. Add  $\beta$ -Vertex**

$$\mathbf{A}_\beta = \begin{bmatrix} \mathbf{0}^{1,1} & \mathbf{1}^{1,n} \\ \mathbf{1}^{n,1} & \mathbf{A} \end{bmatrix}$$

**2. Dominate Diagonal**  
(*signless Laplacian mod*)

$$\mathbf{A}_\beta^{+\varepsilon} = \mathbf{A}_\beta + \mathbf{D}_\beta + \mathbf{D}_\beta^{-1}$$

$$n_\beta = n + 1$$

**3. Compute Inverse**

$$\mathbf{S}_\beta = (\mathbf{A}_\beta^{+\varepsilon})^{-1}$$

$$\mathbf{S}'_\beta = \mathbf{S}_\beta(2:n_\beta, :)$$

*Construct Deterministic Information Matrix*

**6. Construct Information Matrix**

$$\mathbf{X}_\beta = \begin{bmatrix} \mathbf{T}'_\beta & \mathbf{T}_\beta & \mathbf{I}^{n,n} \end{bmatrix}$$

**5. Sort Row Vectors**  
(*could be orbits!*)

$$\mathbf{T}'_\beta = \text{sort\_vecs}(\mathbf{T}_\beta)$$

**4. Round Entries**  
(*due to finite precision*)

$$t \in (-\infty, -15]$$

$$\mathbf{T}_\beta = \text{round}(\mathbf{S}'_\beta, t)$$

*Sort Lexicographically on Information Matrix & Apply Permutation*

**7. Sort Lexicographically**

$$n_c = 2 \cdot n_\beta$$

$$\mathbf{X}'_\beta = \text{sort\_lex}(\mathbf{X}_\beta, [1:n_c])$$

**8. Extract P-Matrix**

$$\text{col}_s = 2 \cdot n_\beta + 1$$

$$\text{col}_e = 2 \cdot n_\beta + n$$

$$\mathbf{P}_\omega = \mathbf{X}_\omega(:, \text{col}_s : \text{col}_e)$$

**9. Permute Input**

$$\mathbf{A}_\omega = \mathbf{P}_\omega \cdot \mathbf{A} \cdot \mathbf{P}_\omega^T$$

# Steps 1 & 2: Apply Isomorphism-Preserving Perturbations

$\beta$ -Vertex [Mat78, PrD84, CRS97]  
 Aids distinguishing co-spectral graphs

**Diagonal Dominance** [Has04, Var04]  
 $A^{-1}$  exists using *modified* signless Laplacian & Gershgorin Circle theorem

1. Add  $\beta$ -Vertex

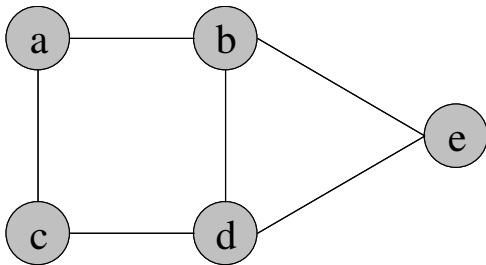
$$A_{\beta} = \begin{bmatrix} \mathbf{0}^{1,1} & \mathbf{1}^{1,n} \\ \mathbf{1}^{n,1} & A \end{bmatrix}$$

2. Dominate Diagonal

$$A_{\beta}^{+\varepsilon} = D_{\beta} + A_{\beta} + D_{\beta}^{-1}$$

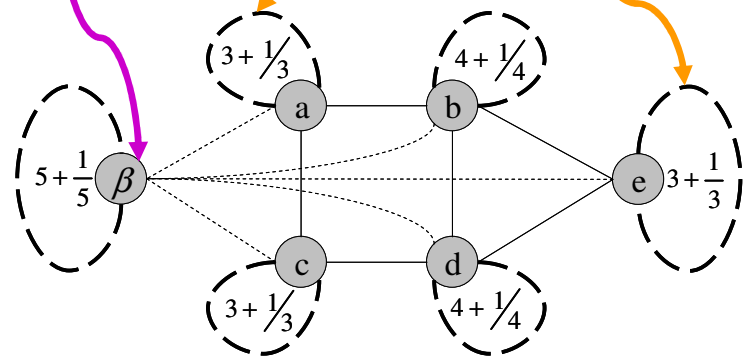
$$n_{\beta} = n + 1$$

“House” Graph  
 [Gol80 (2nd ed., 2004), pg. 14]



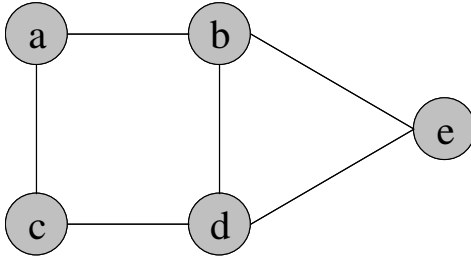
	a	b	c	d	e
a	0	1	1	0	0
b	1	0	0	1	1
c	1	0	0	1	0
d	0	1	1	0	1
e	0	1	0	1	0

Thanks, Terry!



