

General outline for tutorial T21: Graph Metric Spaces

8:00 - 9:00	José Bento	Part I: Introduction to Graph Distances
9:00 - 9:30	Stratis Ioannidis	Part II a: A Family of Tractable Graph Metrics
9:30 - 10:00	Coffee break	ICC Capital Suite Foyer (Level 3)
10:00 - 10:30	Stratis Ioannidis	Part II b: A Family of Tractable Graph Metrics
10:30 - 11:30	Tina Eliassi-Rad	Part III: Non-backtracking Matrix, Graph Distance, and Metric Embedding
11:30 - 12:00	Q&A	Slides

<https://bit.ly/2KJFP4B>



Grants IIS-1741197 & IIS-1741129



Northeastern



BOSTON COLLEGE

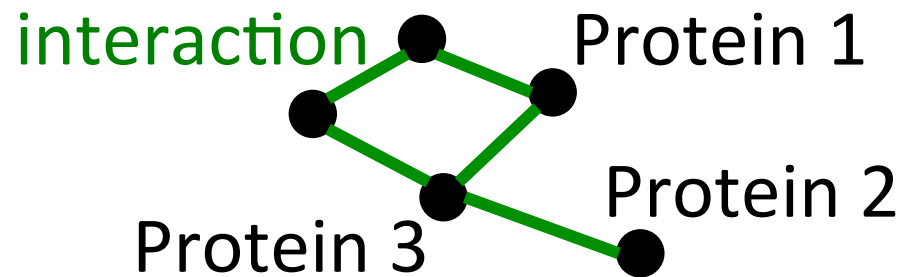
Part I: Introduction to Graph Distances

1. Quick review about graphs
2. Applications where comparing graphs is important
3. Graph distances
4. Why a metric?
5. Scalable alignment algorithms

Graphs, their comparison, and applications

Unlabeled, undirected, non-weighted

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}) \quad \mathcal{V} = [n]$$

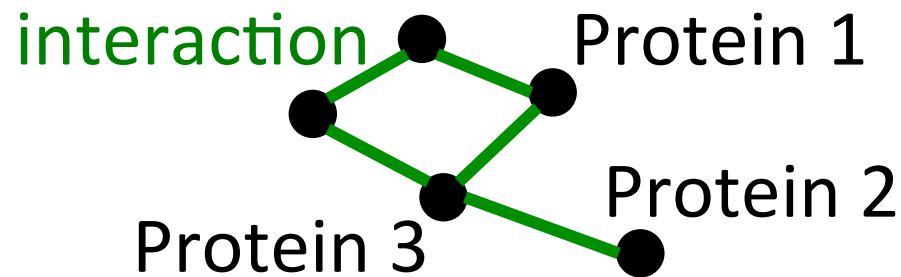


$(i, j) \in \mathcal{E} \Leftrightarrow$ Protein i and j interact transiently
, or in a stable form

Graphs, their comparison, and applications

Unlabeled, undirected, non-weighted

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}) \quad \mathcal{V} = [n]$$

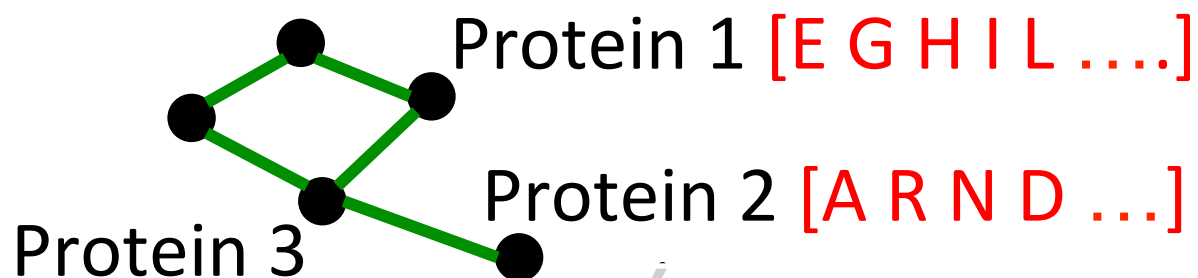


$(i, j) \in \mathcal{E} \Leftrightarrow$ Protein i and j interact transiently
, or in a stable form

Graphs, their comparison, and applications

Labeled, undirected, non-weighted

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$ $\mathcal{V} \subset$ Possible amino acid sequences



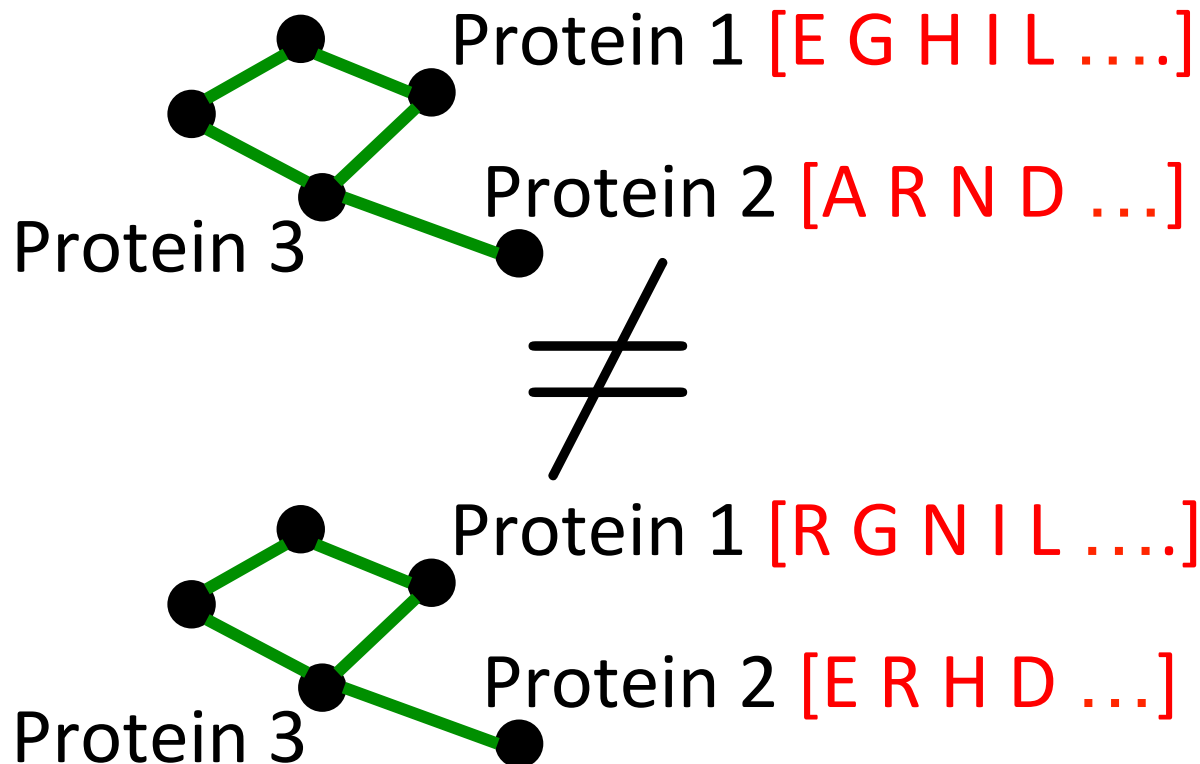
\neq



Graphs, their comparison, and applications

Labeled, undirected, non-weighted

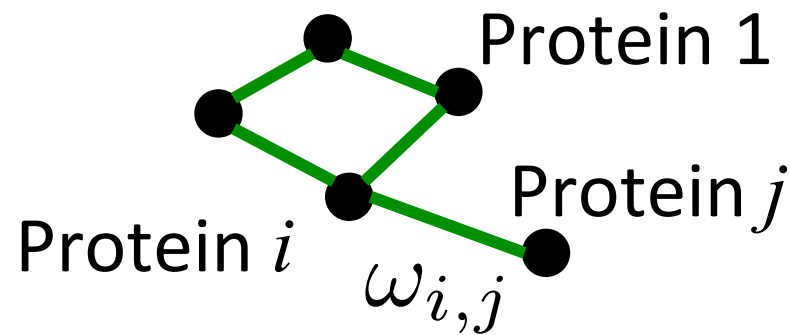
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$ $\mathcal{V} \subset$ Possible amino acid sequences



Graphs, their comparison, and applications

Unlabeled, undirected, weighted

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}) \quad \mathcal{V} = [n]$$

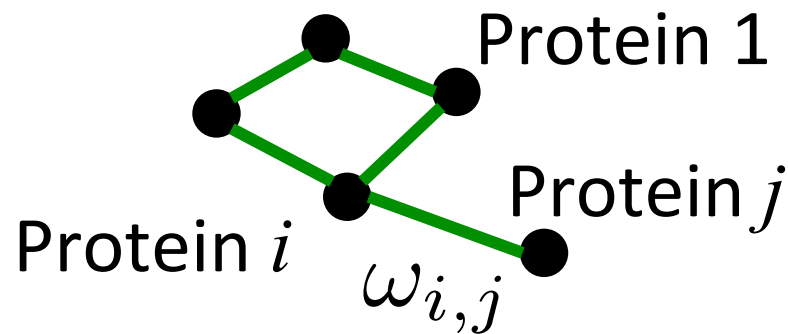


How confident are we that i and j will interact?

Graphs, their comparison, and applications

Unlabeled, undirected, weighted

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}) \quad \mathcal{V} = [n]$$

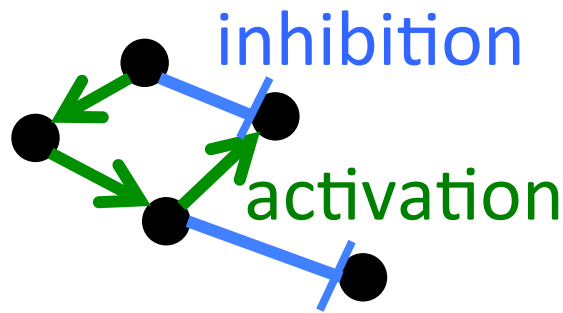


How confident are we that i and j will interact?

Graphs, their comparison, and applications

Unlabeled, directed, weighted

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}) \quad \mathcal{V} = [n]$$

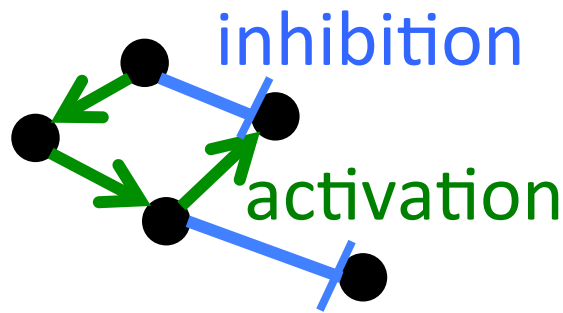


$$(i, j) \in \mathcal{E} \not\Rightarrow (j, i) \in \mathcal{E}$$

Graphs, their comparison, and applications

Unlabeled, directed, weighted

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}) \quad \mathcal{V} = [n]$$



$$(i, j) \in \mathcal{E} \not\Rightarrow (j, i) \in \mathcal{E}$$

Graphs, their comparison, and applications

All of these types of graphs, and other combinations, can be described by an adjacency matrix.

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix}$$

- Non-weighted graphs have 0/1 entries in A
- Undirected graphs have symmetric A
- $a_{i,j} \neq 0$ iff node i and j are not connected

Graphs, their comparison, and applications

All of these types of graphs, and other combinations, can be described by an adjacency matrix.

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix}$$

- Non-weighted graphs have 0/1 entries in A
- Undirected graphs have symmetric A
- $a_{i,j} \neq 0$ iff node i and j are not connected

Talk outline

1. Quick review about graphs
- 2. Applications where comparing graphs is important**
3. Graph distances
4. Why a metric?
5. Scalable alignment algorithms

Graph comparison methods

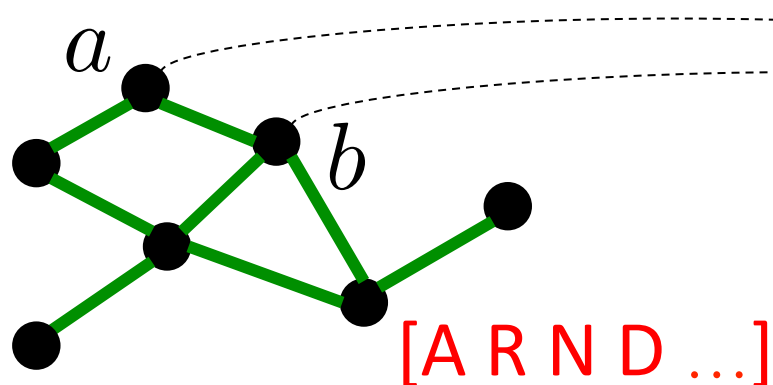
- **Alignment-based**
- **Alignment-free**

(although this distinction might not always be clear)

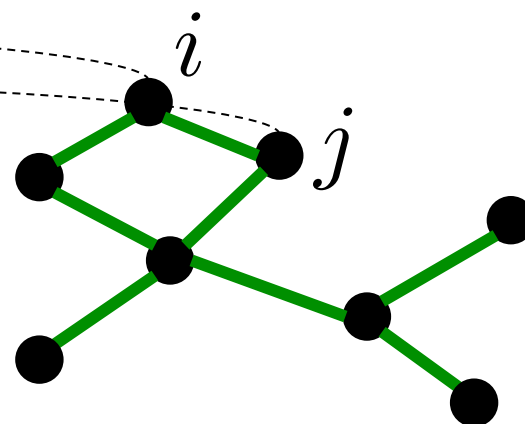
Graphs, their comparison, and applications

Example of alignment-based method in biology: Alignment of protein-protein interaction (PPI) networks

Network 1



Network 2

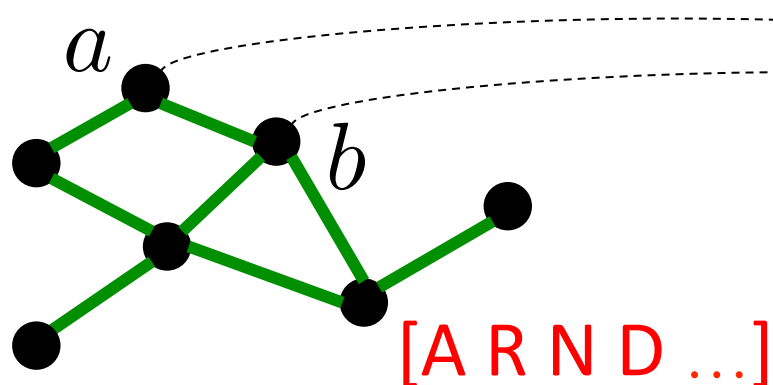


We want to maximize the # of (a, b, i, j) such that (a, i) are evolutionarily related, and (b, j) are too, and the interaction between (a, b) and (i, j) is the same.

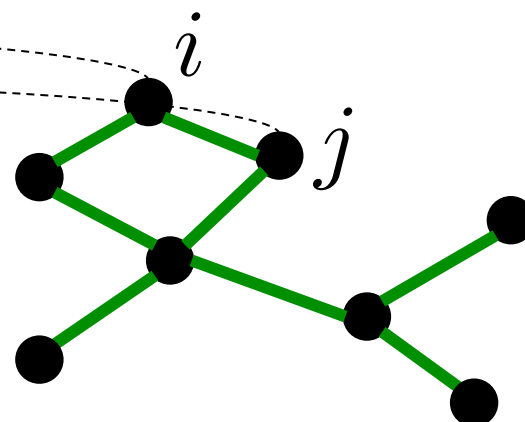
Graphs, their comparison, and applications

Example of alignment-based method in biology:
Alignment of protein-protein interaction (PPI) networks

Network 1



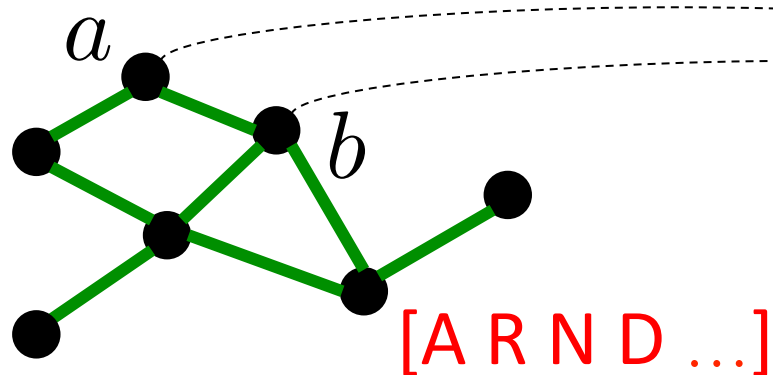
Network 2



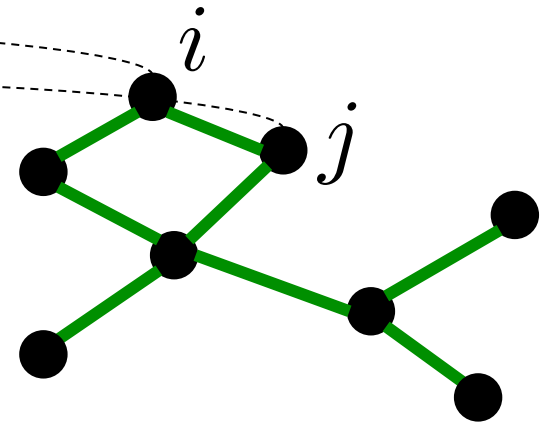
We want to maximize the # of (a, b, i, j) such that (a, i) are evolutionarily related, and (b, j) are too, and the interaction between (a, b) and (i, j) is the same.

Graphs, their comparison, and applications

Network 1



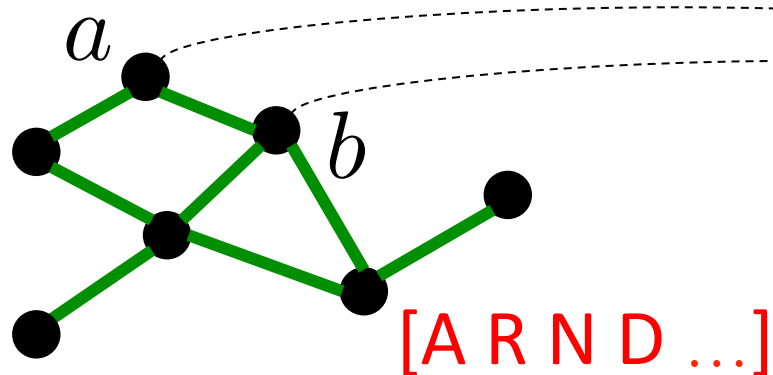
Network 2



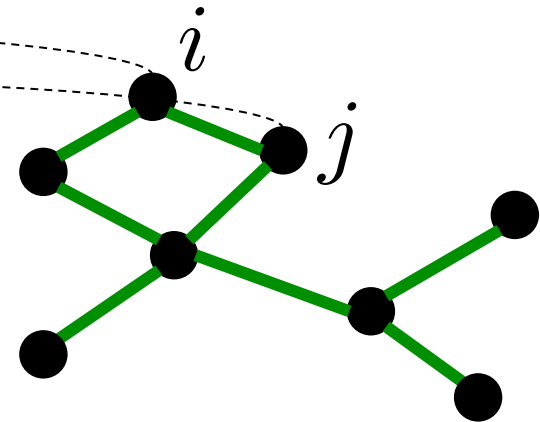
- Do we give more importance to the **labels** (matching nodes with similar sequences) **or** to the **topology**?
- How do we get a **score** from the alignment between the two graphs?

Graphs, their comparison, and applications

Network 1



Network 2

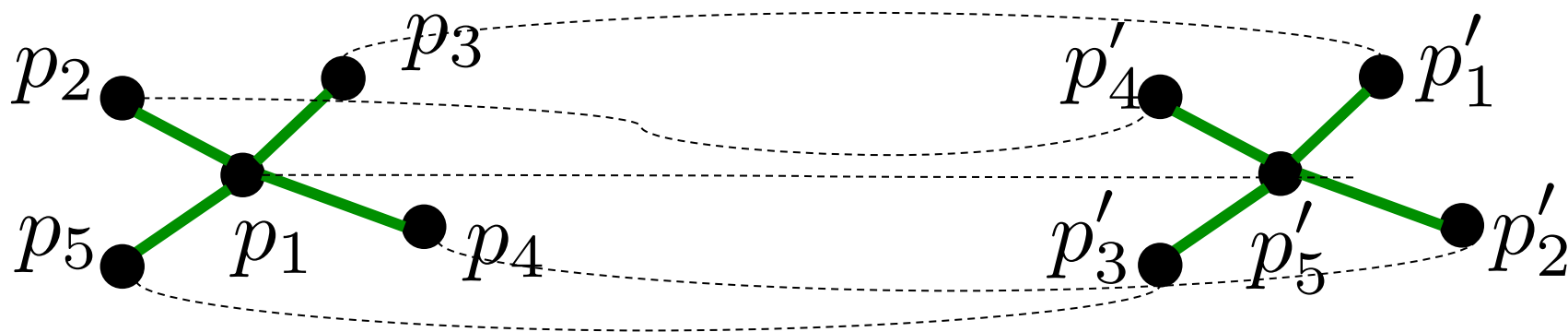


- Do we give more importance to the **labels** (matching nodes with similar sequences) **or** to the **topology**?
- How do we get a **score** from the alignment between the two graphs?