	$\beta$	a	b	c	d	e
$\beta$	$5\frac{1}{5}$	1	1	1	1	1
a	1	$3\frac{1}{3}$	1	1	0	0
b	1	1	$4\frac{1}{4}$	0	1	1
c	1	1	0	$3\frac{1}{3}$	1	0
d	1	0	1	1	$4\frac{1}{4}$	1
e	1	0	1	0	1	$3\frac{1}{3}$

# Steps 3 & 4: Compute & Round Inverse



## 3. Compute Inverse & Remove $\beta$ -vertex

$$\mathbf{S}_\beta = (\mathbf{A}_\beta^{+\varepsilon})^{-1}$$

$$\mathbf{S}'_\beta = \mathbf{S}_\beta(2:n_\beta, :)$$

	$\beta$	a	b	c	d	e
$\beta$	$\frac{11490}{49853}$	$\frac{2370}{49853}$	$\frac{1220}{49853}$	$\frac{2370}{49853}$	$\frac{1220}{49853}$	$\frac{2715}{49853}$
a	$-\frac{2370}{49853}$	$\frac{1488912}{3938387}$	$-\frac{382068}{3938387}$	$-\frac{455355}{3938387}$	$\frac{216168}{3938387}$	$\frac{1341}{49853}$
b	$-\frac{1220}{49853}$	$-\frac{382068}{3938387}$	$\frac{1153772}{3938387}$	$\frac{216168}{3938387}$	$-\frac{242112}{3938387}$	$-\frac{3096}{49853}$
c	$-\frac{2370}{49853}$	$-\frac{455355}{3938387}$	$\frac{216168}{3938387}$	$\frac{1488912}{3938387}$	$-\frac{382068}{3938387}$	$\frac{1341}{49853}$
d	$-\frac{1220}{49853}$	$\frac{216168}{3938387}$	$-\frac{242112}{3938387}$	$-\frac{382068}{3938387}$	$\frac{1153772}{3938387}$	$-\frac{3096}{49853}$
e	$-\frac{2715}{49853}$	$\frac{1341}{49853}$	$-\frac{3096}{49853}$	$\frac{1341}{49853}$	$-\frac{3096}{49853}$	$\frac{17628}{49853}$

	$\beta$	a	b	c	d	e
a	-0.048	0.378	-0.097	-0.116	0.055	0.027
b	-0.024	-0.097	0.293	0.055	-0.061	-0.062
c	-0.048	-0.116	0.055	0.378	-0.097	0.027
d	-0.024	0.055	-0.061	-0.097	0.293	-0.062
e	-0.054	0.027	-0.062	0.027	-0.062	0.354

## 4. Round Entries

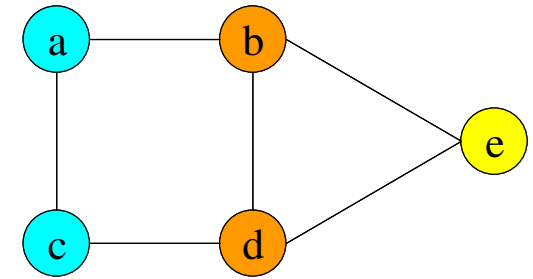
$$t \in (-\infty, -14]$$

$$\mathbf{T}_\beta = \text{round}(\mathbf{S}'_\beta, t)$$

# Step 5: Sort Inverse's Row Vectors

Note same inverse entries for vertices in different orbits!

	$\beta$	a	b	c	d	e
a	-0.048	0.378	-0.097	-0.116	0.055	0.027
b	-0.024	-0.097	0.293	0.055	-0.061	-0.062
c	-0.048	-0.116	0.055	0.378	-0.097	0.027
d	-0.024	0.055	-0.061	-0.097	0.293	-0.062
e	-0.054	0.027	-0.062	0.027	-0.062	0.354



$$\mathbf{P} \cdot \mathbf{A} \cdot \mathbf{P}^T \leftrightarrow \mathbf{P} \cdot \mathbf{A}^{-1} \cdot \mathbf{P}^T$$

	Equal Rows?					
a	-0.116	-0.097	-0.048	0.027	0.055	0.378
b	-0.097	-0.062	-0.061	-0.024	0.055	0.293
c	-0.116	-0.097	-0.048	0.027	0.055	0.378
d	-0.097	-0.062	-0.061	-0.024	0.055	0.293
e	-0.062	-0.062	-0.054	0.027	0.027	0.354

	Equal Rows?
a	{a, c}
b	{b, d}
c	{a, c}
d	{b, d}
e	{e}

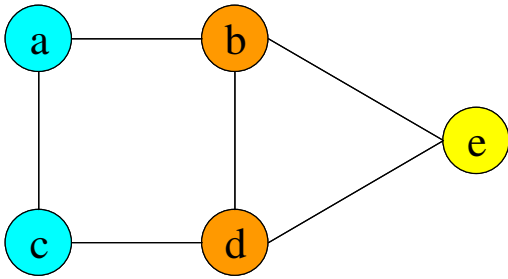
**5. Sort Row Vectors**

$$\mathbf{T}'_{\beta} = \text{sort\_vecs}(\mathbf{T}_{\beta})$$

Sort *within* each row from left to right

We pre-group the rows for *presentation* herein

# Step 6: Construct Information Matrix

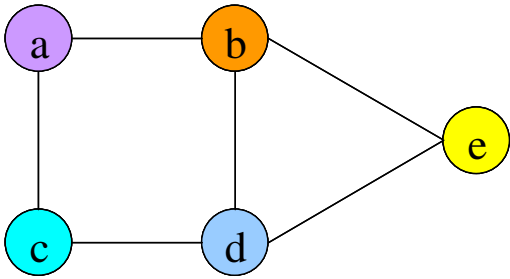


**6. Construct Information Matrix**

$$\mathbf{X}_\beta = [\mathbf{T}'_\beta \quad \mathbf{T}_\beta \quad \mathbf{I}^{n,n}]$$

	$\mathbf{T}'_\beta$	$\mathbf{T}_\beta$					$\mathbf{I}^{n,n}$					
		$\beta$	a	b	c	d	e	a	b	c	d	e
a	{a, c}	-0.048	0.378	-0.097	-0.116	0.055	0.027	1	0	0	0	0
b	{b, d}	-0.024	-0.097	0.293	0.055	-0.061	-0.062	0	1	0	0	0
c	{a, c}	-0.048	-0.116	0.055	0.378	-0.097	0.027	0	0	1	0	0
d	{b, d}	-0.024	0.055	-0.061	-0.097	0.293	-0.062	0	0	0	1	0
e	{e}	-0.054	0.027	-0.062	0.027	-0.062	0.354	0	0	0	0	1

# Step 7: Lexicographically Sort on Constructed Information Matrix



## 7. Sort Lexicographically

$$n_c = 2 \cdot n_\beta$$

$$\mathbf{X}'_\beta = \text{sort\_lex}(\mathbf{X}_\beta, [1:n_c])$$

	$\mathbf{T}'_\beta$	$\mathbf{T}_\beta$						$\mathbf{P}_\omega$				
		$\beta$	a	b	c	d	e	a	b	c	d	e
c	{a, c}	-0.048	-0.116	0.055	0.378	-0.097	0.027	0	0	1	0	0
a	{a, c}	-0.048	0.378	-0.097	-0.116	0.055	0.027	1	0	0	0	0
b	{b, d}	-0.024	-0.097	0.293	0.055	-0.061	-0.062	0	1	0	0	0
d	{b, d}	-0.024	0.055	-0.061	-0.097	0.293	-0.062	0	0	0	1	0
e	{e}	-0.054	0.027	-0.062	0.027	-0.062	0.354	0	0	0	0	1

Sorting terminates,  
unique sorted row determined!