Graphs, their comparison, and applications

The alignment of PPI networks (or networks in general) allows us to

1. **Transfer knowledge** from one graph into another



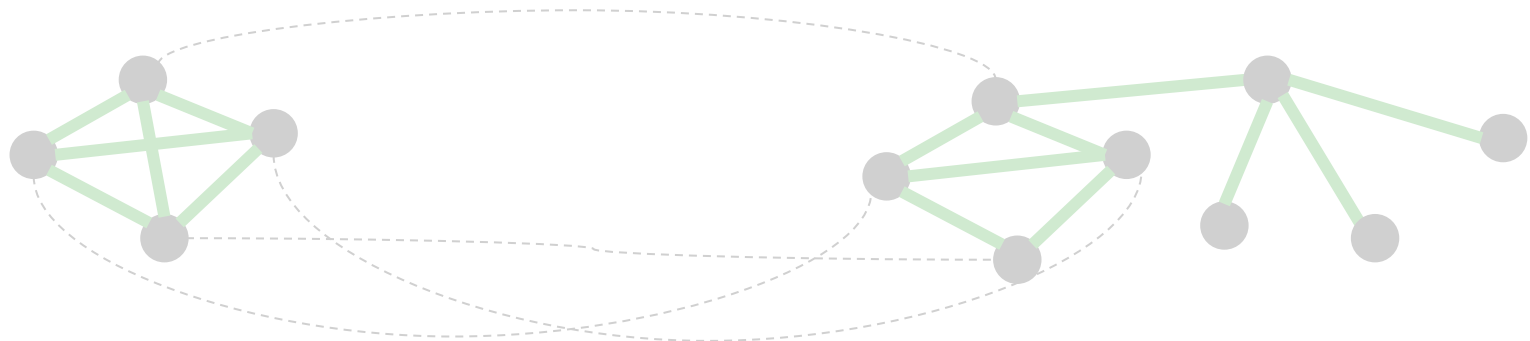
known function for p_1

inferred function for p'_5

Graphs, their comparison, and applications

The alignment of PPI networks (or networks in general) allows us to

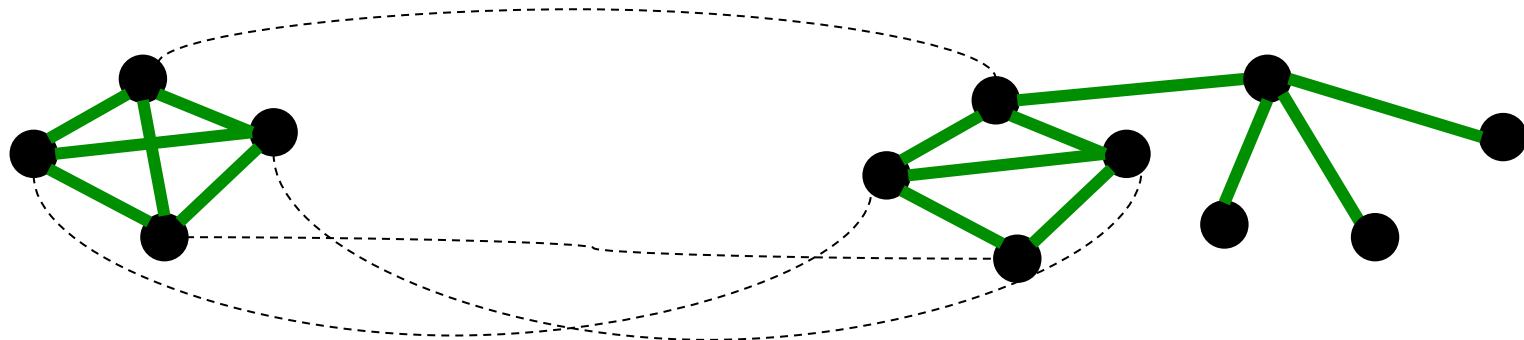
2. Get a **score** of how similar, or dissimilar, two graphs are
3. How well is a **sub network** of proteins found in a large network of proteins?



Graphs, their comparison, and applications

The alignment of PPI networks (or networks in general) allows us to

2. Get a **score** of how similar, or dissimilar, two graphs are
3. How well a **sub network** of proteins can be found in a large network of proteins?



Alignment-free methods facilitate comparing graphs from different domains.

We can compare networks from different domains using, e.g., **degree distribution**, and a method to compare distributions.

For example, several networks from different domains are similar in the sense that they have heavy tail degree distributions: e.g. social networks [e.g. Ahn et al. 2007], WWW [e.g. Crovella et al. 1998], PPI [e.g. Hormozdiari et al. 2007].

Graphs, their comparison, and applications

Alignment-free methods facilitate comparing graphs from different domains.

We can compare networks from different domains using, e.g., **degree distribution**, and a method to compare distributions.

For example, several networks from different domains are similar in the sense that they have heavy tail degree distributions: e.g. social networks [e.g. Ahn et al. 2007], WWW [e.g. Crovella et al. 1998], PPI [e.g. Hormozdiari et al. 2007].

There are many other kinds of networks that, if compared, bring more knowledge than the sum of their parts.

There are many other kinds of networks that, if compared, bring more knowledge than the sum of their parts.

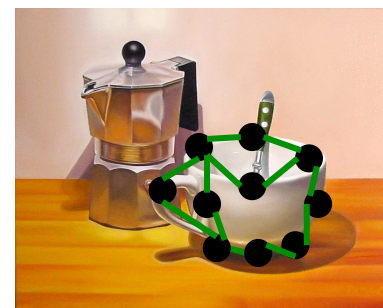
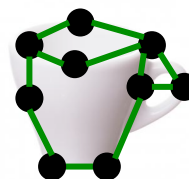
In **biology**

- PPI networks, e.g. [Rohit et al. 2008]
- Gene regulatory networks
- Metabolic networks, e.g. [Pinter et al. 2005]
- Signaling networks
- Neural networks (of the real kind), e.g. [Milano et al. 2017]

Graphs, their comparison, and applications

In **computer vision**, comparing graphs, known usually as **graph matching**, is useful in several tasks.

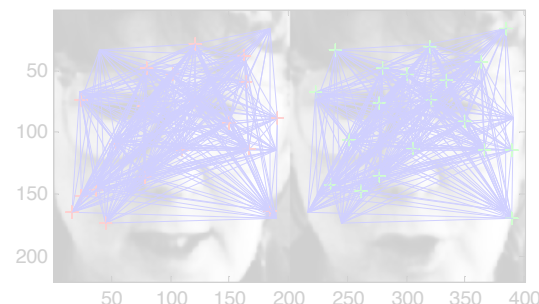
1. Locate objects from features,
e.g. [Gold & Rangarajan 96]



2. Transfer knowledge, e.g.
[Zhang et al. 2010]



3. Find matches in database,
e.g. [Kisku et al. 2007]



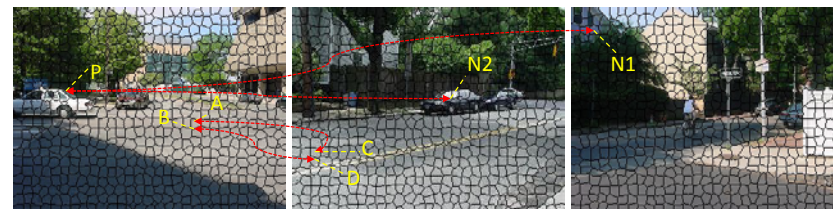
Graphs, their comparison, and applications

In **computer vision**, comparing graphs, known usually as **graph matching**, is useful in several tasks.

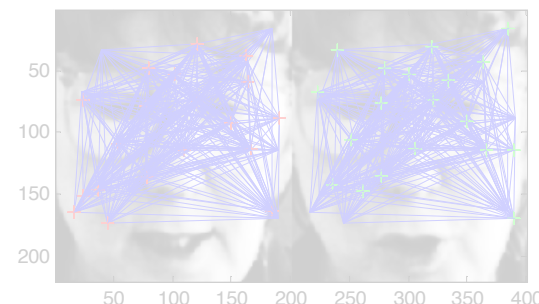
1. **Locate objects** from features,
e.g. [Gold & Rangarajan 96]



2. **Transfer knowledge**, e.g.
[Zhang et al. 2010]



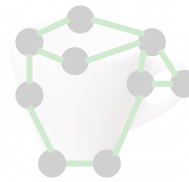
3. **Find matches** in database,
e.g. [Kisku et al. 2007]



Graphs, their comparison, and applications

In **computer vision**, comparing graphs, known usually as **graph matching**, is useful in several tasks.

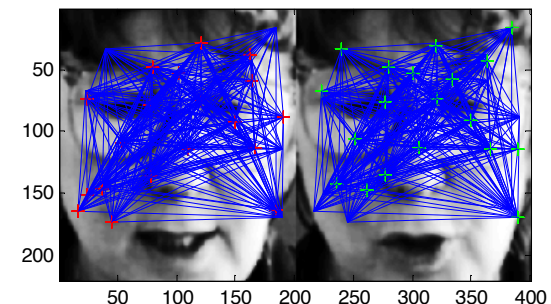
1. **Locate objects** from features,
e.g. [Gold & Rangarajan 96]



2. **Transfer knowledge**, e.g.
[Zhang et al. 2010]



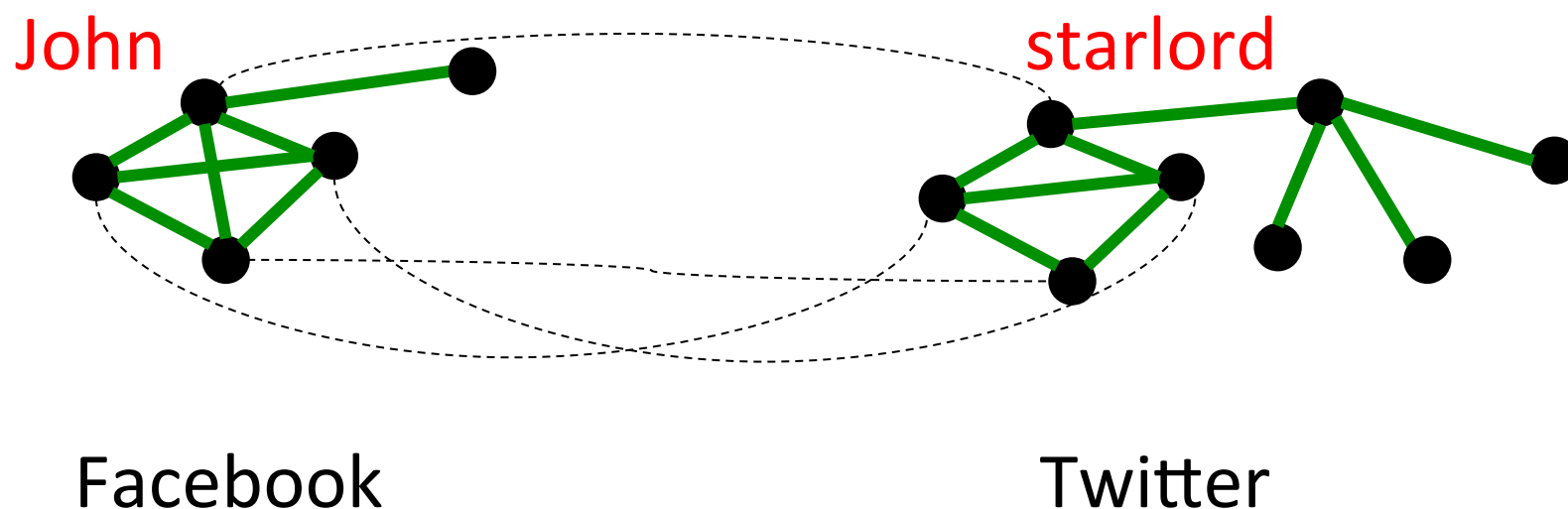
3. **Find matches in database**,
e.g. [Kisku et al. 2007]



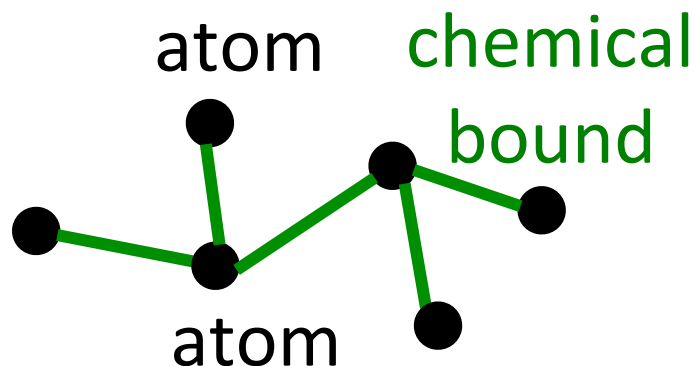
Graphs, their comparison, and applications

Comparing **social networks** is very important.

E.g. it allows us to **uncover identities**, or communities
[Kong et al. 2013]



Graphs, their comparison, and applications



In **chemistry**, comparing graphs is important to answer the following questions [Akutsu et al. 2013]:

- Are two chemicals identical?
- Is one compound part of another compound?
- What is the maximum common part of two chemicals?

Graphs, their comparison, and applications

In general, most of these tasks are hard to complete exactly/optimally. However, domain knowledge can make tasks tractable.

For example, in chemistry, most graphs have maximum degree 8, which simplifies many of these tasks.

Alignment-based methods: usually chosen for tasks such as

- Transfer knowledge
- Sub graph matching

Alignment-free methods: more flexible in terms of what kinds of graphs can be compared

Is there a family of methods that encompass both alignment-free and alignment-based methods?

(stay tuned for **Part II**)

Is there a family of methods that encompass both alignment-free and alignment-based methods?

(stay tuned for **Part II**)

Talk outline

1. Quick review about graphs
2. Applications where comparing graphs is important
- 3. Graph distances**
4. Why a metric?
5. Scalable alignment algorithms

Exact graph comparisons (output a “yes” or “no”)

- **Graph isomorphism:** given two graphs

$$\mathcal{G}_1 = ([n], \mathcal{E}_1), \mathcal{G}_2 = ([n], \mathcal{E}_2)$$

is there a permutation $P : [n] \mapsto [n]$ such that

$$(i, j) \in \mathcal{E}_1 \text{ iff } (P(i), P(j)) \in \mathcal{E}_2$$

(this problem belongs to NP but it is not known if it belongs to P or NP-complete)

Exact graph comparisons (output a “yes” or “no”)

- **Sub graph isomorphism:** given two graphs

$$\mathcal{G}_1 = ([n], \mathcal{E}_1), \mathcal{G}_2 = ([m], \mathcal{E}_2), n \geq m$$

is there a sub-graph of \mathcal{G}_1 , i.e. $\mathcal{G}_0 = (\mathcal{V}_0, \mathcal{E}_0)$ s.t.
 $\mathcal{V}_0 \subseteq [n], \mathcal{E}_0 \subseteq \mathcal{E}_1 \cup \mathcal{V}_0 \times \mathcal{V}_0$, s.t. $\mathcal{G}_0 \cong \mathcal{G}_2$?

Inexact graph comparison (output a closeness score):

- Matching: 1-to-1; 1-to-many; many-to-many;
- Score: edge overlap; spectrum overlap; etc.

- **Chemical distance** (alignment-based)

Let A_1 and A_2 be the adjacency matrices of two graphs of equal size.

$$\min_{P \in \Pi} \|A_1 P - P A_2\|_F = \min_{P \in \Pi} |P(\mathcal{E}_1) \Delta \mathcal{E}_2|$$

- **Chartrand-Kubicki-Schultz dist.** (alignment-based)

Same as chemical distance but with A_1 and A_2 equal to the hop distance between each two nodes in each of the two graphs.

Inexact graph comparisons

- **Chemical distance** (alignment-based)

Let A_1 and A_2 be the adjacency matrices of two graphs of equal size.

$$\min_{P \in \Pi} \|A_1 P - P A_2\|_F = \min_{P \in \Pi} |P(\mathcal{E}_1) \Delta \mathcal{E}_2|$$

- **Chartrand-Kubicki-Schultz dist.** (alignment-based)

Same as chemical distance but with A_1 and A_2 containing the hop distances between each two nodes in each of the two graphs.

Inexact graph comparisons

- **Edit distance** (alignment-free)

Given two graphs, not necessarily of equal size, and a set of operations, e.g.

$$\mathcal{O} = \{\text{vertex/edge/label} \\ \text{insertion/deletion/substitution}\}$$

, and a cost function $c : \mathcal{O} \mapsto \mathbb{R}$, we want to find the cheapest sequence of operations that take \mathcal{G}_1 into \mathcal{G}_2 .

$$\min_{\{e_i\}_{i=1}^k \in \mathcal{O}^k : \mathcal{G}_2 = (e_k \circ \dots \circ e_1) \circ \mathcal{G}_1} \sum_{i=1}^k c(e_k)$$

For certain c and \mathcal{O} this reduces to the Chemical dist.

Inexact graph comparisons

- **Spectral distance** (alignment-free)

Given two graphs of equal size with adjacency matrices A_1 and A_2 , let $\mu_1 \geq \cdots \geq \mu_n$ and $\nu_1 \geq \cdots \geq \nu_n$ be the spectrum of A_1 and A_2 .

$$\sum_{i=1}^n |\mu_i - \nu_i|$$

- Zelinks distance, **common sub graph distance** (alignment-based) is: $\max\{|\mathcal{V}_1|, |\mathcal{V}_2|\} - n(\mathcal{G}_1, \mathcal{G}_2)$
 $n(\mathcal{G}_1, \mathcal{G}_2) = \max |S_1|$ s.t. $S_1 \subseteq \mathcal{V}_1, S_2 \subseteq \mathcal{V}_2,$
 $|S_1| = |S_2|, \mathcal{G}_1(S_1) \cong \mathcal{G}_2(S_2)$

Inexact graph comparisons

- **Spectral distance** (alignment-free)

Given two graphs of equal size with adjacency matrices A_1 and A_2 , let $\mu_1 \geq \cdots \geq \mu_n$ and $\nu_1 \geq \cdots \geq \nu_n$ be the spectrum of A_1 and A_2 . The spectral distance is

$$\sum_{i=1}^n |\mu_i - \nu_i|$$

- **Zelinks distance, common sub graph distance** (alignment-based): $\max\{|\mathcal{V}_1|, |\mathcal{V}_2|\} - n(\mathcal{G}_1, \mathcal{G}_2)$

$$n(\mathcal{G}_1, \mathcal{G}_2) = \max |S_1| \text{ s.t. } S_1 \subseteq \mathcal{V}_1, S_2 \subseteq \mathcal{V}_2, \\ |S_1| = |S_2|, \mathcal{G}_1(S_1) \cong \mathcal{G}_2(S_2)$$

- **Bunke-Shearer metric** (alignment-based)

Using the same setup as the common sub graph distance, this is

$$1 - \frac{n(\mathcal{G}_1, \mathcal{G}_2)}{\max\{|\mathcal{V}_1|, |\mathcal{V}_2|\}}$$

- **Common super graph distance** (alignment-based)

$$N(\mathcal{G}_1, \mathcal{G}_2) - \min\{|\mathcal{V}_1|, |\mathcal{V}_2|\}$$

where

$$N(\mathcal{G}_1, \mathcal{G}_2) = \min |\mathcal{V}| \text{ s.t. } \mathcal{G} = (\mathcal{V}, \mathcal{E}), S_1, S_2 \subseteq \mathcal{V}, \\ \mathcal{G}(S_1) \cong \mathcal{G}_1, \mathcal{G}(S_2) \cong \mathcal{G}_2$$

Inexact graph comparisons

- **Bunke-Shearer metric** (alignment-based)

using the same setup as the common sub graph distance, this is

$$1 - \frac{n(\mathcal{G}_1, \mathcal{G}_2)}{\max\{|\mathcal{V}_1|, |\mathcal{V}_2|\}}$$

- **Common super graph distance** (alignment-based)

$$N(\mathcal{G}_1, \mathcal{G}_2) - \min\{|\mathcal{V}_1|, |\mathcal{V}_2|\}$$

where

$$N(\mathcal{G}_1, \mathcal{G}_2) = \min |\mathcal{V}| \text{ s.t. } \mathcal{G} = (\mathcal{V}, \mathcal{E}), S_1, S_2 \subseteq \mathcal{V}, \\ \mathcal{G}(S_1) \cong \mathcal{G}_1, \mathcal{G}(S_2) \cong \mathcal{G}_2$$

- **Edge distance**

$$|\mathcal{E}_1| + |\mathcal{E}_2| - 2E(\mathcal{G}_1, \mathcal{G}_2) + ||\mathcal{V}_1| - |\mathcal{V}_2||$$

$$E(\mathcal{G}_1, \mathcal{G}_2) = \max |\mathcal{E}| \text{ s.t. } \mathcal{G} = (\mathcal{V}, \mathcal{E}), S_1 \subseteq \mathcal{V}_1, S_2 \subseteq \mathcal{V}_2, \\ |S_1| = |S_2|, \mathcal{G} \cong \mathcal{G}_1(S_1) \cong \mathcal{G}_2(S_2)$$

- **Fernández-Valiente metric**

If n and m are the sizes of the maximum common sub graph and minimum common super graph, then $m - n$ is a metric.

Inexact graph comparisons

- **Edge distance**

$$|\mathcal{E}_1| + |\mathcal{E}_2| - 2E(\mathcal{G}_1, \mathcal{G}_2) + ||\mathcal{V}_1| - |\mathcal{V}_2||$$

$$E(\mathcal{G}_1, \mathcal{G}_2) = \max |\mathcal{E}| \text{ s.t. } \mathcal{G} = (\mathcal{V}, \mathcal{E}), S_1 \subseteq \mathcal{V}_1, S_2 \subseteq \mathcal{V}_2, \\ |S_1| = |S_2|, \mathcal{G} \cong \mathcal{G}_1(S_1) \cong \mathcal{G}_2(S_2)$$

- **Fernández-Valiente metric**

If n and m are the sizes of the maximum common sub graph and minimum common super graph, then $m - n$ is a metric.