Induced permutation matrix

# Steps 8 & 9: Extract & Apply Permutation to Input Matrix

## 8. Extract Permutation Matrix

$$col_s = 2 \cdot n_\beta + 1$$

$$col_e = 2 \cdot n_\beta + n$$

$$\mathbf{P}_\omega = \mathbf{X}_\omega(:, col_s : col_e)$$

## 9. Permute Input

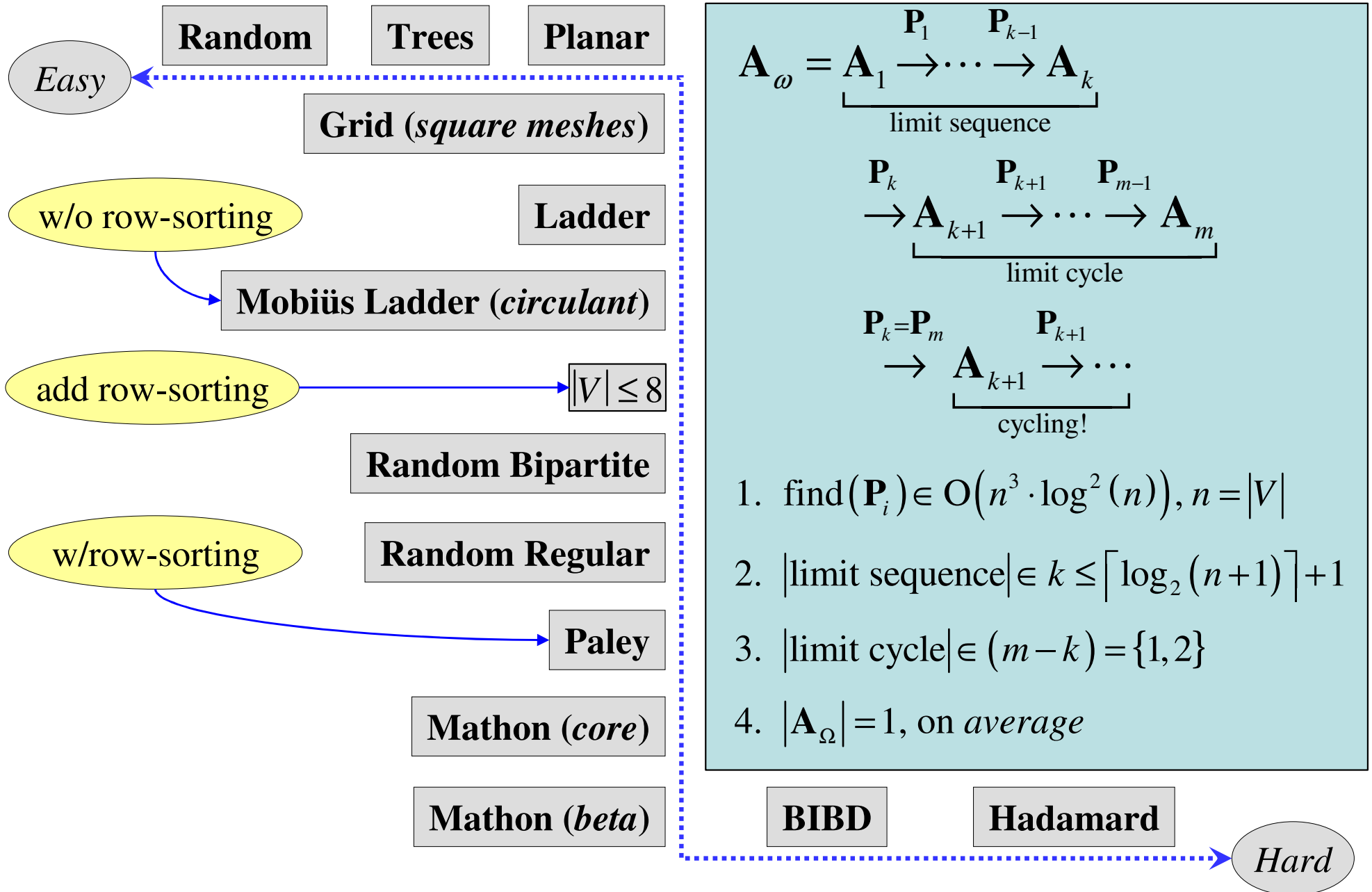
$$\mathbf{A}_\omega = \mathbf{P}_\omega \cdot \mathbf{A} \cdot \mathbf{P}_\omega^T$$

$\mathbf{P}_\omega$					×	$\mathbf{A}$					×	$\mathbf{P}_\omega^T$				
0	0	1	0	0		0	1	1	0	0		0	1	0	0	0
1	0	0	0	0	1	0	0	1	1	0	0	1	0	0		
0	1	0	0	0	1	0	0	1	0	1	0	0	0	0		
0	0	0	1	0	0	1	1	0	1	0	0	0	1	0		
0	0	0	0	1	0	1	0	1	0	0	0	0	0	1		

$\mathbf{A}_\omega$					
	c	a	b	d	e
c	0	1	0	1	0
a	1	0	1	0	0
b	0	1	0	1	1
d	1	0	1	0	1
e	0	0	1	1	0

This process (steps 1–9) may be repeated up to  $(\log_2(n) + 1)$  iterations

# Graph Difficulty



# A New Paradigm

What is the complexity of causing an arbitrary graph to be *easy* to identify?

**OR**

How much *noise* must be added such that an arbitrary graph is *easy* to identify?

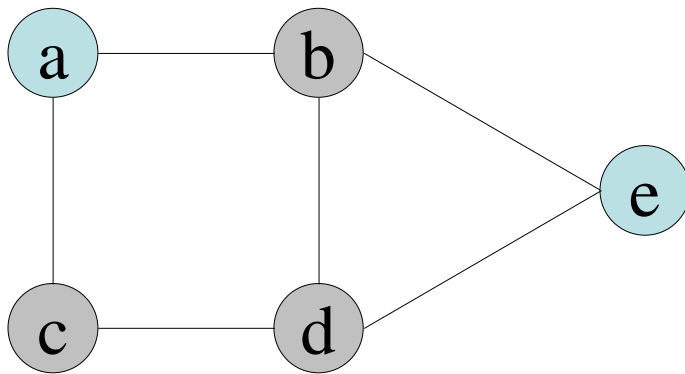
**OR**

How many vertices must be colored such that a graph can be identified in polynomial time?

# Random Coloring Methods

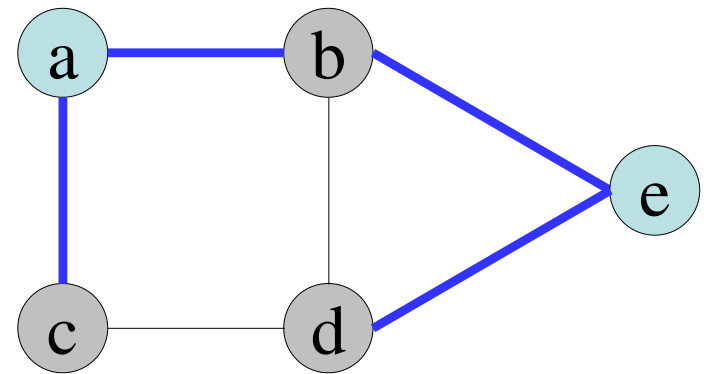
(multiply by constant)

color  $k$  vertices



	a	b	c	d	e
a	2	1	1	0	0
b	1	0	0	1	1
c	1	0	0	1	0
d	0	1	1	0	1
e	0	1	0	1	2

color incident edges  
of  $k$  vertices

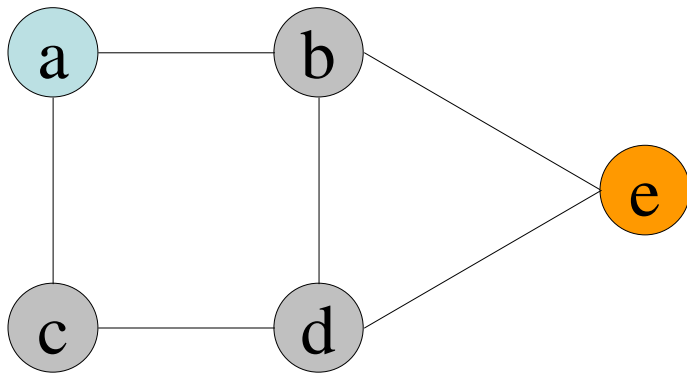


	a	b	c	d	e
a	0	2	2	0	0
b	2	0	0	1	2
c	2	0	0	1	0
d	0	1	1	0	2
e	0	2	0	2	0

# Random Coloring Methods

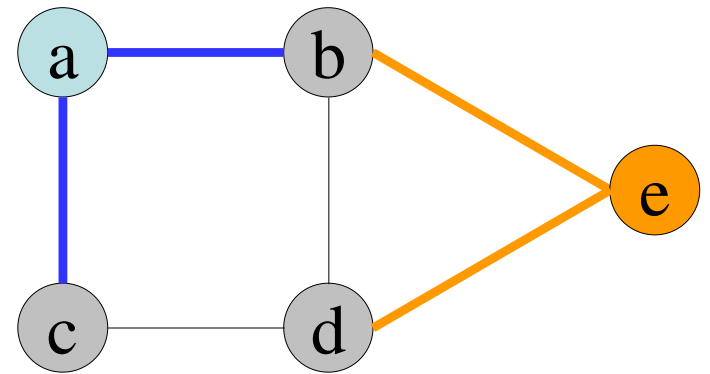
## (multiply by primes)