Talk outline

1. Quick review about graphs
2. Applications where comparing graphs is important
3. Graph distances
- 4. Why a metric?**
5. Scalable alignment algorithms

All of the notions of comparing graphs just mentioned are metrics (if we consider an appropriate quotient space, where the equivalence classes are the graphs for which the respective distance is zero).

Metric: $d : \Omega \times \Omega \mapsto \mathbb{R}$ such that $\forall A, B, C \in \Omega$

$d(A, B) \geq 0$ (non-negativity)

$d(A, B) = 0$ iff $A = B$ (identity of indiscernibles)

$d(A, B) = d(B, A)$ (symmetry)

$d(A, B) + d(B, C) \geq d(A, C)$ (triangle inequality)

The pair (Ω, d) is called a **metric space**.

All of the notions of comparing graphs just mentioned are metrics (if we consider an appropriate quotient space, where the equivalence classes are the graphs for which the respective distance is zero).

Metric: $d : \Omega \times \Omega \mapsto \mathbb{R}$ such that $\forall A, B, C \in \Omega$

$d(A, B) \geq 0$ (non-negativity)

$d(A, B) = 0$ iff $A = B$ (identity of indiscernibles)

$d(A, B) = d(B, A)$ (symmetry)

$d(A, B) + d(B, C) \geq d(A, C)$ (triangle inequality)

The pair (Ω, d) is called a **metric space**.

All of the notions of comparing graphs just mentioned are metrics (if we consider an appropriate quotient space, where the equivalence classes are the graphs for which the respective distance is zero).

Metric: $d : \Omega \times \Omega \mapsto \mathbb{R}$ such that $\forall A, B, C \in \Omega$

$d(A, B) \geq 0$ (non-negativity)

$d(A, B) = 0$ iff $A = B$ (identity of indiscernibles)

$d(A, B) = d(B, A)$ (symmetry)

$d(A, B) + d(B, C) \geq d(A, C)$ (triangle inequality)

The pair (Ω, d) is called a **metric space**.

Pseudo-metrics and quasi-metrics

A **pseudo-metric** relaxes the condition

$d(A, B) = 0$ iff $A = B$ (identity of indiscernibles)
to $d(A, A) = 0$

Given a pseudo-metric, we can obtain a metric if we define the equivalence relation $A \sim B$ iff $d(A, B) = 0$ and define the metric $\tilde{d} : (\Omega \setminus \sim) \times (\Omega \setminus \sim) \mapsto \mathbb{R}$ such that $\tilde{d}([A], [B]) = d(A, B)$.

A **quasi-metric** removes the symmetry condition from the definition of metrics. From a quasi-metric d we can obtain a metric as $\tilde{d}(A, B) = d(A, B) + d(B, A)$

Pseudo-metrics and quasi-metrics

A **pseudo-metric** relaxes the condition $d(A, B) = 0$ iff $A = B$ (identity of indiscernibles) to $d(A, A) = 0$

Given a pseudo-metric, we can obtain a metric if we define the equivalence relation $A \sim B$ iff $d(A, B) = 0$ and define the metric $\tilde{d} : (\Omega \setminus \sim) \times (\Omega \setminus \sim) \mapsto \mathbb{R}$ such that $\tilde{d}([A], [B]) = d(A, B)$.

A **quasi-metric** removes the symmetry condition from the definition of metrics. From a quasi-metric d we can obtain a metric as $\tilde{d}(A, B) = d(A, B) + d(B, A)$

Pseudo-metrics and quasi-metrics

A **pseudo-metric** relaxes the condition

$d(A, B) = 0$ iff $A = B$ (identity of indiscernibles)
to $d(A, A) = 0$

Given a pseudo-metric, we can obtain a metric if we
define the equivalence relation $A \sim B$ iff $d(A, B) = 0$
and define the metric $\tilde{d} : (\Omega \setminus \sim) \times (\Omega \setminus \sim) \mapsto \mathbb{R}$
such that $\tilde{d}([A], [B]) = d(A, B)$.

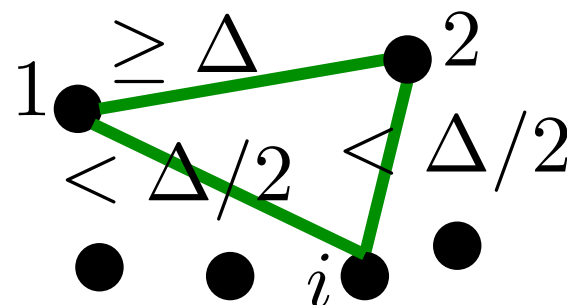
A **quasi-metric** removes the symmetry condition from
the definition of metrics. From a quasi-metric d we
can obtain a metric as $\tilde{d}(A, B) = d(A, B) + d(B, A)$

Why a metric?

Metrics allow fast algorithms for several tasks

- **Diameter estimation:** Given a set S with n elements, we want to find $\max_{x,y \in S} d(x,y)$

[Indyk 1999] 1/2-approximation algorithm with $\mathcal{O}(n)$ expected run time. **Idea:** if there exists two points with $\text{dist.} \geq \Delta$ then there are at least $n - 1$ pairs of points with $\text{dist.} \geq \Delta / 2$.



Hence, sampling $\mathcal{O}(n)$ pairs is enough on average.

Why a metric?

Metrics allow fast algorithms for several tasks

- **Nearest neighbor search:** For a set S with n elements and query point q drawn randomly from V in \mathbb{R}^k [Kenneth & Clarkson 1999] propose a data structure with the following expected run times
 - Preprocessing : $\mathcal{O}(n)(\log n)^{\mathcal{O}(\log \log \gamma)}$ where γ is the ratio of the maximum to minimum distance between points
 - Query: $\mathcal{O}((\log n)^{\mathcal{O}(1)+2 \log \log \gamma})$
 - Space: $\mathcal{O}(n(\log n)^{\mathcal{O}(1)+2 \log \log \gamma})$

Why a metric?

Metrics allow fast algorithms for several tasks

- Fast NN algorithms can be used to derive fast **outlier detection algorithms**. E.g. [Angiulli & Pizzuti 2002] do this with the following definition of outlier p

$$\text{weight}_k(p) = \sum_{s \in k\text{-NN}(p)} d(s, p)$$

outlier = large w_k

Why a metric?

Metrics allow fast algorithms for several tasks

- **Clustering** (k -means clustering): Given a set P of size n , find

$$\min_{C \subseteq P: |C|=k} \left\{ \sum_{p \in P} \min_{c \in C} d(p, c) \right\}$$

In arbitrary metric spaces this problem is NP-hard. Approximation algorithms are possible. E.g. in \mathbb{R}^d [Badiou et al. 2002/2003] get an $(1 + \varepsilon)$ -approx. in time

$$\mathcal{O}(d^{\mathcal{O}(1)} n (\log^{\mathcal{O}(k)} n) 2^{(\frac{k}{\varepsilon})^{\mathcal{O}(1)}})$$

Why a metric?

Metrics allow fast algorithms for several tasks

- **Clustering** (k -means clustering): Given a set P of size n , find

$$\min_{C \subseteq P: |C|=k} \left\{ \sum_{p \in P} \min_{c \in C} d(p, c) \right\}$$

Arbitrary metrics: [Ackermann, Blomer & Sohler 2008]: If the 1-median can be solved in linear time then there exists an $(1 + \varepsilon)$ -approximation algorithm with run time

$$\mathcal{O}\left(n 2^{\left(\frac{k}{\varepsilon}\right)^{\mathcal{O}(1)}}\right)$$

Why a metric?

Metrics allow fast algorithms for several tasks

- **Clustering** (k -means clustering): Given a set P of size n , find

$$\min_{C \subseteq P: |C|=k} \left\{ \sum_{p \in P} \min_{c \in C} d(p, c) \right\}$$

Arbitrary dissimilarity: There are a few results for the Kullback Leibler divergence and Bergman divergence. But, in general, results are rare for non-metrics.

Talk outline

1. Quick review about graphs
2. Applications where comparing graphs is important
3. Graph distances
4. Why a metric?
5. Scalable alignment algorithms

Many of the metrics just mentioned cannot be computed easily. We now go over a few efficient algorithms that try to find globally-optimal alignment between two graphs.

Any alignment algorithms can be transformed into distance functions. For example, given an alignment P between two graphs, with adjacency matrices A_1, A_2 we can compute $\|A_1 P - P A_2\|_F$.

These distances often do not result in metrics.
(Stay tuned for **Part II**)

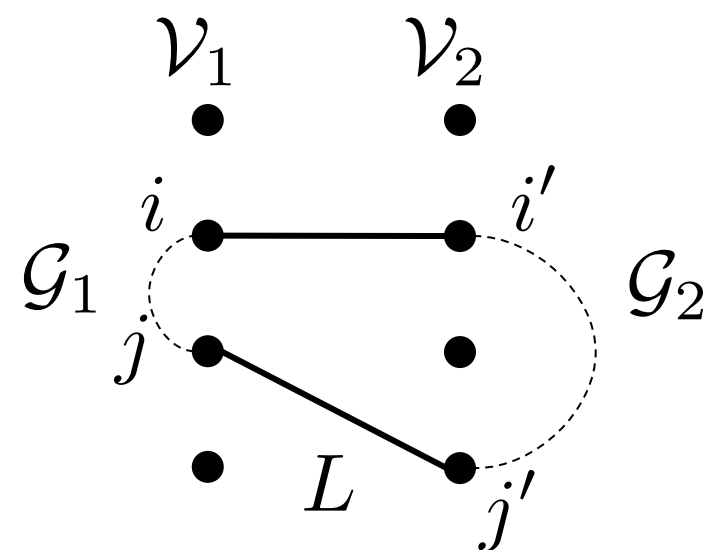
Scalable algorithms for graph alignment

Many of the metrics just mentioned cannot be computed easily. We now go over a few efficient algorithms that try to find globally-optimal alignment between two graphs.

Any alignment algorithms can be transformed into distance functions. For example, given an alignment P between two graphs, with adjacency matrices A_1, A_2 we can compute $\|A_1 P - P A_2\|_F$.

These distances often do not result in metrics.
(Stay tuned for **Part II**)

Quadratic formulation



$L =$ possible node matches

$$S_{i,i',j,j'} = 1$$

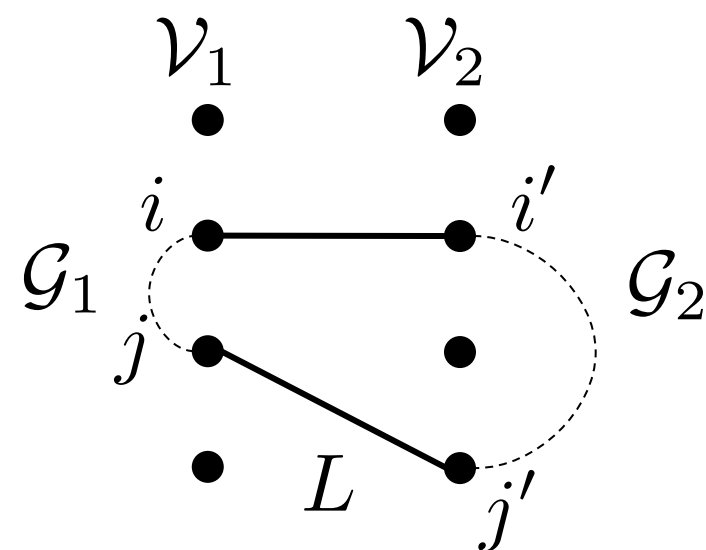
iff i, i', j, j' make a square
through L, \mathcal{G}_1 and \mathcal{G}_2

$$\max \left\{ \alpha \sum_{(i,i') \in L} w_{i,i'} x_{i,i'} + \sum_{(i,i'), (j,j') \in L} x_{i,i'} x_{j,j'} S_{i,i',j,j'} \right\}$$

$$\text{subject to } \sum_{i': (i,i') \in L} x_{i,i'} \leq 1, \forall i; \quad \sum_{i: (i,i') \in L} x_{i,i'} \leq 1, \forall i';$$

$$x_{i,i'} \in \{0, 1\}, \forall (i, i') \in L$$

Quadratic formulation



$L =$ possible node matches

$$S_{i,i',j,j'} = 1$$

iff i, i', j, j' make a square
through L, \mathcal{G}_1 and \mathcal{G}_2

If L is the complete graph, and $w = 0$, this reduced to maximum common sub graph distance, which is NP-hard, even NP-hard to approximate.

Several relaxations of this IQP lead to scalable algorithms for matching to graphs.

- Linear relaxation: Replace

$$(\dots) + \sum_{(i,i'),(j,j') \in L} x_{i,i'} x_{j,j'} S_{i,i',j,j'}$$

$$\text{s.t. } (\dots) \text{ and } x_{i,i'} \in \{0, 1\}, \forall (i, i') \in L$$

$$\text{by } (\dots) + \sum_{(i,i'),(j,j') \in L} y_{i,i',j,j'} S_{i,i',j,j'} \quad \text{s.t. } (\dots) \text{ and}$$

$$y_{i,i',j,j'} = y_{j,j',i,i'}, y_{i,i',j,j'} \leq x_{i,i'}, x_{i,i'} \in [0, 1], \forall (i, i'), (j, j') \in L$$

This LP relaxation requires a final rounding scheme.

- [Klau 2009] First, move the symmetry constraint in the linear relaxation into the objective as

$$(\dots) + \sum_{(i,i'),(j,j') \in L} U_{i,i',j,j'} (y_{i,i',j,j'} - y_{j,j',i,i'})$$

and add the constraint that

$$\sum_{j:(j,j') \in L} y_{i,i',j,j'} \leq 1 \forall j', \quad \sum_{j':(j,j') \in L} y_{i,i',j,j'} \leq 1, \forall j, \forall (i,i') \in L$$

For any U , this problem upper bounds the linear relaxation. The resulting LP has the form of a maximum weight matching, hence produces 0/1 sols.

Second, use sub gradient descent to optimize over U . **This algorithm does not require a final rounding scheme.**

Natalie 2.0 [El-Kebir 2015] is an improvement on Klau's algorithm, where the sub gradient descent is combined with a dual descent step.

Quadratic formulation: Klau's algorithm & Natalie 2.0

Second, use sub gradient descent to optimize over U . **This algorithm does not require a final rounding scheme.**

Natalie 2.0 [El-Kebir 2015] is an improvement on Klau's algorithm, where the sub gradient descent is combined with a dual descent step.

Write the quadratic problem ($w = 0$ for simplicity) as

$$\max \sum_{i,i',j,j' \in \square(L)} x_{i,i',j,j'} \quad \square(L) = \text{potential square in } L$$

$$\text{subject to } \sum_{i':(i,i') \in L} x_{i,i'} \leq 1, \forall i; \quad \sum_{i:(i,i') \in L} x_{i,i'} \leq 1, \forall i';$$

$$x_{i,i',j,j'} = x_{i,i'} x_{j,j'} \forall i, i', j, j' \in \square(L),$$

$$x_{i,i'} \in \{0, 1\}, \forall (i, i') \in L$$

Build the probability distribution

$$\mathbb{P}(\{x_{i,i'}\}, \{x_{i,i',j,j'}\}) = \frac{1}{Z} \left(e^{\frac{\beta}{2} \sum_{i,i',j,j' \in \square(L)} x_{i,i',j,j'}} \right) \prod_{i'} \Psi \left(\sum_{i:(i,i') \in L} x_{i,i'} \leq 1 \right) \\ \prod_i \Psi \left(\sum_{i':(i,i') \in L} x_{i,i'} \leq 1 \right) \prod_{i,i',j,j' \in \square(L)} \Psi(x_{i,i',j,j'} = x_{i,i'} x_{j,j'})$$

Write the quadratic problem ($w = 0$ for simplicity) as

$$\max \sum_{i,i',j,j' \in \square(L)} x_{i,i',j,j'} \quad \square(L) = \text{potential square in } L$$

$$\text{subject to } \sum_{i':(i,i') \in L} x_{i,i'} \leq 1, \forall i; \quad \sum_{i:(i,i') \in L} x_{i,i'} \leq 1, \forall i';$$

$$x_{i,i',j,j'} = x_{i,i'} x_{j,j'} \forall i, i', j, j' \in \square(L),$$

$$x_{i,i'} \in \{0, 1\}, \forall (i, i') \in L$$

Build the probability distribution

$$\mathbb{P}(\{x_{i,i'}\}, \{x_{i,i',j,j'}\}) = \frac{1}{Z} \left(e^{\frac{\beta}{2} \sum_{i,i',j,j' \in \square(L)} x_{i,i',j,j'}} \right) \prod_{i'} \Psi \left(\sum_{i:(i,i') \in L} x_{i,i'} \leq 1 \right) \\ \prod_i \Psi \left(\sum_{i':(i,i') \in L} x_{i,i'} \leq 1 \right) \prod_{i,i',j,j' \in \square(L)} \Psi(x_{i,i',j,j'} = x_{i,i'} x_{j,j'})$$

The support of $\mathbb{P}(\{x_{i,i'}\}, \{x_{i,i',j,j'}\})$ is the set of feasible solution to the original IQP.

We can use max-product BP, a message-passing algorithm over the factor-graph associated to this distribution, to find

$$\arg \max \mathbb{P}(x_{i,i'}) \text{ or } \arg \max \mathbb{P}(x_{i,i',j,j'})$$

With a rounding scheme, we can then extract an approximate solution to the original problem from the maximizers of the marginal probabilities.

The support of $\mathbb{P}(\{x_{i,i'}\}, \{x_{i,i',j,j'}\})$ is the set of feasible solution to the original IQP.

We can use max-product BP, a message-passing algorithm over the factor-graph associated to this distribution, to find

$$\arg \max \mathbb{P}(x_{i,i'}) \text{ or } \arg \max \mathbb{P}(x_{i,i',j,j'})$$

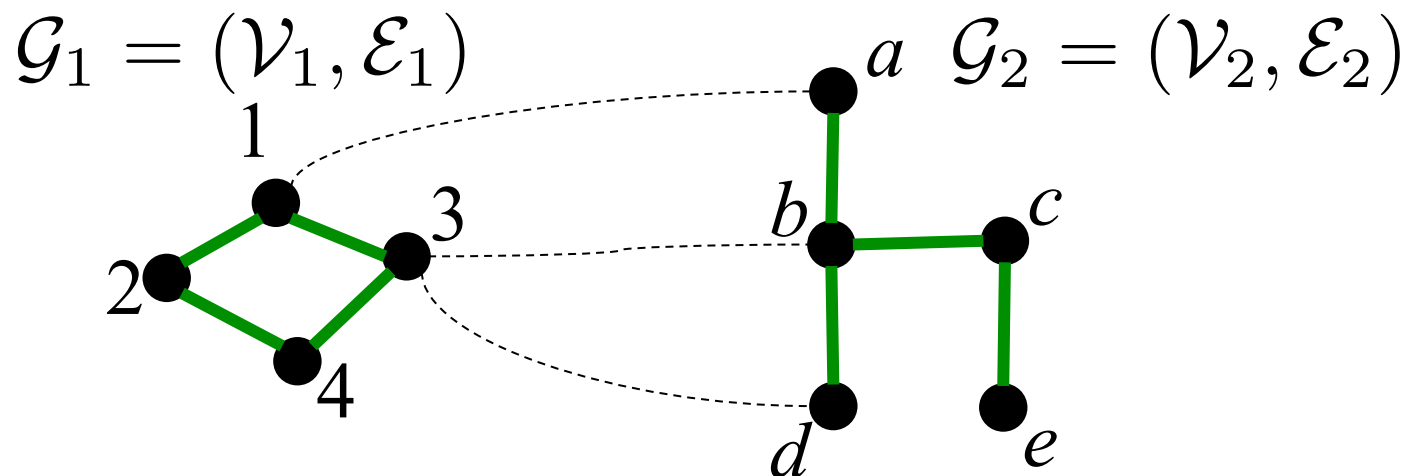
With a rounding scheme, we can then extract an approximate solution to the original problem from the maximizers of the marginal probabilities.

The support of $\mathbb{P}(\{x_{i,i'}\}, \{x_{i,i',j,j'}\})$ is the set of feasible solution to the original IQP.

We can use max-product BP, a message-passing algorithm over the factor-graph associated to this distribution, to find

$$\arg \max \mathbb{P}(x_{i,i'}) \text{ or } \arg \max \mathbb{P}(x_{i,i',j,j'})$$

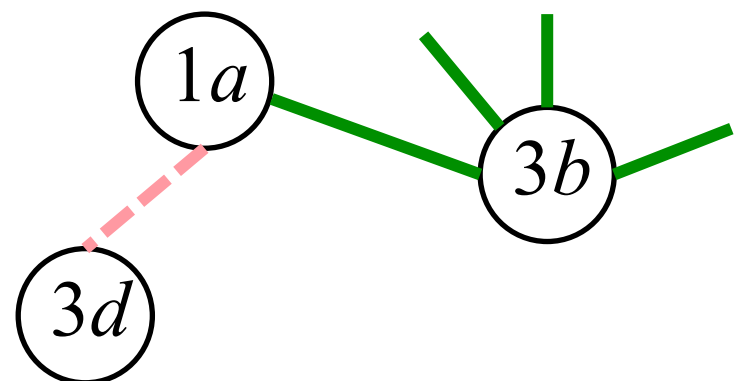
With a rounding scheme, we can then extract an approximate solution to the original problem from the maximizers of the marginal probabilities.