color  $k$  vertices



	a	b	c	d	e
a	2	1	1	0	0
b	1	0	0	1	1
c	1	0	0	1	0
d	0	1	1	0	1
e	0	1	0	1	3

color incident edges  
of  $k$  vertices



	a	b	c	d	e
a	0	2	2	0	0
b	2	0	0	1	3
c	2	0	0	1	0
d	0	1	1	0	3
e	0	3	0	3	0

# Random Coloring Results

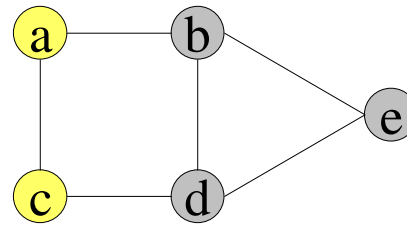
1. Randomly permute, i.e., “shuffle” graph  $\log_2(n)$  times
2. Color  $k$  random vertices by some method, *rand\_coloring*

Result	$k$	Method
<b>Unique Rows</b>	$\lceil \log_2(n) \rceil \cdot \lceil \sqrt{n} \rceil$	$V$ (primes)
<b>IsoCanon</b>	$2 \cdot \lceil \log_2(n) \rceil$	$V$ (constant)
<b>IsoCanon</b>	$2 \cdot \lceil \ln(n) \rceil$	$V$ (primes)
<b>IsoCanon</b>	$\lceil \ln(n) \rceil$	$E$ (primes)
<b>IsoCanon</b>	$\lceil \ln(n) \rceil$	$E$ (constant)

# Deterministic Coloring I (naïve version)

	$\beta$	a	b	c	d	e
$\beta$	1	2	2	2	2	2
a	2	1	2	2	1	1
b	2	2	1	1	2	2
c	2	2	1	1	2	1
d	2	1	2	2	1	2
e	2	1	2	1	2	1

Possible Vertices

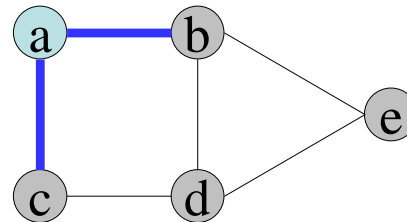


**Matrix Shift:**  $A = A + 1$   
 “spreads” coloring info faster  
**Choose Vertex:** from smallest equivalent row set,  $A(i, :) \cdot 2$

	Equal Rows?						
a	1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
b	3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150
c	1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
d	3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150
e	2	-0.1074	0.0035	0.0035	0.0121	0.0138	0.0138

	$\beta$	a	b	c	d	e
$\beta$	12.1	4	2	2	2	2
a	4	16.2	4	4	2	2
b	2	4	11.1	1	2	2
c	2	4	1	10.1	2	1
d	2	2	2	2	10.1	2
e	2	2	2	1	2	11.1

Chosen Vertex (a)



**Compute Inverse:**  $A'$   
 assess effect of coloring  
**Done Coloring?:** if all rows unique  $\parallel \ln(n)$  vertices colored

	Equal Rows?						
a	3	-0.0783	0.0019	0.0039	0.0090	0.0115	0.0169
b	5	-0.0705	0.0007	0.0045	0.0066	0.0081	0.0115
c	1	-0.0994	0.0007	0.0031	0.0072	0.0103	0.0169
d	4	-0.0719	0.0019	0.0051	0.0066	0.0085	0.0103
e	2	-0.0838	0.0031	0.0039	0.0069	0.0081	0.0085

# Lexicographic Linearization

(equivalent to comparing rows)

Raw Matrix

1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150
1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150
2	-0.1074	0.0035	0.0035	0.0121	0.0138	0.0138

Lexicographic  
Sorting

1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
2	-0.1074	0.0035	0.0035	0.0121	0.0138	0.0138
3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150
3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150