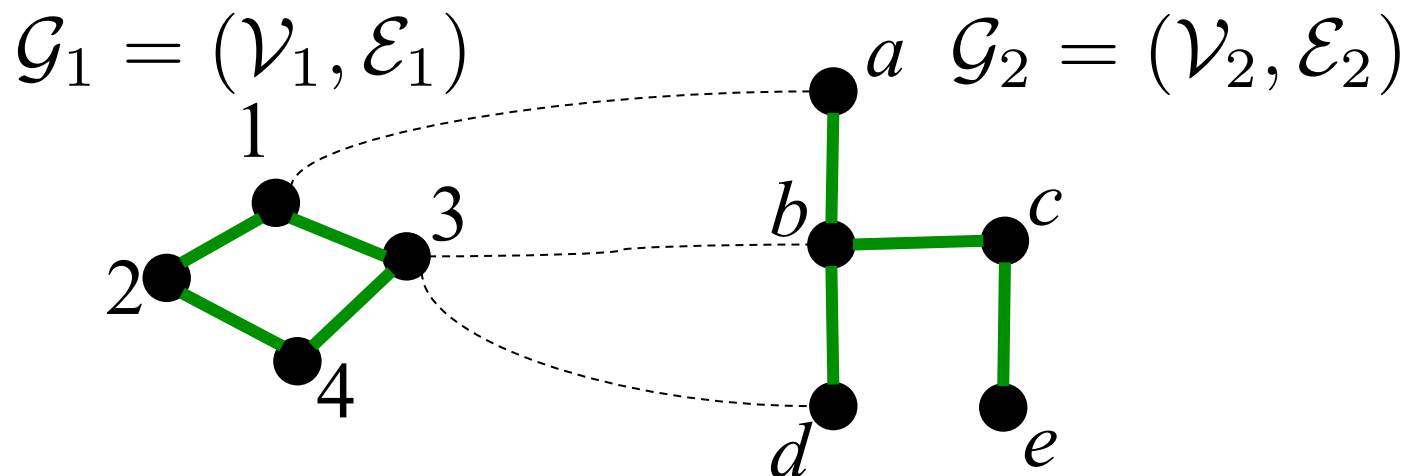
Let A be the adjacency matrix of the product graph

$$\mathcal{G}_1 \times \mathcal{G}_2 = (\mathcal{V}_1 \times \mathcal{V}_2, \{((i, j), (u, v)) : (i, u) \in \mathcal{E}_1, (j, v) \in \mathcal{E}_2\})$$

$$\deg(3b) = \deg(3) \times \deg(b)$$



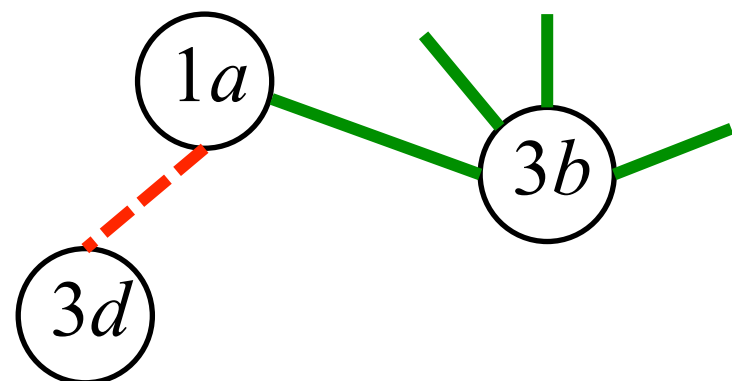
Isorank: (1) Find $R: R = A R$ and (2) then use $R \in \mathbb{R}^{|\mathcal{V}_1| \times |\mathcal{V}_2|}$ as weights in a maximum matching problem to match nodes of \mathcal{G}_1 and \mathcal{G}_2 .



Let A be the adjacency matrix of the product graph

$$\mathcal{G}_1 \times \mathcal{G}_2 = (\mathcal{V}_1 \times \mathcal{V}_2, \{((i, j), (u, v)) : (i, u) \in \mathcal{E}_1, (j, v) \in \mathcal{E}_2\})$$

$$\deg(3b) = \deg(3) \times \deg(b)$$



Isorank: (1) Find R : $R = A R$ and (2) then use $R \in \mathbb{R}^{|\mathcal{V}_1| \times |\mathcal{V}_2|}$ as weights in a maximum matching problem to match nodes of \mathcal{G}_1 and \mathcal{G}_2 .

Note that $R = A R$ corresponds to finding the stationary distribution of a random walk on $\mathcal{G}_1 \times \mathcal{G}_2$:

This stationary distribution is $R_{i,j} = \frac{\deg(i)\deg(j)}{2|\mathcal{E}_1||\mathcal{E}_2|}$

The equation $R = A R$ can be generalized to include node-similarity information as $R = \alpha A R + (1 - \alpha)E$, where E is normalized to sum to 1 and α is in $[0, 1]$.

We can solve $R = \alpha A R + (1 - \alpha)E$ using a PageRank-like method. A match then can be produced using a maximum weight matching with R as weights.

Note that $R = A R$ corresponds to finding the stationary distribution of a random walk on $\mathcal{G}_1 \times \mathcal{G}_2$:

This stationary distribution is $R_{i,j} = \frac{\deg(i)\deg(j)}{2|\mathcal{E}_1||\mathcal{E}_2|}$

The equation $R = A R$ can be generalized to include node-similarity information as $R = \alpha A R + (1 - \alpha)E$, where E is normalized to sum to 1 and α is in $[0, 1]$.

We can solve $R = \alpha A R + (1 - \alpha)E$ using a PageRank-like method. A match then can be produced using a maximum weight matching with R as weights.

Note that $R = A R$ corresponds to finding the stationary distribution of a random walk on $\mathcal{G}_1 \times \mathcal{G}_2$:

This stationary distribution is $R_{i,j} = \frac{\deg(i)\deg(j)}{2|\mathcal{E}_1||\mathcal{E}_2|}$

The equation $R = A R$ can be generalized to include node-similarity information as $R = \alpha A R + (1 - \alpha)E$, where E is normalized to sum to 1 and α is in $[0, 1]$.

We can solve $R = \alpha A R + (1 - \alpha)E$ using a PageRank-like method. A match then can be produced using a maximum weight matching with R as weights.

Let $S \in \mathbb{R}^{|\mathcal{V}_1| |\mathcal{V}_2| \times |\mathcal{V}_1| |\mathcal{V}_2|}$ be as in the original IQP for a complete L . Recall that

$S_{i,i',j,j'} = 1$ iff i, i', j, j' make a square through L, \mathcal{G}_1 and \mathcal{G}_2

We can write the adjacency matrix of the product graph $\mathcal{G}_1 \times \mathcal{G}_2$ as $A = D_S^{-1} S^\top$ where $D_S = \text{diag}(\text{rowsum}(S))$

Hence, we can generalize the Isorank to sparse-Isorank as $\alpha D_S^{-1} S^\top R + (1 - \alpha)E = R$ where S , now, can be sparse and allow only some matches.

Sparse Isorank [Bayati et al 09]

Let $S \in \mathbb{R}^{|\mathcal{V}_1| |\mathcal{V}_2| \times |\mathcal{V}_1| |\mathcal{V}_2|}$ be as in the original IQP for a complete L . Recall that

$S_{i,i',j,j'} = 1$ iff i, i', j, j' make a square through L, \mathcal{G}_1 and \mathcal{G}_2

We can write the adjacency matrix of the product graph $\mathcal{G}_1 \times \mathcal{G}_2$ as $A = D_S^{-1} S^\top$ where
 $D_S = \text{diag}(\text{rowsum}(S))$

Hence, we can generalize the Isorank to sparse-Isorank as $\alpha D_S^{-1} S^\top R + (1 - \alpha)E = R$ where S , now, can be sparse and allow only some matches.

Start with the Chemical distance definition. Then do,

$$\begin{aligned} (1) \arg \min_{P \in \Pi} \|AP - PB\|_F^2 &= \arg \min_{P \in \Pi} \|AP\|_F^2 + \|PB\|_F^2 - 2\langle AP, PB \rangle \\ &= \arg \max_{P \in \Pi} \langle AP, PB \rangle \rightarrow (2) \arg \max_{P \in \text{Doubly stochastic}} \langle AP, PB \rangle \end{aligned}$$

Theorem: For several families of random graphs, with high probability, solving (2) gives a solution to (1).

(2) Can be approx. solved easily using, e.g., projected gradient descent. When (2) does not return a permutation, we use rounding methods.

Start with the Chemical distance definition. Then do,

$$\begin{aligned} (1) \arg \min_{P \in \Pi} \|AP - PB\|_F^2 &= \arg \min_{P \in \Pi} \|AP\|_F^2 + \|PB\|_F^2 - 2\langle AP, PB \rangle \\ &= \arg \max_{P \in \Pi} \langle AP, PB \rangle \rightarrow (2) \arg \max_{P \in \text{Doubly stochastic}} \langle AP, PB \rangle \end{aligned}$$

Theorem: For several families of random graphs, with high probability, solving (2) gives a solution to (1).

(2) Can be approx. solved easily using, e.g., projected gradient descent. When (2) does not return a permutation, we use rounding methods.

This alg. can, sometimes, determine if two graphs are non-isomorphic:

1. Color the two graphs with nodes of equal color
2. For each graph do, for each node i
$$color_i^{t+1} = \text{hash}(\text{sort}(\{color_j^t\}_{j \in \text{neig. of } i}))$$
3. Once colors are stable, compare the distributions of the colors in the two graphs. If they are different, output non-isomorphic.

The final colors can be use to find an (inexact) matching. Find a cost function to compare colors, and use a maximum weight matching to match nodes.

These algorithms can align large graphs ($\sim 300k$ nodes per graph, and $\sim 20M$ possible matches between nodes [Bayati et al. 2009]).

A few can be solved using distributed message passing schemes and hence, at least in principle, can scale to very large graphs: WL alg., NetAlignBP, Isorank.

Optimization-based algs., e.g. the LP relaxation, Klau's alg., Natalie 2.0, and Inner alg., can be solved using standard distributed optimization methods.

Among existing methods for large scale distributed optimization, worth noting is the **Alternating Direction Method of Multipliers**.

1. Can deal with non-smooth functions
2. Easily distributed and parallelized
3. Good convergence properties, and empirically good performance in several non-convex problems
4. Fastest possible first-order method among strongly convex functions with Lipschitz gradients [França & Bento 2016]

Unfortunately, if we use the alignment produced by these algorithms to produce a distance between graphs, e.g. using an alignment's permutation matrix to compute

$$\|AP - PB\|$$

, the resulting distances are not metrics. This is the case for Natalie 2.0, Klau's alg., IsoRank, SparselsoRank and Inner alg.

Can we find a rich set of scalable graph comparison methods that result in metrics?

(part II of this tutorial)



Grants IIS-1741197 & IIS-1741129



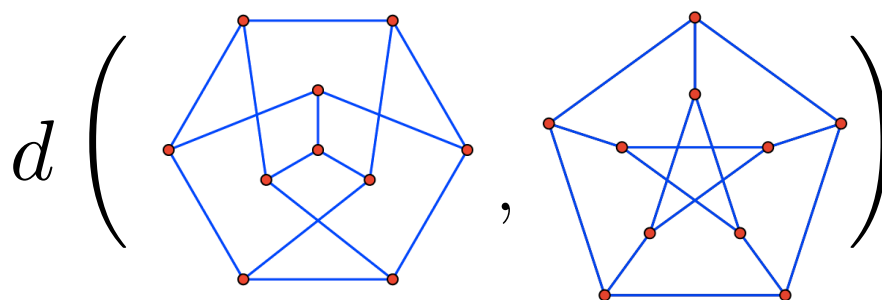
Northeastern



BOSTON COLLEGE

Part II: A Family of Tractable Graph Metrics

A Highly Desirable Property



- $d(x, y) \geq 0$ (non-negativity)
- $d(x, y) = 0$ iff $x = y$ (pos. definiteness)
- $d(x, y) = d(y, x)$ (symmetry)
- $d(x, y) \leq d(x, z) + d(z, y)$ (triangle inequality)

Distance score d is a **metric**

❑ Poly-time Algorithms with provable guarantees:

- ❑ k-NN
- ❑ Clustering
- ❑ Dataset diameter
- ❑ ...

❑ Work very well in practice.

Example: Chemical Distance

Given two adjacency matrices $A, B \in \{0, 1\}^{n \times n}$,
their **chemical distance** is:

$$d_{\mathbb{P}^n}(A, B) = \min_{P \in \mathbb{P}^n} \|AP - PB\|_F,$$

Optimal mapping
between nodes

where

$$\mathbb{P}^n = \{P \in \{0, 1\}^{n \times n} : P\mathbf{1} = \mathbf{1}, P^\top \mathbf{1} = \mathbf{1}\}$$

Count edge
differences

is the set of **permutation matrices** and $\|\cdot\|_F$ is the Frobenius norm.

- ✓ **"Natural"**: minimal edge discrepancy
- ✓ Zero iff graphs are **isomorphic**
- ✓ **Metric**

✗ Intractable

Example: Chartrand-Kubiki-Shultz (CKS) Distance

Given two shortest path distance matrices $A, B \in \mathbb{R}_{+}^{n \times n}$,
their **CKS distance** is:

$$d_{\mathbb{P}^n}(A, B) = \min_{P \in \mathbb{P}^n} \|AP - PB\|_F,$$

where

$$\mathbb{P}^n = \{P \in \{0, 1\}^{n \times n} : P\mathbf{1} = \mathbf{1}, P^\top \mathbf{1} = \mathbf{1}\}$$

is the set of **permutation matrices** and $\|\cdot\|_F$ is the Frobenius norm.

✓ Zero iff graphs are **isomorphic**

✓ **Metric**

✗ Intractable

Tractable Approaches

$$d_{\mathbb{P}^n}(A, B) = \min_{P \in \mathbb{P}^n} \|AP - PB\|_F,$$

- ❑ Inexact solution: approximate optimal matrix $P \in \mathbb{P}^n$

[Singh et al., 2007], [Bayati et al. 2009]
[Klau 2009], [Koutra et al. 2013]
[Kebir et al. 2015], [Lyzinski et al. 2016]

- ❑ Convex relaxations: SDP relaxation, method of moments,...
- ❑ Resulting d is **not** a metric!!
- ❑ Challenge: produce **tractable metrics**



A Family of Tractable Graph Metrics

- Generalization of chemical & CKS distances:

[Bento, Ioannidis SDM 2018]

$$d_S(A, B) = \min_{P \in S} \|AP - PB\| \quad (*)$$

Closed & Bounded Set

Matrix Norm

- Conditions on $S, \|\cdot\|$ under which d_S is a metric

- d_S is indeed a metric when either:

$S = \text{Doubly Stochastic Matrices}, \quad \text{or} \quad S = \text{Orthogonal Matrices}$

In both cases, $(*)$ is **tractable**.

- Extension of $(*)$ to incorporate **node attributes**.



Outline

- ❑ Relaxations of Chemical & CKS distances
- ❑ Incorporating Metric Embeddings
- ❑ Distributed Computation

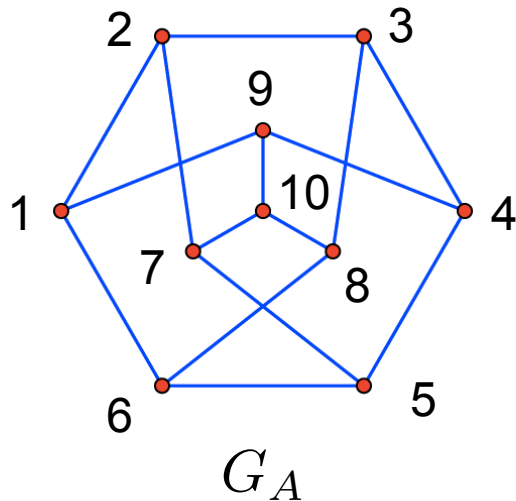


- ❑ Relaxations of Chemical & CKS distances

- ❑ Incorporating Metric Embeddings

- ❑ Distributed Computation

Graph Spaces



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

adjacency matrix A

□ Undirected graphs:

$$\Omega = \{0, 1\}^{n \times n} \cap \mathbb{S}^n \quad (\text{binary, symmetric})$$

□ Weighted, undirected graphs:

$$\Omega = \mathbb{S}^n \quad (\text{real, symmetric})$$

□ Weighted, directed graphs:

$$\Omega = \mathbb{R}^{n \times n} \quad (\text{real})$$



Problem Formulation

□ For $A, B \in \Omega$:

$$d_S(A, B) = \min_{P \in S} \|AP - PB\|$$

Closed & Bounded Set

Matrix Norm

□ $\Omega =$ e.g., binary, binary symmetric, real symmetric, real

□ $S =$

$$\mathbb{P}^n = \{P \in \{0, 1\}^{n \times n} : P\mathbf{1} = P^\top \mathbf{1} = \mathbf{1}\}$$

$$\mathbb{W}^n = \{P \in [0, 1]^{n \times n} : P\mathbf{1} = P^\top \mathbf{1} = \mathbf{1}\}$$

$$\mathbb{O}^n = \{P \in \mathbb{R}^{n \times n} : PP^\top = P^\top P = I\}$$

permutation matrices

doubly-stochastic matrices
(a.k.a. the *Birkhoff Polytope*)

orthogonal matrices
(a.k.a. the *Stiefler Manifold*)



Why These Relaxations?

$$\mathbb{P}^n = \{P \in \{0, 1\}^{n \times n} : P\mathbf{1} = P^\top \mathbf{1} = \mathbf{1}\}$$

permutation matrices

$$\mathbb{W}^n = \{P \in [0, 1]^{n \times n} : P\mathbf{1} = P^\top \mathbf{1} = \mathbf{1}\}$$

doubly-stochastic matrices

$$\mathbb{O}^n = \{P \in \mathbb{R}^{n \times n} : PP^\top = P^\top P = I\}$$

orthogonal matrices

$$\mathbb{W}^n = \text{conv}(\mathbb{P}^n)$$

(Birkhoff-von Neumann Theorem)

$$\mathbb{P}^n = \mathbb{W}^n \cap \mathbb{O}^n$$



Metrics

□ Function $d : \Omega \times \Omega \rightarrow \mathbb{R}$ is a **metric** over set Ω if it satisfies the following properties:

$$d(x, y) \geq 0 \quad (\text{non-negativity})$$

$$d(x, y) = 0 \text{ iff } x = y \quad (\text{pos. definiteness})$$

$$d(x, y) = d(y, x) \quad (\text{symmetry})$$

$$d(x, y) \leq d(x, z) + d(z, y) \quad (\text{triangle inequality})$$

Pair (Ω, d) is then called a **metric space**.



Pseudo-metrics & Quasi-metrics

□ Function $d : \Omega \times \Omega \rightarrow \mathbb{R}$ is a **pseudo-metric** over set Ω if it satisfies the following properties:

$$d(x, y) \geq 0 \quad (\text{non-negativity})$$

$$d(x, y) = 0 \text{ if } x = y$$

$$d(x, y) = d(y, x) \quad (\text{symmetry})$$

$$d(x, y) \leq d(x, z) + d(z, y) \quad (\text{triangle inequality})$$

If d is a pseudo-metric, then it is a metric over equivalence classes implied by $d(x, y) = 0$ (i.e., the **quotient** space)

Pseudo-metrics & Quasi-metrics

□ Function $d : \Omega \times \Omega \rightarrow \mathbb{R}$ is a **quasi-metric** over set Ω if it satisfies the following properties:

$$d(x, y) \geq 0 \quad (\text{non-negativity})$$

$$d(x, y) = 0 \text{ iff } x = y$$

~~$$d(x, y) = d(y, x) \quad (\text{symmetry})$$~~

$$d(x, y) \leq d(x, z) + d(z, y) \quad (\text{triangle inequality})$$

If d is a quasi-metric, then the **symmetric extension**

$$\bar{d}(x, y) = d(x, y) + d(y, x)$$

is a metric.

Optimization over Permutation Matrices

$$d_S(A, B) = \min_{P \in S} \|AP - PB\| \quad (*)$$

$$S = \mathbb{P}^n = \{P \in \{0, 1\}^{n \times n} : P\mathbf{1} = P^\top \mathbf{1} = \mathbf{1}\}$$

Theorem: If $S = \mathbb{P}^n$ and $\|\cdot\|$ is an arbitrary entry-wise or operator matrix norm, then d_S is a pseudo-metric over $\Omega = \mathbb{R}^{n \times n}$

- ❑ Weighted, directed graphs
- ❑ Equivalence relation=Isomorphism



Optimization over Doubly Stochastic Matrices

$$d_S(A, B) = \min_{P \in S} \|AP - PB\| \quad (*)$$

$$S = \mathbb{W}^n = \{P \in [0, 1]^{n \times n} : P\mathbf{1} = P^\top \mathbf{1} = \mathbf{1}\}$$

Theorem: If $S = \mathbb{W}^n$ and $\|\cdot\|$ is an arbitrary entry-wise matrix norm, then d_S is a pseudo-metric over $\Omega = \mathbb{S}^n$.

If $\|\cdot\|$ is an operator norm or $\Omega = \mathbb{R}^{n \times n}$, then it is a quasi-metric.

- ❑ Weighted, **undirected** graphs + entry-wise norms o.k.
- ❑ Operator norms/directed graphs break symmetry
- ❑ Equivalence classes characterized by Weisfeiler-Lehman algorithm.
- ❑ **Tractable:**(*) is a convex optimization problem!



Optimization over Orthogonal Matrices

$$d_S(A, B) = \min_{P \in S} \|AP - PB\| \quad (*)$$

$$S = \mathbb{O}^n = \{P \in \mathbb{R}^{n \times n} : PP^\top = P^\top P = I\}$$

Theorem: If $S = \mathbb{O}^n$ and $\|\cdot\|$ is either the operator or the entry-wise 2-norm, then d_S is a pseudo-metric over $\Omega = \mathbb{R}^{n \times n}$

- ❑ Weighted, directed graphs
- ❑ Restricted to 2-norms
- ❑ Equivalence classes characterized by co-spectrality.
- ❑ **Tractable:** (*) is not convex, but can be solved via a spectral decomposition.



Summary

- We can construct pseudo-metrics for $S = \mathbb{P}^n$, \mathbb{W}^n , and \mathbb{O}^n .
- In the latter two cases, computing $d_S(A, B)$ is tractable.



Clustering Performance

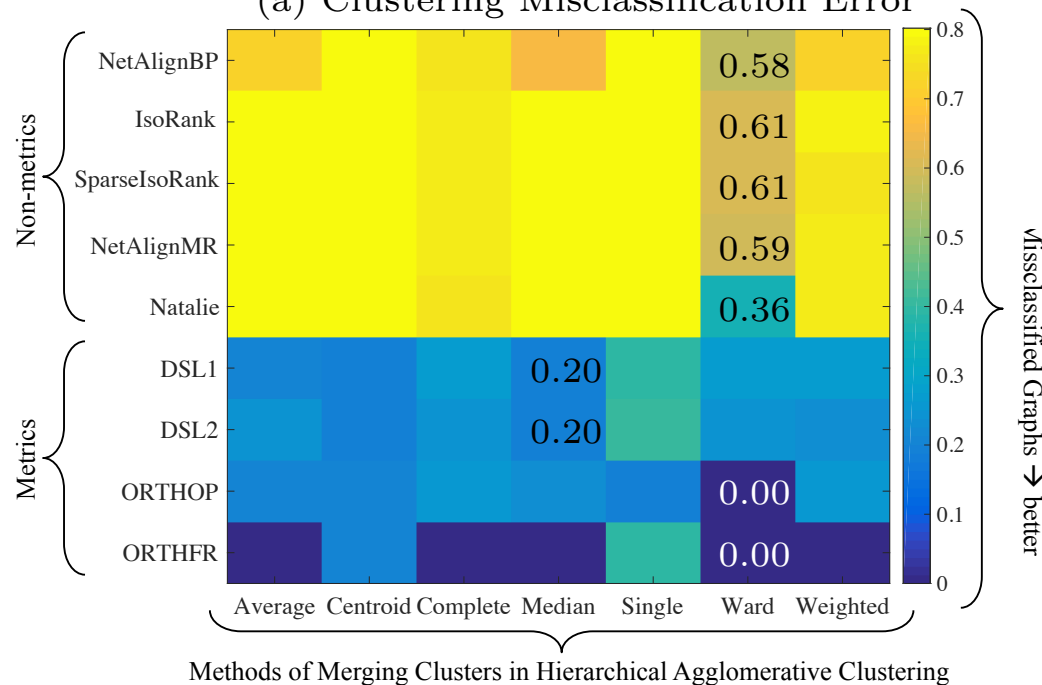
Distance Score Algorithms

(Non-metric) Distance Score Algorithms	
NetAlignBP	Network Alignment using Belief Propagation [9, 33]
IsoRank	Neighborhood Topology Isomorphism using Page Rank [54, 33]
SparseIsoRank	Neighborhood Topology Sparse Isomorphism using Page Rank [9, 33]
InnerPerm	Inner Product Matching with Permutations [42]
InnerDSL1	Inner Product Matching with Matrices in \mathbb{W}^n and entry-wise 1-norm [42]
InnerDSL2	Inner Product Matching with Matrices in \mathbb{W}^n and Frobenius norm [42]
NetAlignMR	Iterative Matching Relaxation [34, 33]
Natalie (V2.0)	Improved Iterative Matching Relaxation [23, 22]
Metrics from our Family (2.4)	
EXACT	Chemical Distance via brute force search over GPU
DSL1	Doubly Stochastic Chemical Distance $d_{\mathbb{W}n}$ with entry-wise 1-norm
DSL2	Doubly Stochastic Chemical Distance $d_{\mathbb{W}n}$ with Frobenius norm
ORTHOP	Orthogonal Relaxation of Chemical Distance $d_{\mathbb{O}n}$ with operator 2-norm
ORTHFR	Orthogonal Relaxation of Chemical Distance $d_{\mathbb{O}n}$ with Frobenius norm

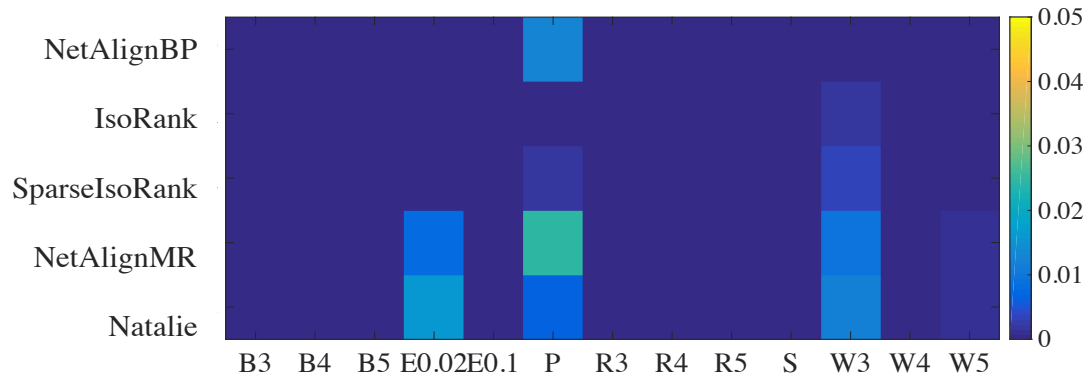
Graph Classes

	Description
Bd	Barabasi Albert of degree d [5]
Ep	Erdős-Rényi with probability p [25]
P	Power Law Tree [44]
Rd	Regular Graph of degree d [13]
S	Small World [35]
Wd	Watts Strogatz of degree d [58]

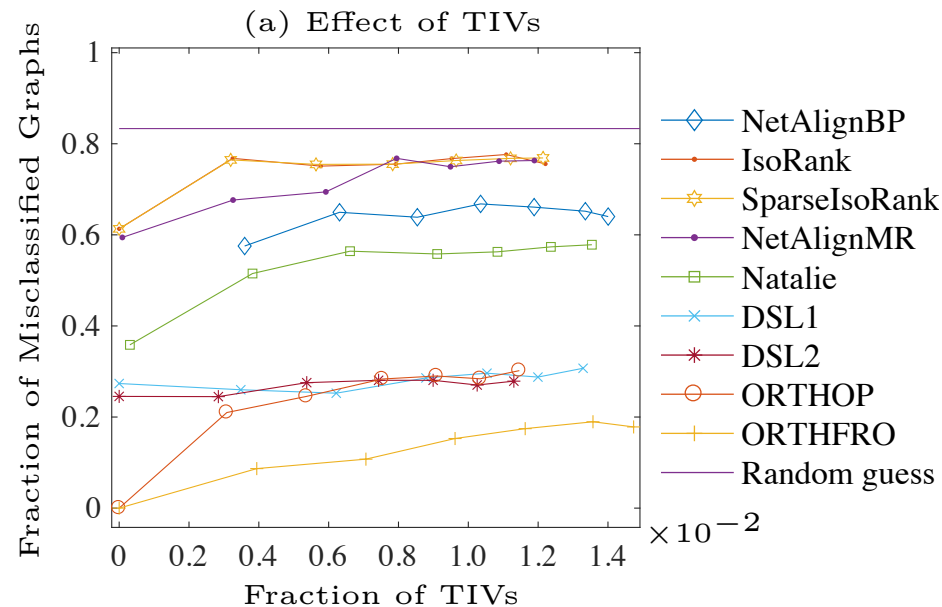
(a) Clustering Misclassification Error



Triangle Inequality Violations (TIVs)



Bd	Barabasi Albert of degree d [5]
Ep	Erdős-Rényi with probability p [25]
P	Power Law Tree [44]
Rd	Regular Graph of degree d [13]
S	Small World [35]
Wd	Watts Strogatz of degree d [58]



Outline

- ❑ Relaxations of Chemical & CKS distances
- ❑ Incorporating Metric Embeddings
- ❑ Distributed Computation

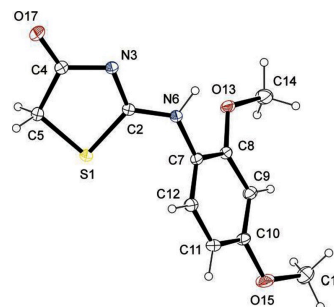


Node Attributes

❑ Nodes often have attributes

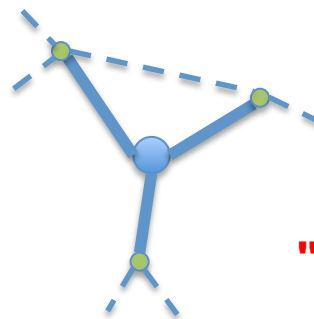
❑ Exogenous (gender, age, in social network, atomic number in molecule, etc.)

$$x_v = \begin{array}{c} \text{Gender} \\ \text{Salary} \\ \text{Age} \end{array} \begin{array}{|c|c|c|} \hline \text{♀} & 70\text{K} & 20\text{ys} \\ \hline \end{array}$$



❑ Endogenous (degree, number of triangles, pagerank, etc.)

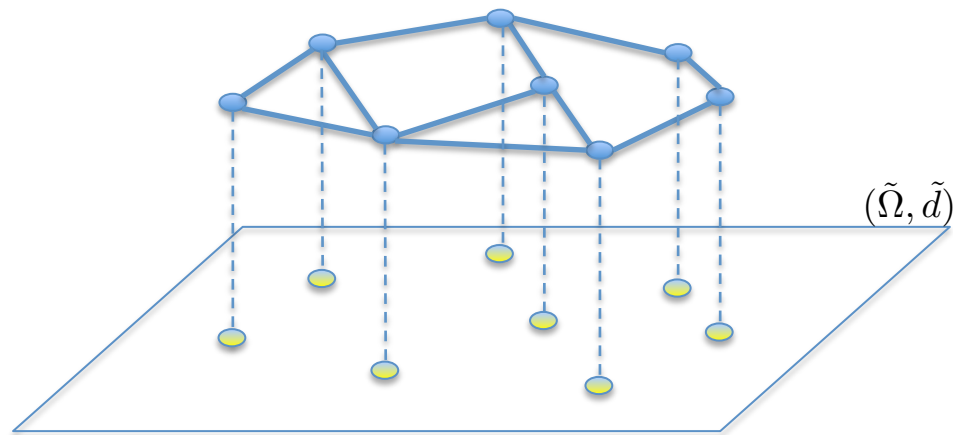
$$x_v = \begin{array}{c} \text{Degree} \\ \text{\#Triangles} \\ \text{PageRank} \end{array} \begin{array}{|c|c|c|} \hline 3 & 1 & 0.46 \\ \hline \end{array}$$



**Find mappings that map
"similar" nodes to each other**

Metric Embeddings

- **Metric embedding:** Mapping of nodes to metric space $(\tilde{\Omega}, \tilde{d})$.



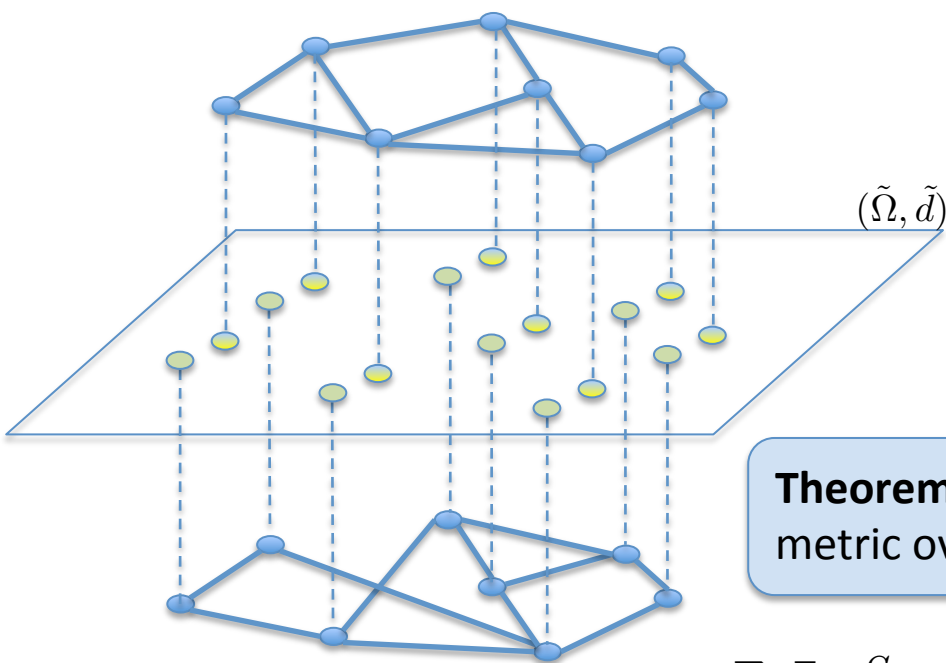
E.g. :

$$x_v = \begin{array}{c|c|c} \text{Gender} & \text{Salary} & \text{Age} \\ \hline \text{♀} & 70\text{K} & 20\text{ys} \end{array}$$

$$\tilde{\Omega} = \mathbb{R}^d, \quad \tilde{d} = \text{Euclidian distance}$$

Incorporating Node Attributes

- ❑ Consider two graphs embedded in the **same metric space** $(\tilde{\Omega}, \tilde{d})$.
- ❑ Seek permutations that map nodes to other **nearby/proximal** nodes.



$\tilde{D}_{A,B} \in \mathbb{R}^{n \times n}$ Pairwise distances between nodes in $(\tilde{\Omega}, \tilde{d})$

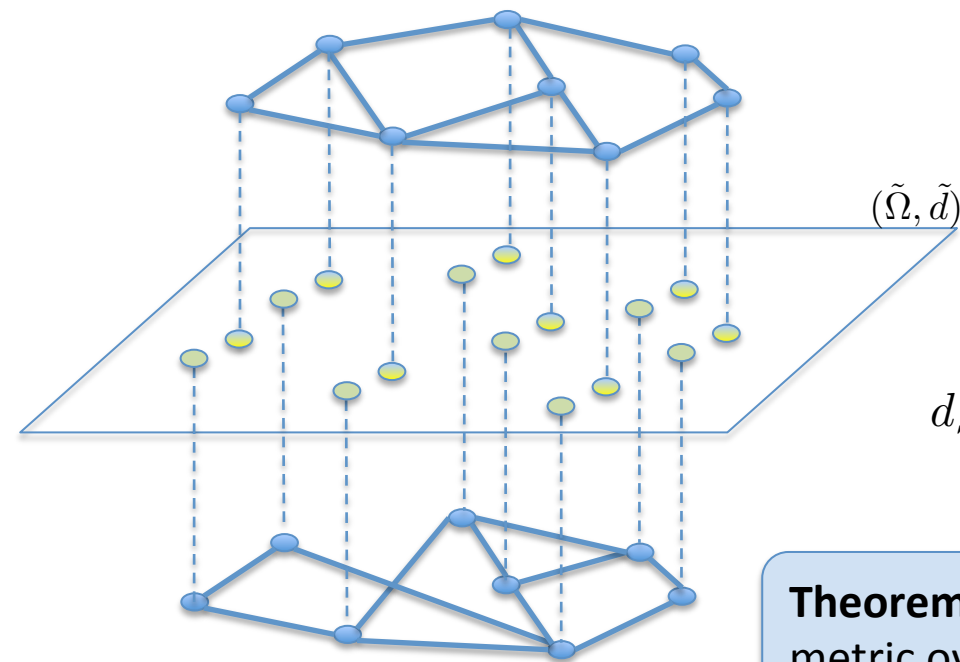
$$d_S(A, B) = \min_{P \in \mathcal{S}} \text{tr}(P^\top \tilde{D}_{A,B})$$

Theorem: If $S = \mathbb{P}^n$ or $S = \mathbb{W}^n$, then d_S is a pseudo-metric over graphs embedded in $(\tilde{\Omega}, \tilde{d})$.

- ❑ For $S = \mathbb{W}^n$, optimization is **convex**!
- ❑ For $S = \mathbb{P}^n$, optimization is **polytime-solvable!**
(Hungarian algorithm)

Incorporating Node Attributes

- ❑ Consider two graphs embedded in the **same metric space** $(\tilde{\Omega}, \tilde{d})$.
- ❑ Seek permutations that map nodes to other **nearby/proximal** nodes.



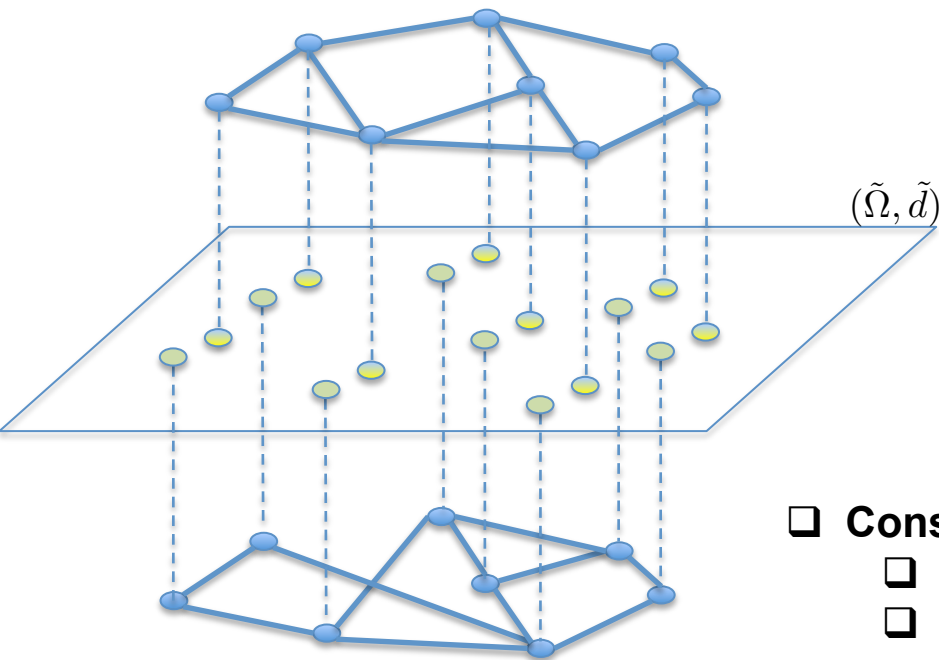
Pairwise distances between nodes in $(\tilde{\Omega}, \tilde{d})$

$$d_S(A, B) = \min_{P \in S} \left(\|AP - PB\| + \text{tr}(P^\top \tilde{D}_{A,B}) \right)$$

Theorem: If $S = \mathbb{P}^n$ or $S = \mathbb{W}^n$, then d_S is a pseudo-metric over graphs embedded in $(\tilde{\Omega}, \tilde{d})$.

- ❑ For $S = \mathbb{W}^n$, optimization is **still tractable**!

Important Practical Implications



$$d_S(A, B) = \min_{P \in \mathcal{S}} \left(\|AP - PB\| + \text{tr}(P^\top \tilde{D}_{A,B}) \right)$$

- ❑ **Endogenous features** in \mathbb{R}^d : degree, pagerank, etc.
 - ❑ Speed up convergence
- ❑ **Constraints**
 - ❑ Map nodes of degree k only to nodes of degree k
 - ❑ Map females to females, oxygens to oxygens, etc.
- ❑ Reduces # variables in optimization

$$\tilde{\Omega} = \{0, 1, 2, 3, \dots\} \quad (\text{degrees, atomic numbers, ...})$$

$$\tilde{d}(x, y) = \begin{cases} 0, & \text{if } x = y, \\ \infty, & \text{if } x \neq y. \end{cases} \quad (\text{Dirac distance})$$

Constraints maintain metric property!!

How to Pick an Embedding?

- ❑ Embedding must place nodes in the *same metric space*.
- ❑ Mapping must be *unique* (in particular, *deterministic*)
- ❑ Preferred property: embedding is *permutation invariant*.

❑ Possible Example Embeddings

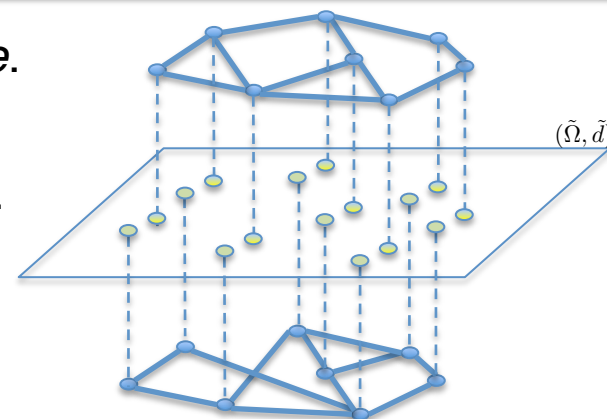
- ❑ Local/node centric features
 - ❑ WL counts, degrees, cycles, k-hop neighborhoods
- ❑ Laplacian Eigenmaps [Hoffman and Buchanan 94, Balasubramanian and Swartz 02, He and Nyogi 03]
- ❑ Eigenmaps of Non-Backtracking Walks **[See Part III of tutorial!!!]**

❑ Non-Examples:

- ❑ Nearly-isometric embeddings [Linial et al. 96, Matousek 99, Bourgain 86, Rao 95]
- ❑ Matrix factorization [Nikoletzos et al. 17, Ou et al 16, Shaw and Jebara 09]
- ❑ Path-NLP based methods [DeepWalk, node2vec, etc.]