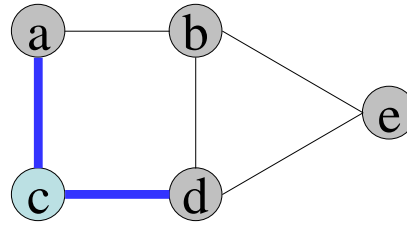
Linearization

<del>1</del>	<del>-0.1080</del>	<del>0.0012</del>	<del>0.0035</del>	<del>0.0118</del>	<del>0.0150</del>	<del>0.0170</del>
<del>1</del>	<del>-0.1080</del>	<del>0.0012</del>	<del>0.0035</del>	<del>0.0118</del>	<del>0.0150</del>	<del>0.0170</del>
2	-0.1074	0.0035	0.0035	0.0121	0.0138	0.0138
<del>3</del>	<del>-0.0993</del>	<del>0.0012</del>	<del>0.0095</del>	<del>0.0121</del>	<del>0.0138</del>	<del>0.0150</del>
<del>3</del>	<del>-0.0993</del>	<del>0.0012</del>	<del>0.0095</del>	<del>0.0121</del>	<del>0.0138</del>	0.0150

# Deterministic Coloring II (less naïve version)

	$\beta$	a	b	c	d	e
$\beta$	12.1	2	2	4	2	2
a	2	10.1	2	4	1	1
b	2	2	10.1	2	2	2
c	4	4	2	16.2	4	2
d	2	1	2	4	11.1	2
e	2	1	2	2	2	9.1

First Possible Vertex (c)

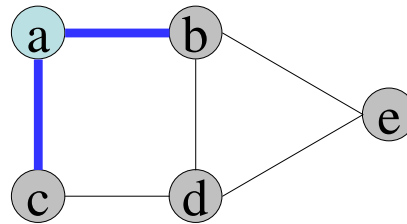


In this example, {a, c} are in the same orbit & there are only 2 available vertices. If more than one vertex, we...

1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
2	-0.1074	0.0035	0.0035	0.0121	0.0138	0.0138
3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150
3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150

	$\beta$	a	b	c	d	e
$\beta$	12.1	4	2	2	2	2
a	4	16.2	4	4	2	2
b	2	4	11.1	1	2	2
c	2	4	1	10.1	2	1
d	2	2	2	2	10.1	2
e	2	2	2	1	2	9.1

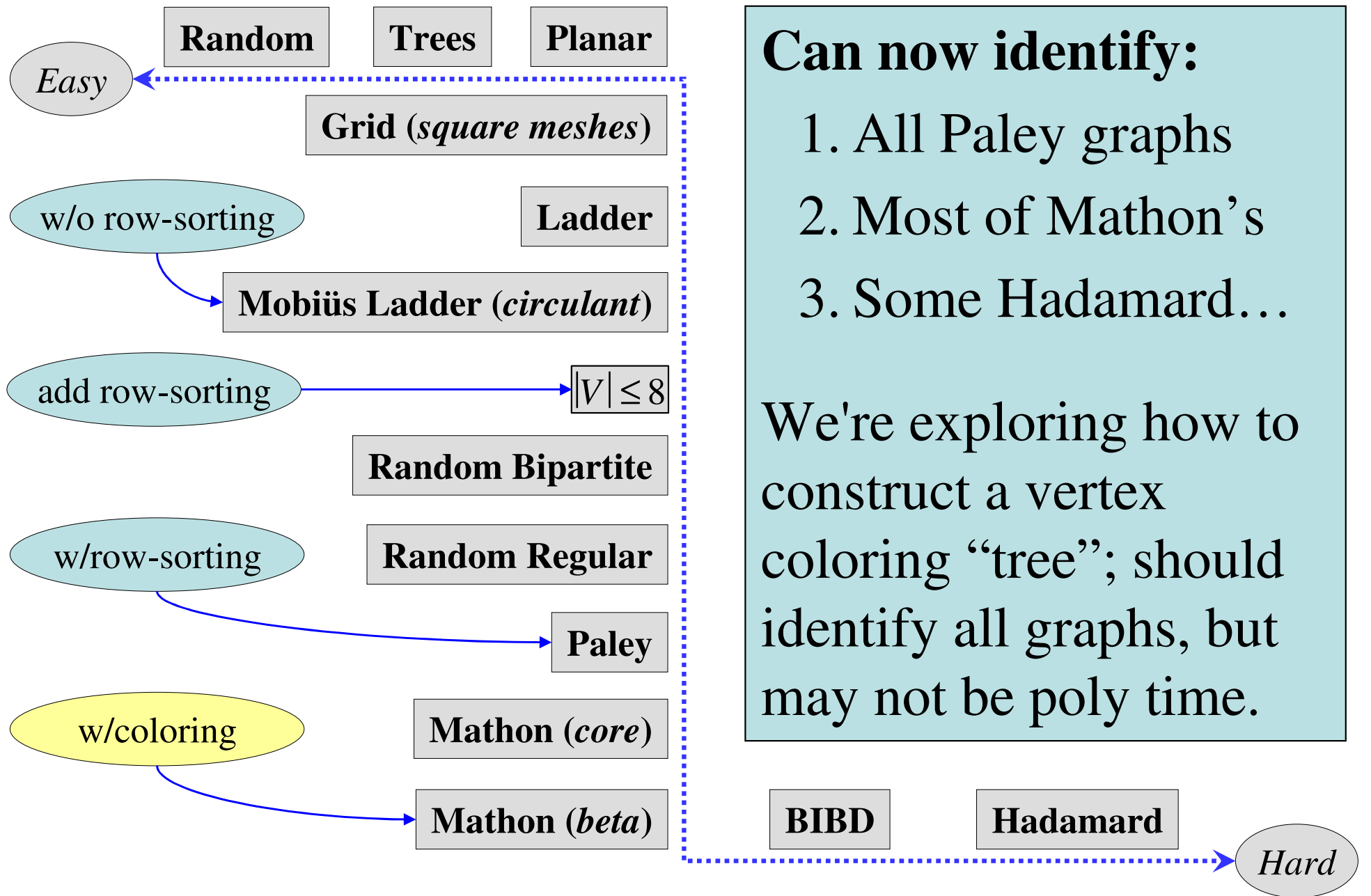
First Possible Vertex (a)