*Need for
research on how
to co-embed*



More on Embeddings: T6/44:

[Modeling Data With Networks + Network Embedding: Problems, Methodologies and Frontiers](#)

Peng Cui (Tsinghua University), Jian Pei (SFU), Wenwu Zhu (Tsinghua University),

Tanya Berger-Wolf (UIC), Ivan Brugere (UIC) Bryan Perozzi (Google)

ICC Capital Suite Room 11 (Level 3), 1:00 PM - 5:00 PM



Outline

- ❑ Relaxations of Chemical & CKS distances
- ❑ Incorporating Metric Embeddings
- ❑ **Distributed Computation**



Distributing Computation

$$\|AP - PB\|_1 + \text{tr}(P^\top \tilde{D}_{A,B}) = \sum_{i=1}^n \sum_{j=1}^n \left| \sum_k a_{ik} p_{kj} - \sum_k p_{ik} b_{kj} \right| + \sum_{i=1}^n \sum_{j=1}^n p_{ij} \tilde{d}_{ij}$$

$$\sum_{i=1}^n p_{ij} = 1 \quad \sum_{j=1}^n p_{ij} = 1$$

- ❑ Objective can be written as sum of convex functions
- ❑ Solution can be parallelized via *consensus ADMM*

Boyd, Stephen, et al. "Distributed optimization and statistical learning via the alternating direction method of multipliers." *Foundations and Trends® in Machine learning* 3.1 (2011): 1-122.



Distributing Computation

$$\|AP - PB\|_p + \text{tr}(P^\top \tilde{D}_{A,B}) = \sqrt[p]{\sum_{i=1}^n \sum_{j=1}^n \left| \sum_k a_{ik} p_{kj} - \sum_k p_{ik} b_{kj} \right|^p} + \sum_{i=1}^n \sum_{j=1}^n p_{ij} \tilde{d}_{ij}$$

$$\sum_{i=1}^n p_{ij} = 1 \quad \sum_{j=1}^n p_{ij} = 1$$

- ❑ Objective **cannot** be written as sum of convex functions
- ❑ Solution can still be parallelized through map and reduce operations+ADMM



Parallel Implementation

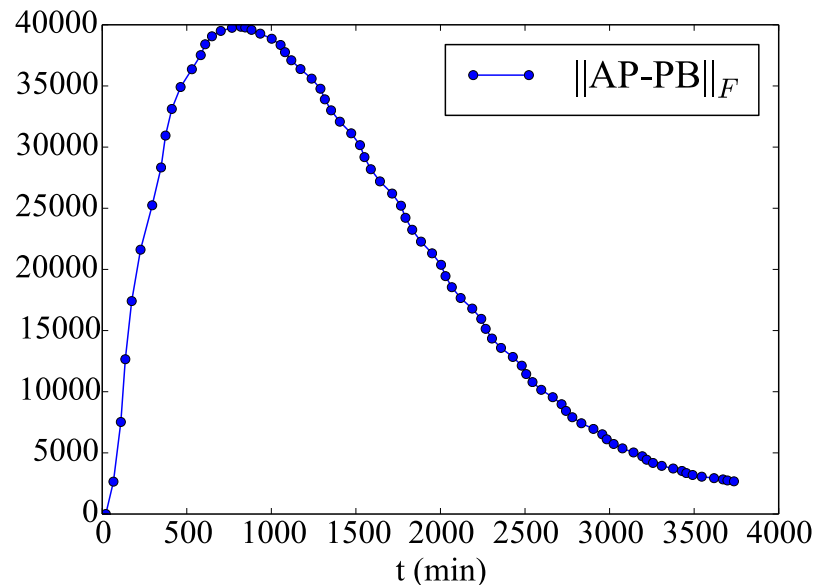
- ❑ Apache Spark
- ❑ ADMM

Slashdot graph

- ❑ $n = 82,168$ nodes
- ❑ $|E| = 948,464$ edges

- ❑ Constraints: WL-coloring algorithm

k	# vars
0	27,478,564
1	3,747,960
2	239,048
3	182,474
4	182,016
5	182,006



- ❑ 448 CPUs = 8 machines x 56 cores each





Northeastern



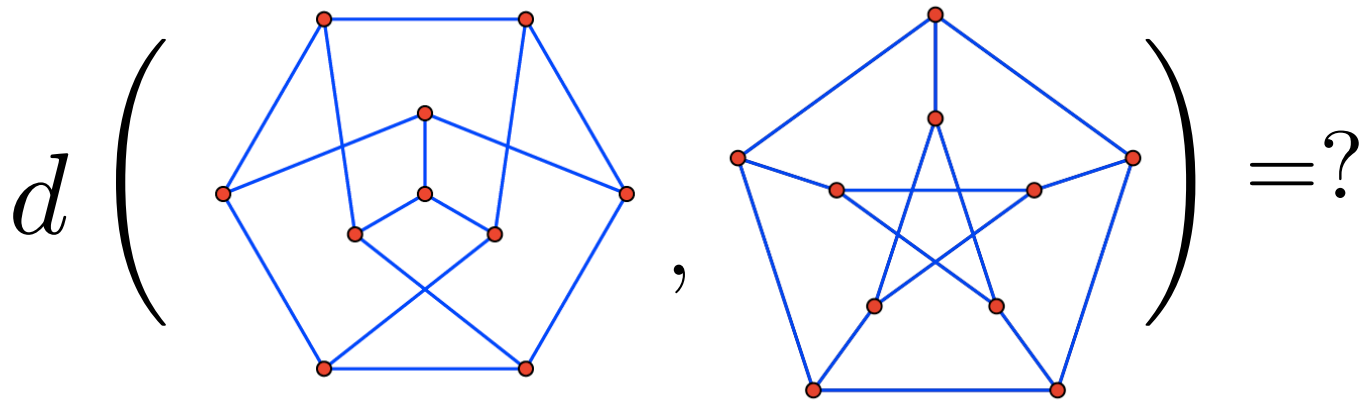
BOSTON COLLEGE

**Coming Up:
...Non-backtracking Matrix,
Graph Distances, and Metric
Embeddings**



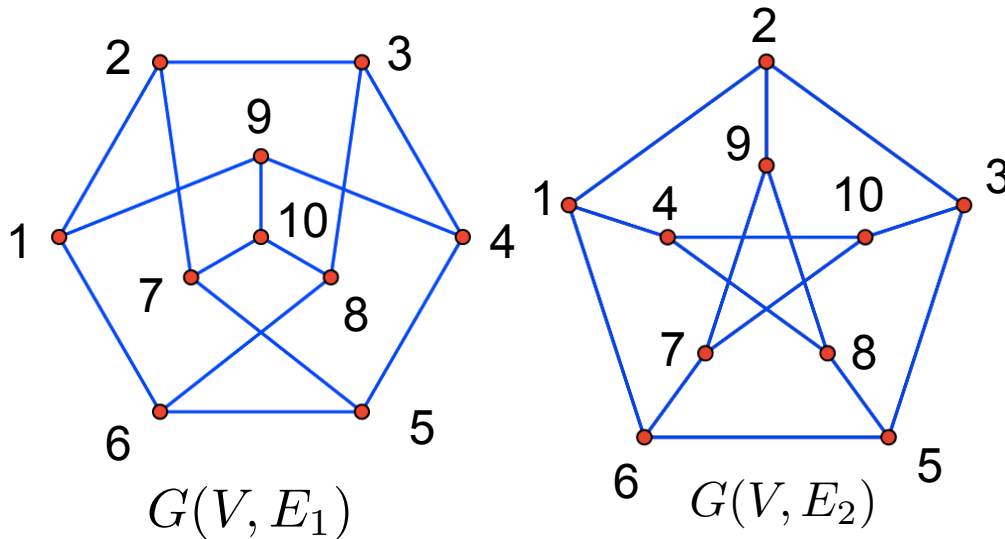
Grants IIS-1741197 and IIS-1741129

Graph Distances & Graph Similarity



□ Given two graphs as input, **how similar are they?**

Graph Distances & Graph Similarity

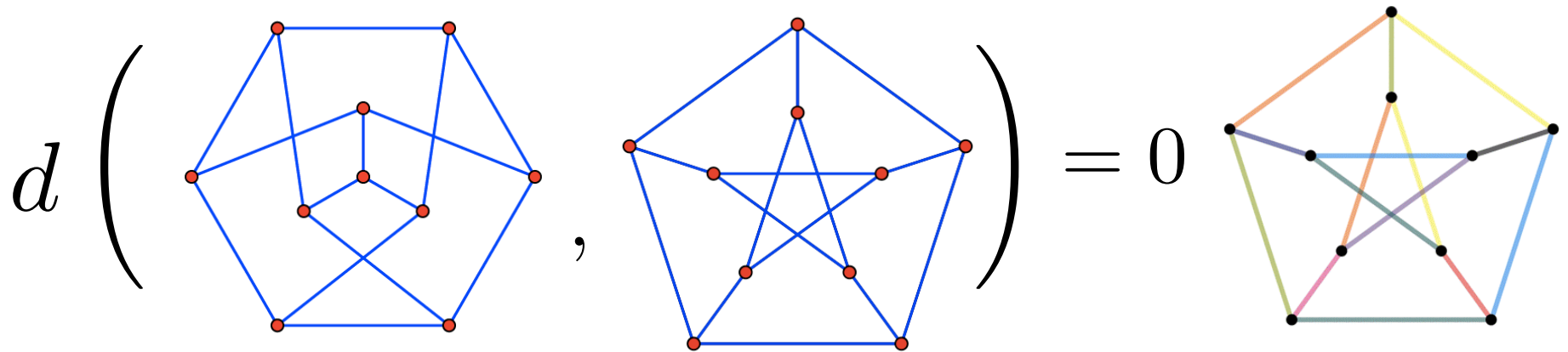


Edit Distance:

$$|E_1 \setminus E_2| + |E_2 \setminus E_1|$$

- ❑ Given two graphs as input, **how similar are they?**
- ❑ "Labeled" setting: correspondence between nodes given.

Graph Distances & Graph Similarity

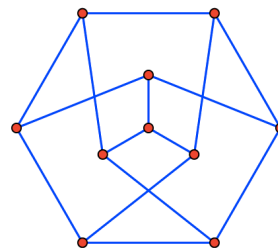


- ❑ Given two graphs as input, **how similar are they?**
- ❑ "Unlabeled" setting: no prior correspondence between nodes.

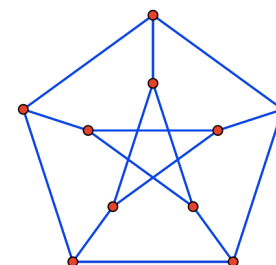
Applications of Graph Distances

❑ Key in many Graph Mining Tasks

- ❑ Graph De-anonymization
- ❑ k-Nearest Neighbors Search
- ❑ Clustering
- ❑ ...



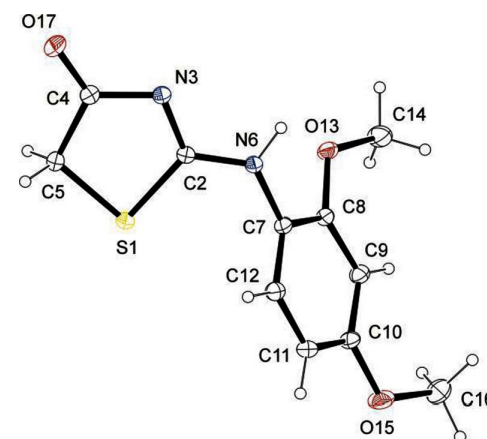
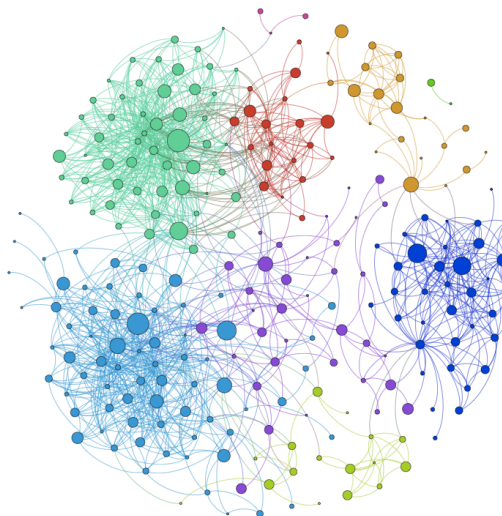
friendship graph



phone-call graph

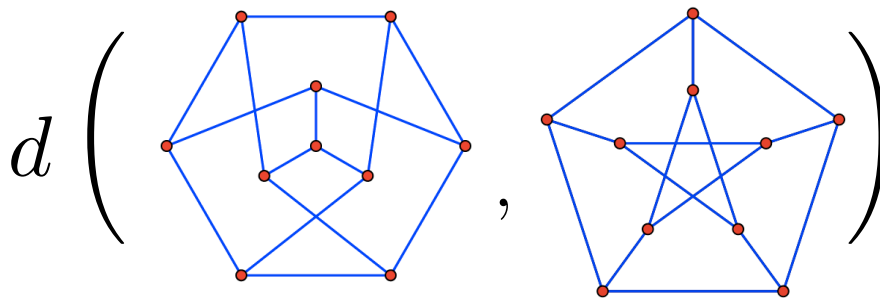
❑ Graphs are Ubiquitous

- ❑ Social Networks
- ❑ Computer Vision
- ❑ Chemistry
- ❑ Computer Networks
- ❑ ...



Cambridge Crystallographic
Data Centre

A Highly Desirable Property



- $d(x, y) \geq 0$ (non-negativity)
- $d(x, y) = 0$ iff $x = y$ (pos. definiteness)
- $d(x, y) = d(y, x)$ (symmetry)
- $d(x, y) \leq d(x, z) + d(z, y)$ (triangle inequality)

Distance score d is a **metric**

❑ Poly-time Algorithms with provable guarantees:

- ❑ k-NN
- ❑ Clustering
- ❑ Dataset diameter
- ❑ ...

❑ Work very well in practice.

Metrics

□ Function $d : \Omega \times \Omega \rightarrow \mathbb{R}$ is a **metric** over set Ω if it satisfies the following properties:

$$d(x, y) \geq 0 \quad (\text{non-negativity})$$

$$d(x, y) = 0 \text{ iff } x = y \quad (\text{pos. definiteness})$$

$$d(x, y) = d(y, x) \quad (\text{symmetry})$$

$$d(x, y) \leq d(x, z) + d(z, y) \quad (\text{triangle inequality})$$

Pair (Ω, d) is then called a **metric space**.





Grants IIS-1741197 & IIS-1741129



Northeastern



BOSTON COLLEGE

Part III: Non-backtracking Matrix, Graph Distance, and Metric Embedding

Outline

Part 1.

Graphs as metric spaces

Leo Torres, Pablo Suárez Serrato, & Tina Eliassi-Rad ([arXiv:1807.09592](https://arxiv.org/abs/1807.09592))

- The length spectrum
- Modifying the length spectrum
- Detour: Non-backtracking matrix (NBM)
- Graph distance
- Properties & examples

Part 2.

Geometry of graph embeddings

Leo Torres & Tina Eliassi-Rad

- Vectorization vs. pattern recognition
- Literature review
- Edge embedding with NBM
- Examples



Isometry in metric spaces

Two metric spaces (Ω_1, d_1) and (Ω_2, d_2) are called **isometric** when there exists a function:

$$f: \Omega_1 \rightarrow \Omega_2 \text{ such that } d_1(x, y) = d_2(f(x), f(y))$$

Isometry and graphs as metric spaces

Two metric spaces (Ω_1, d_1) and (Ω_2, d_2) are called **isometric** when there exists a function:

$$f: \Omega_1 \rightarrow \Omega_2 \text{ such that } d_1(x, y) = d_2(f(x), f(y))$$

Graphs as metric spaces

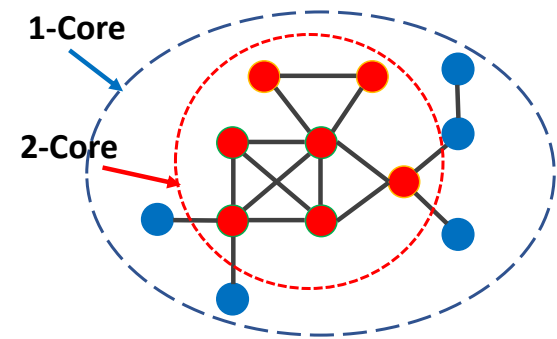
- Graph G
 - $\Omega_1 =$ node set of G and $d_1 =$ a distance function on Ω_1
- Graph H
 - $\Omega_2 =$ node set of H and $d_2 =$ a distance function on Ω_2
- $f =$ a node-correspondence function

From isometry to isomorphism

- Given a graph $G = (V, E)$, the set of nodes is a metric space under the shortest path distance
- If two unweighted graphs are **isometric** with the shortest path distance, then they are **isomorphic**

The length spectrum

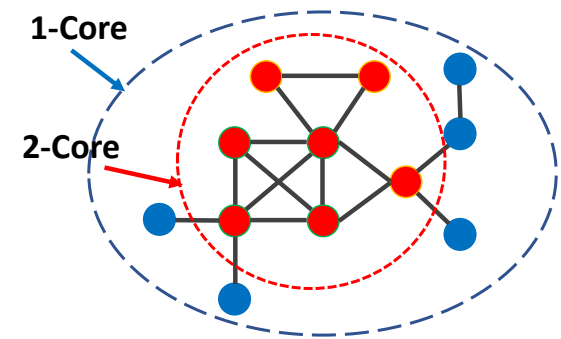
The **length spectrum** of a graph characterizes its 2-core uniquely up to **isometry**.



- Constantine, David, and Jean-François Lafont. “Marked Length Rigidity for One-Dimensional Spaces.” *Journal of Topology and Analysis*, 2018.

The length spectrum

The **length spectrum** of a graph characterizes its 2-core uniquely up to **isometry**.

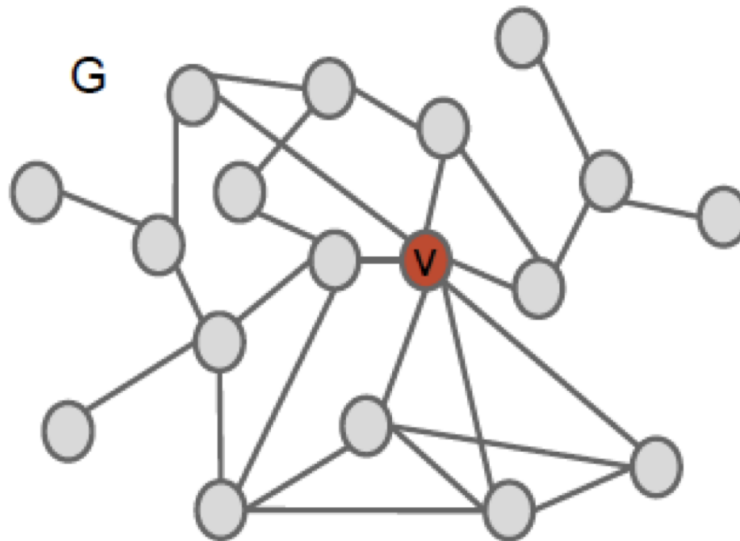


- Constantine, David, and Jean-François Lafont. “Marked Length Rigidity for One-Dimensional Spaces.” *Journal of Topology and Analysis*, 2018.

➔ If two graphs have the same **length spectrum**, then their 2-cores are **isometric**.

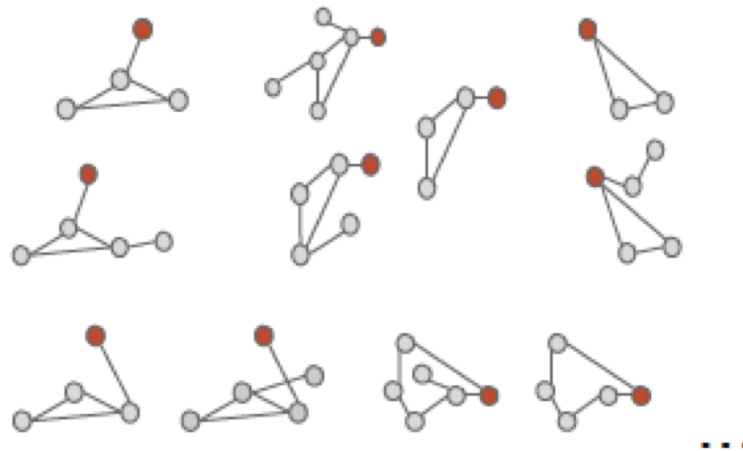
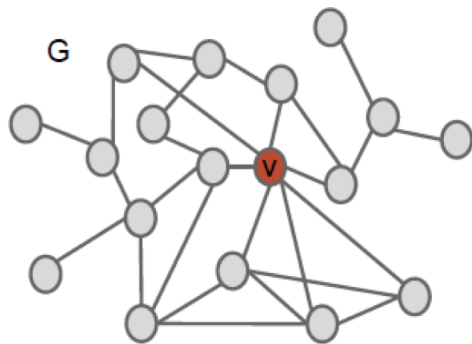
How to construct the length spectrum?