color each vertex in smallest potential equivalence set & select vertex yielding smallest lex-sorted *linearized* inverse.

1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
1	-0.1080	0.0012	0.0035	0.0118	0.0150	0.0170
2	-0.1074	0.0035	0.0035	0.0121	0.0138	0.0138
3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150
3	-0.0993	0.0012	0.0095	0.0121	0.0138	0.0150

# Graph Difficulty Revisited



**Can now identify:**

1. All Paley graphs
2. Most of Mathon's
3. Some Hadamard...

We're exploring how to construct a vertex coloring "tree"; should identify all graphs, but may not be poly time.

# What's Next?

- Study behavior w.r.t. numerical conditioning
- Extend logarithmic coloring method
- Assess usefulness of linearization
  - Data mining
  - Critical nodes
  - User queries

# References

- [BeE96] J. M. Bennett and J. J. Edwards. A graph isomorphism algorithm using pseudoinverses. *BIT Numerical Mathematics*. Springer-Verlag, 36(1):41–53, 1996.
- [BeW04] L. W. Beineke and R. J. Wilson, eds., with Peter J. Cameron. *Topics in Algebraic Graph Theory*. Cambridge University Press, 2004.
- [Gol80] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Elsevier, 1980.
- [HaS04] W. H. Haemers and E. Spence. Enumeration of cospectral graphs. *European Journal of Combinatorics*, 25(2):199–211, 2004.
- [HZL05] P. R. He, W. J. Zhang, and Q. Li. Some further development on the eigensystem approach for graph isomorphism detection. *J. of the Franklin Inst.*, Elsevier, 342(6), 2005.
- [Mat78] R. Mathon. Sample Graphs for Isomorphism Testing. In Proc. of the 9th Southeastern Conference on Combinatorics, Graph Theory, and Computing (CGTC), Boca Raton, FL, *Congressus Numerantium*, 21:499-517, 1978.
- [McK81] B. D. McKay. Practical Graph isomorphism. In *Proc. of the 10th Manitoba Conf. on Num. Mathematics and Comp.*, 45–87. *Congressus Numerantium*, vol. 30, 1981.
- [PBM+99] L. Page, S. Brin, R. Motwani and T. Winograd. *The PageRank Citation Ranking: Bringing Order to the Web*. TR 1999-66, Stanford University, 1999.
- [PrD85] G. M. Prabhu and N. Deo. On the power of a perturbation for testing non-isomorphism of graphs. *BIT*, Springer, 24(3):302–307, 1984.
- [Var04] R. S. Varga. *Geršgorin and His Circles*. Springer, 2004.