- Given a graph $G = (V, E)$ and a node $v \dots$



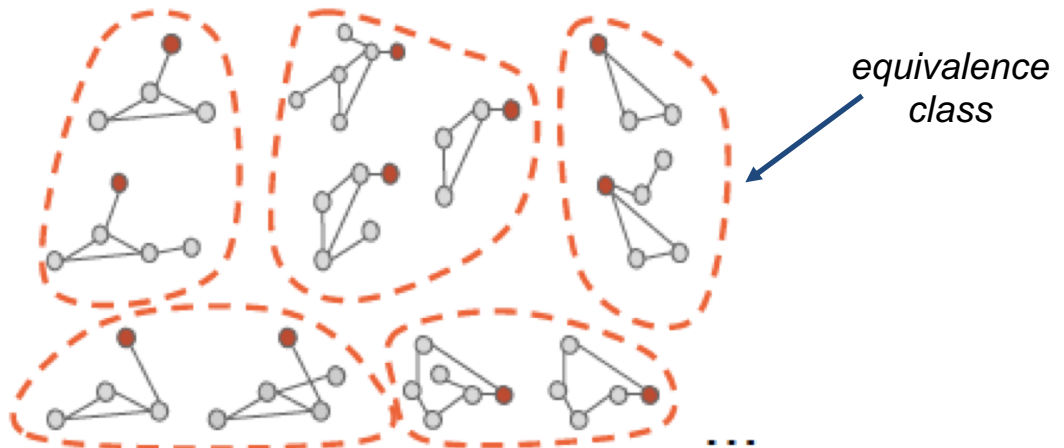
Closed walks

- ... consider the set of all closed walks that start and end at node v



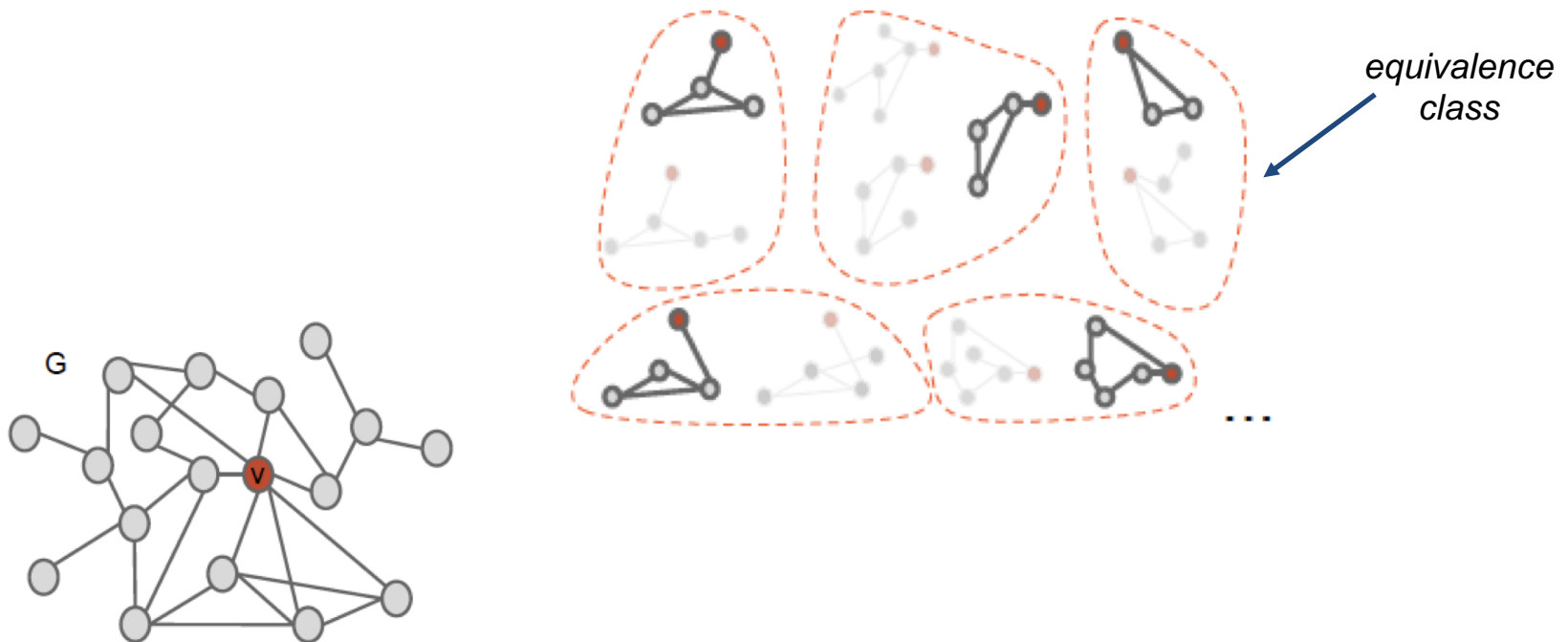
Equivalence of closed walks

- Closed walks are **equivalent** if they differ by tree-like parts that don't go through the basepoint ...



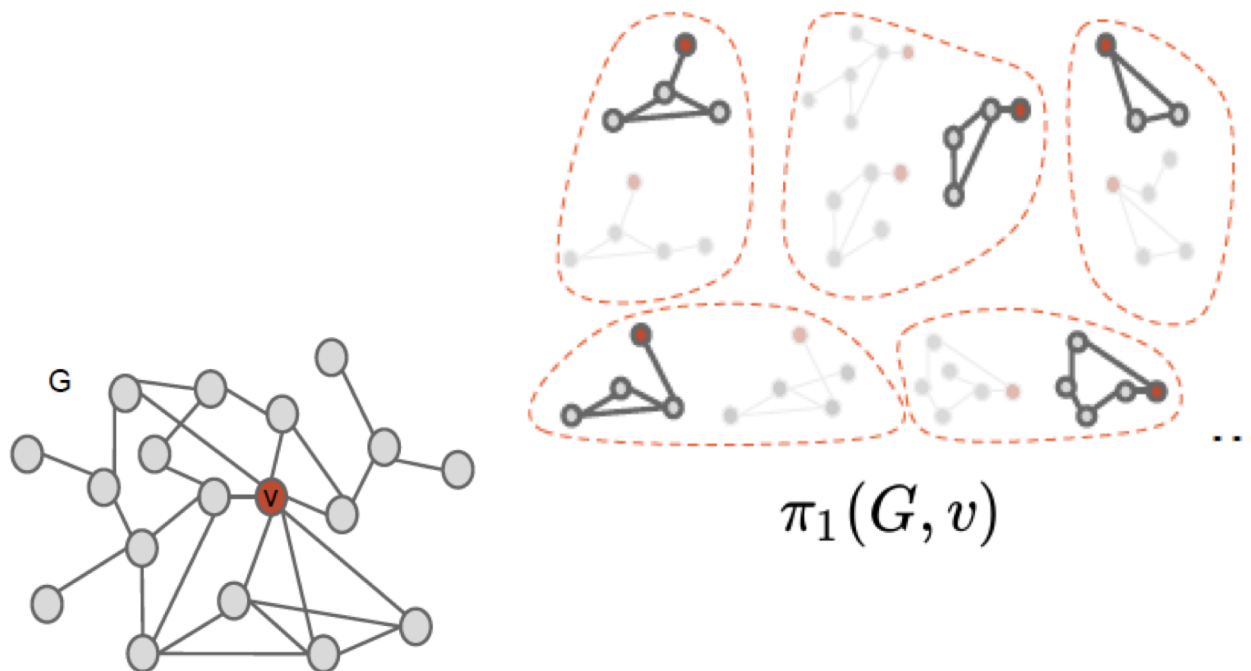
A representative from each equivalence class

- Retain the **shortest closed walk** in each subset



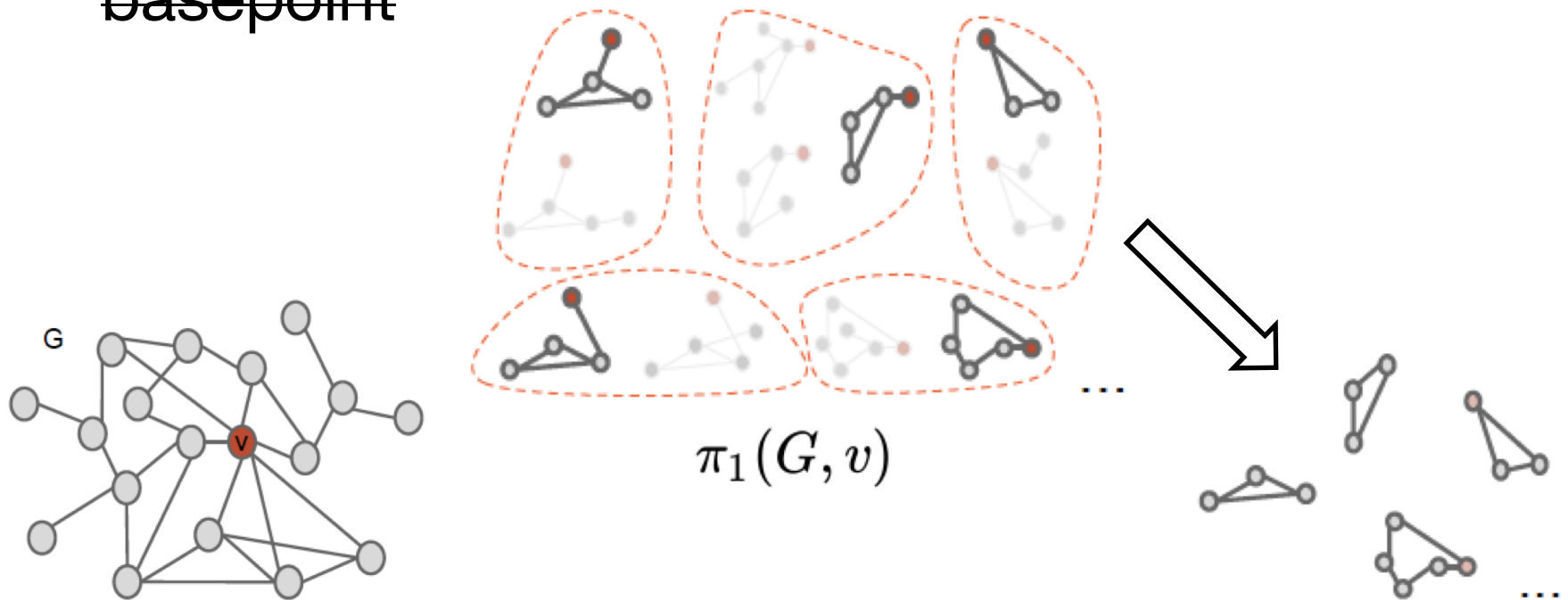
The fundamental group

- The set of representatives is the **fundamental group** of G with basepoint v – a.k.a. $\pi_1(G, v)$



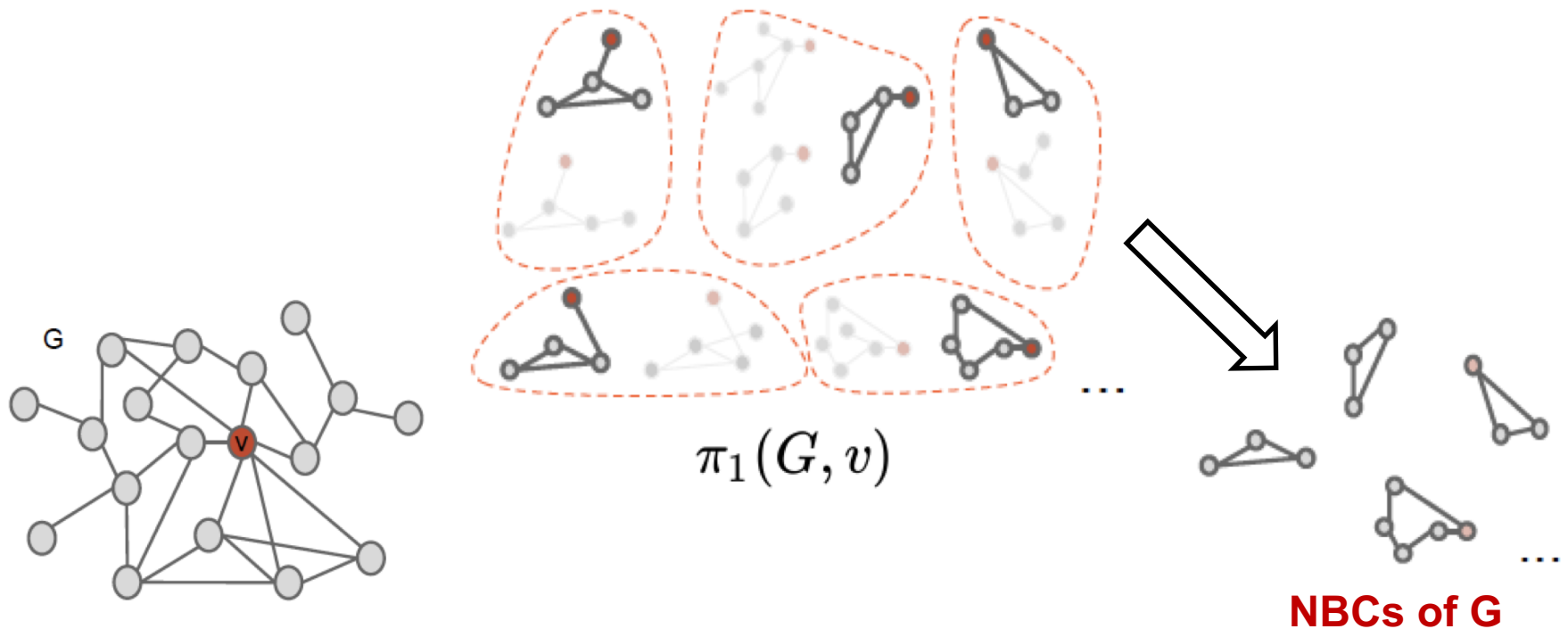
Modifying equivalence of closed walks

- Closed walks are equivalent if they differ by tree-like parts that don't go through the basepoint



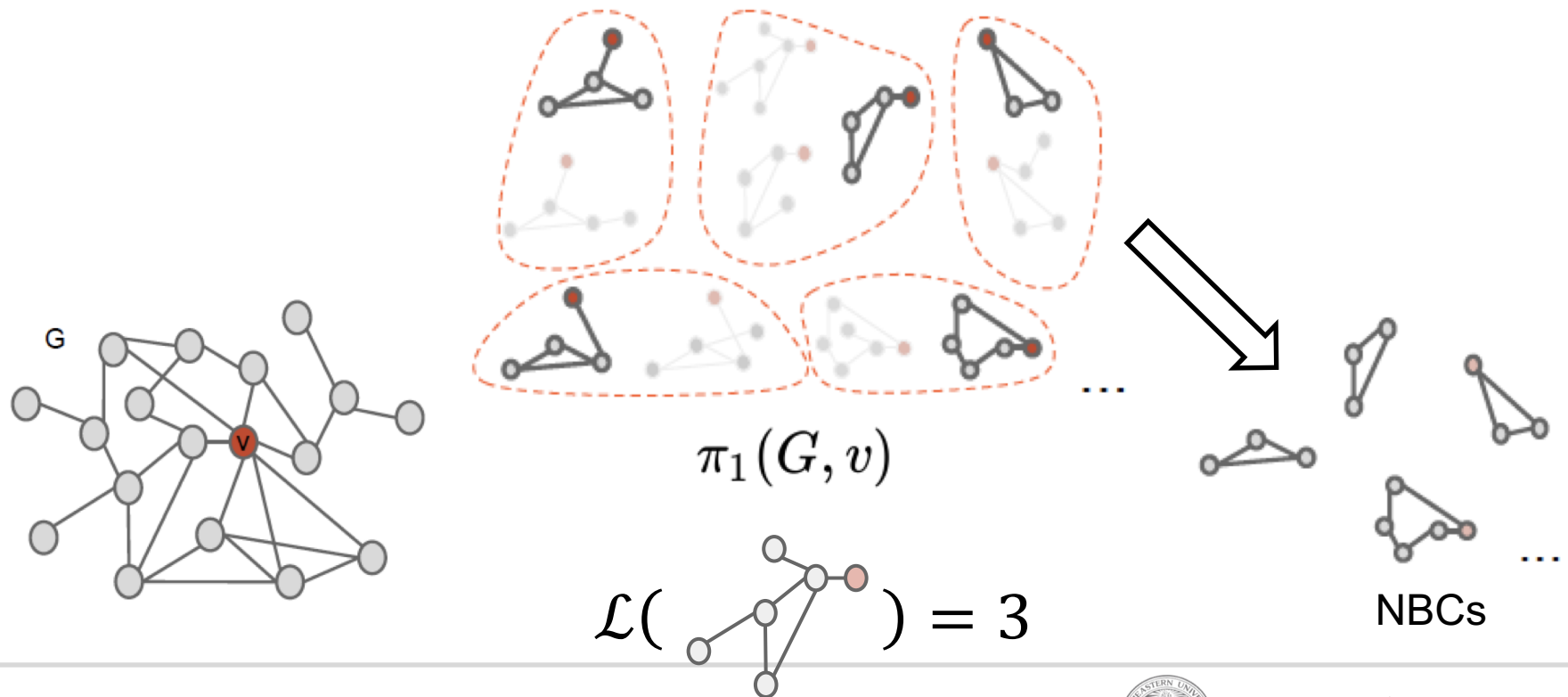
Non-backtracking cycles

- Under this new equivalence definition, we get the set of non-backtracking cycles (NBCs) of G



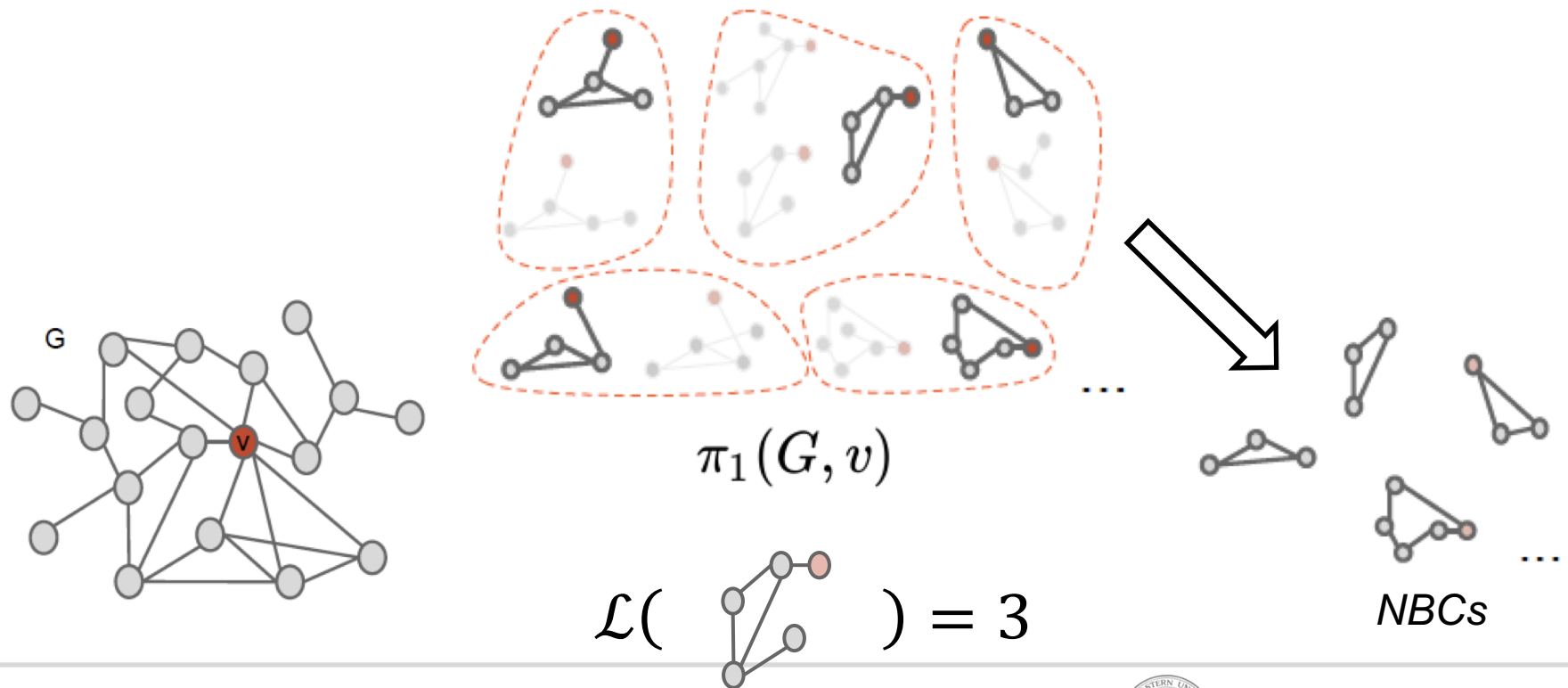
Back to the length spectrum

- The length spectrum, \mathcal{L} , assigns to each closed walk the length of its shaved version



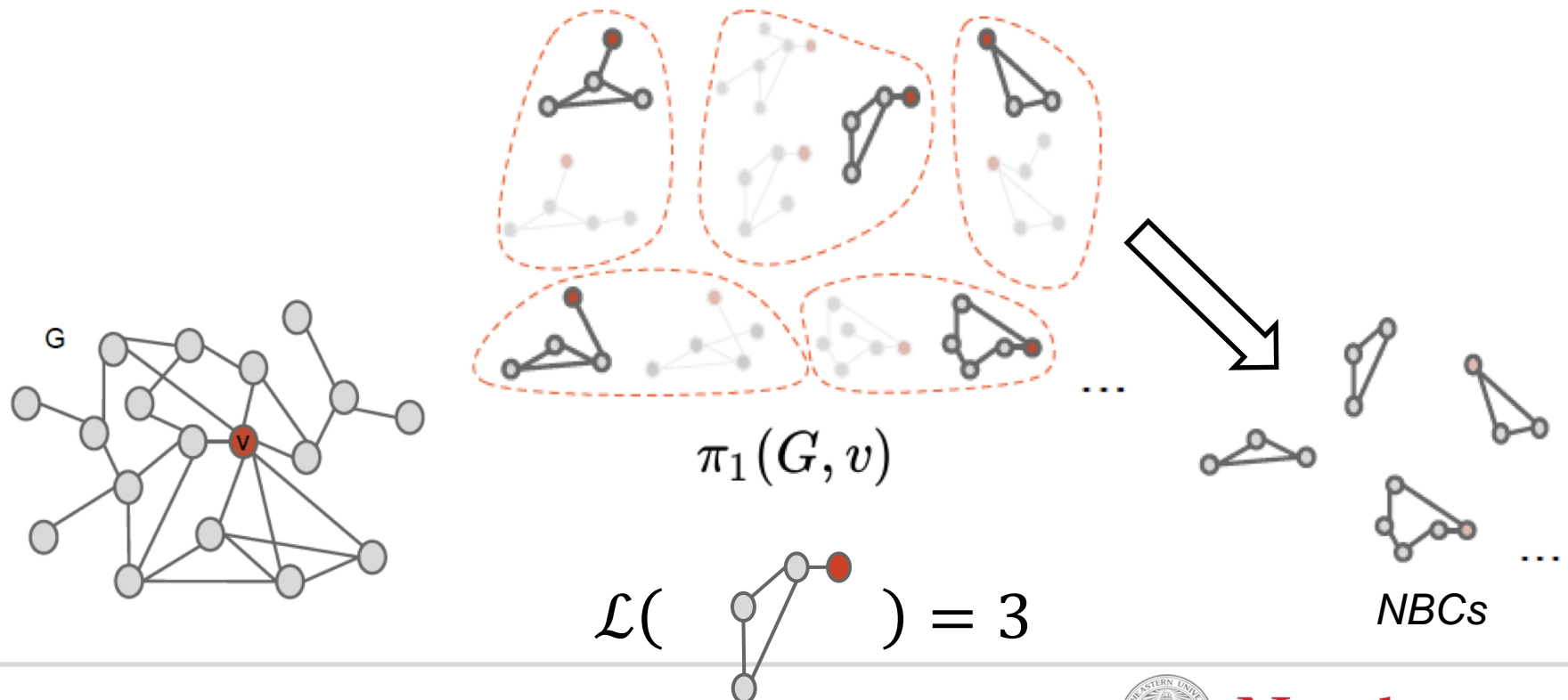
Back to the length spectrum

- The length spectrum, \mathcal{L} , assigns to each closed walk the length of its shaved version



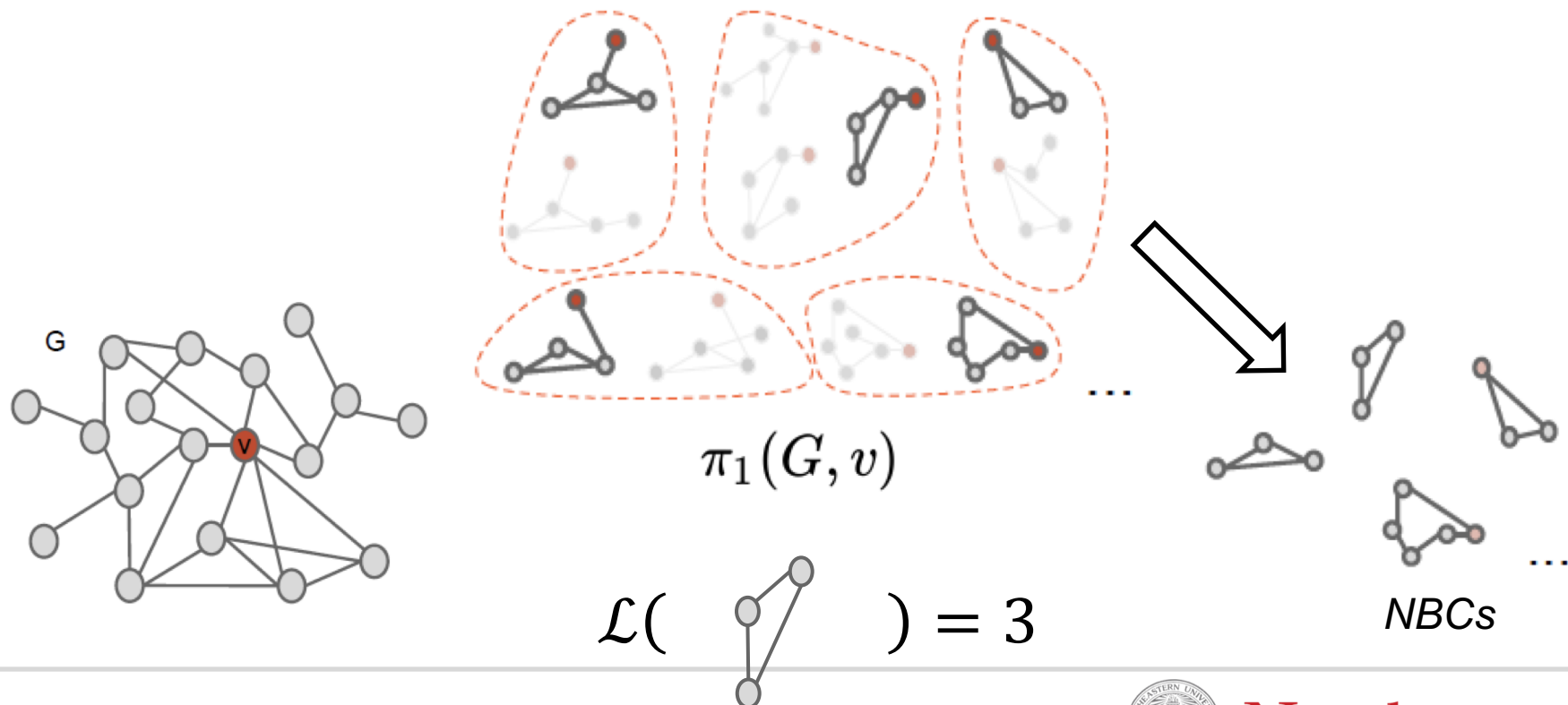
Back to the length spectrum

- The length spectrum, \mathcal{L} , assigns to each closed walk the length of its shaved version



Back to the length spectrum

- The length spectrum, \mathcal{L} , assigns to each closed walk the length of its shaved version



How can we measure distance between graphs with the length spectrum?

$$d(G, H) = d(\mathcal{L}_G, \mathcal{L}_H)$$

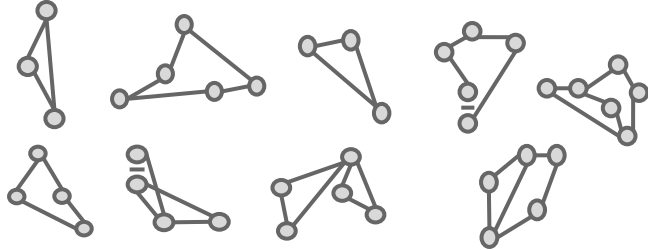
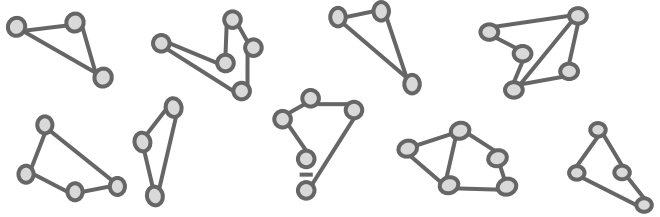
How can we measure distance between graphs with the length spectrum?

$$d(G, H) = d(\mathcal{L}_G, \mathcal{L}_H)$$

Two assumptions	Two problems	Two solutions
$G \rightarrow \mathcal{L}_G$	How to compute?	Image instead of domain
$d(\mathcal{L}_G, \mathcal{L}_H)$	How to compare?	Partition the image

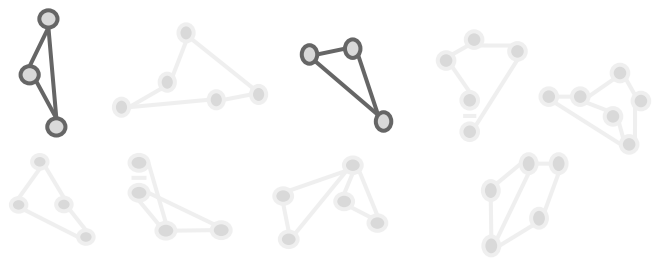
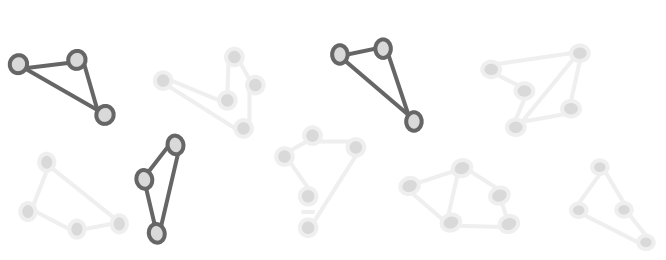
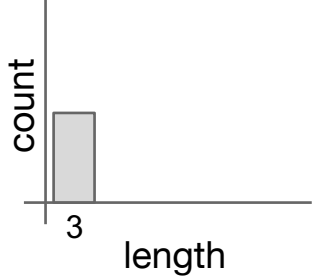
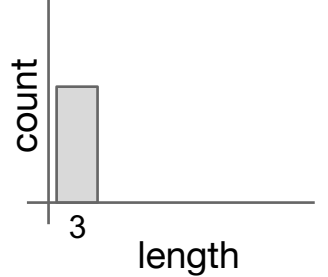
How can we measure distance between graphs with the length spectrum?

Partition the image

	G	H
Domain		
Image		

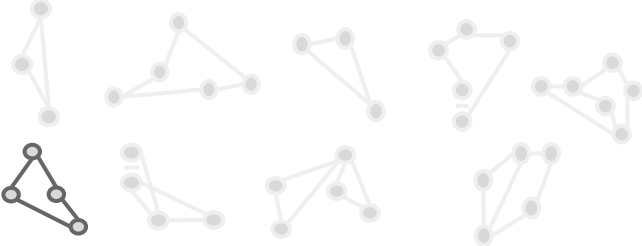

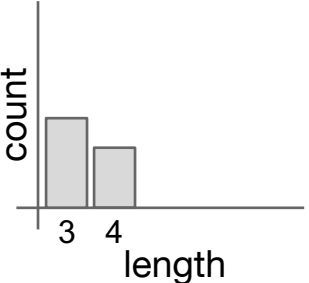
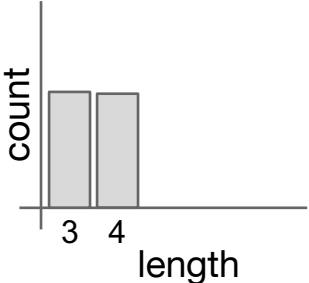
How can we measure distance between graphs with the length spectrum?

Partition the image

	G	H
Domain		
Image		

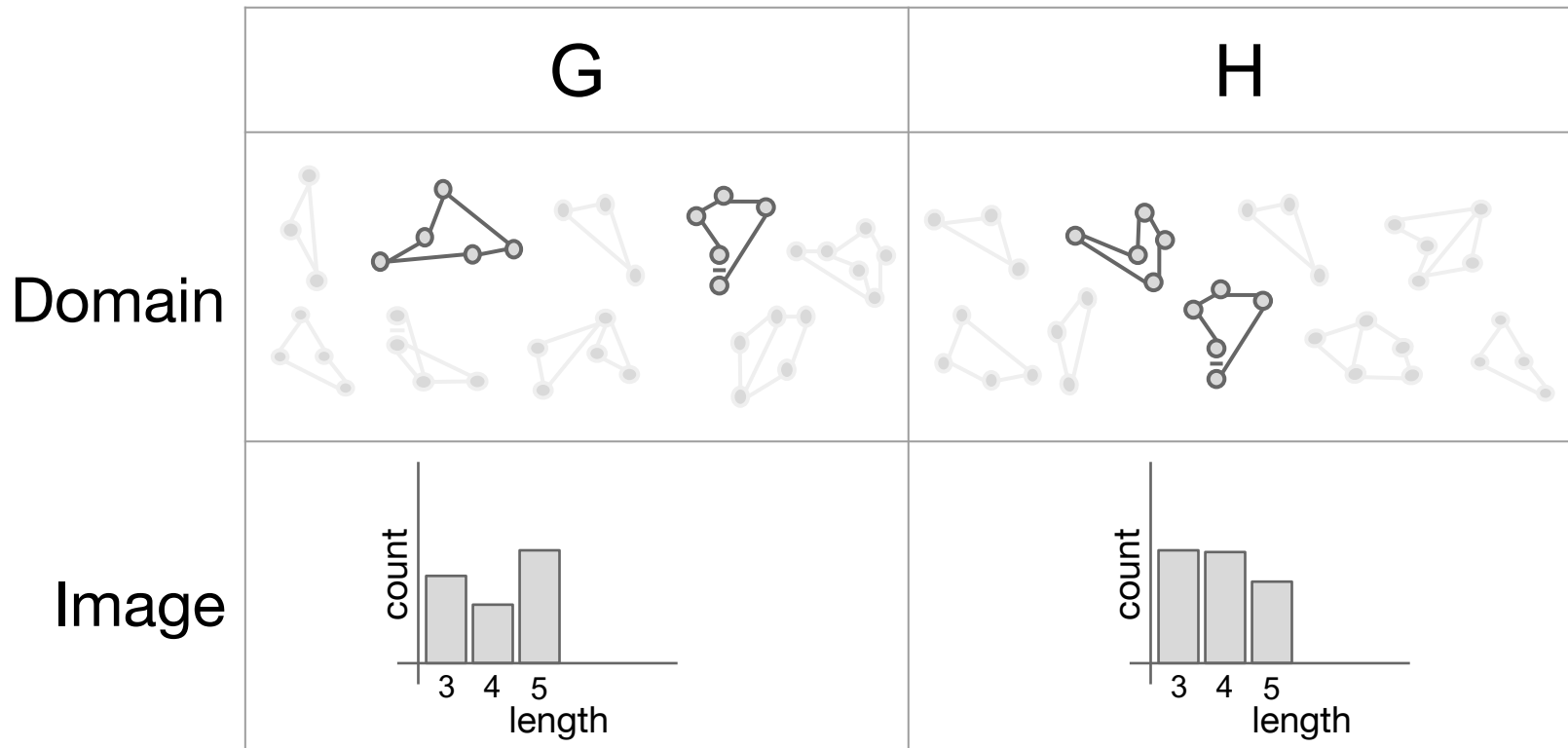
How can we measure distance between graphs with the length spectrum?

Partition the image

	G	H
Domain		
Image		

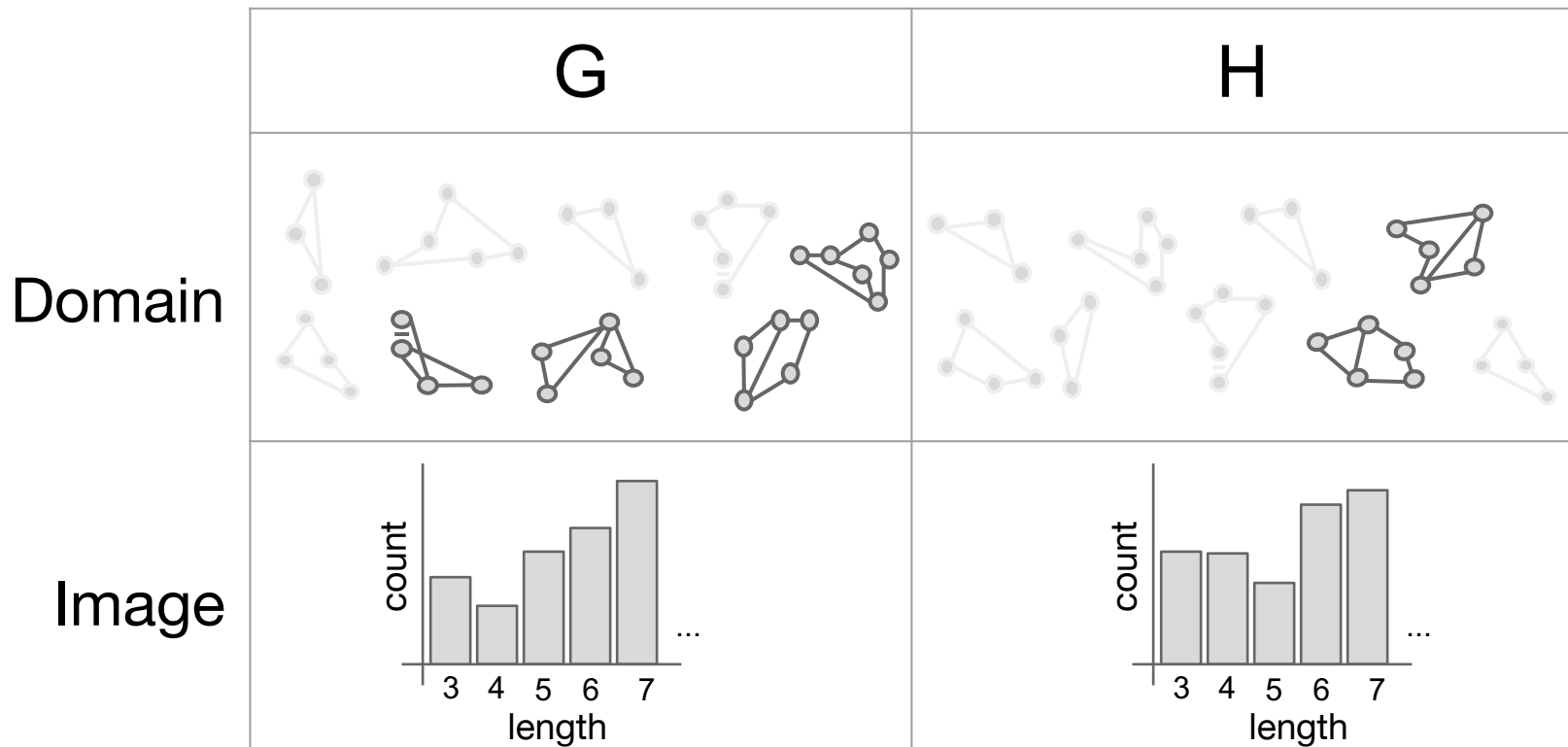
How can we measure distance between graphs with the length spectrum?

Partition the image



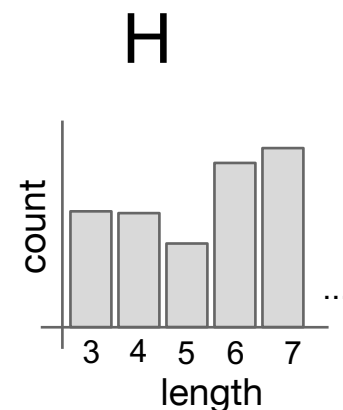
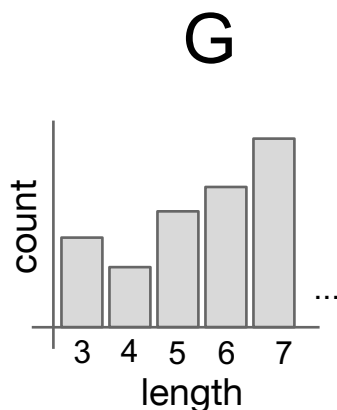
How can we measure distance between graphs with the length spectrum?

Partition the image



From length spectrum to histogram of NBCs

- How should we compare these two histograms?
- Observe the height of each bar is the number of NBCs of a certain length
- We can compute this using the non-backtracking matrix



Outline

Part 1.

Graphs as metric spaces

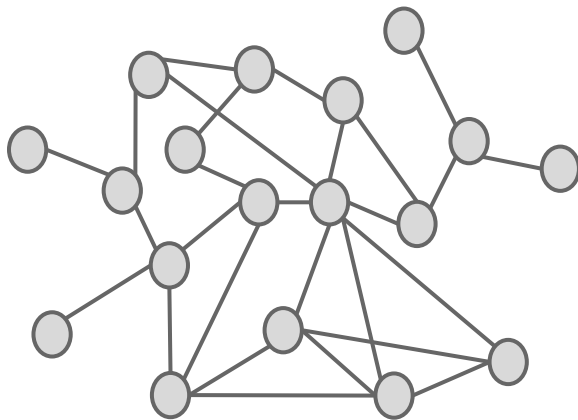
- The length spectrum
- Modifying the length spectrum
- Detour: Non-backtracking matrix (NBM)
- Graph distance
- Properties & examples

Part 2.

Geometry of graph embeddings

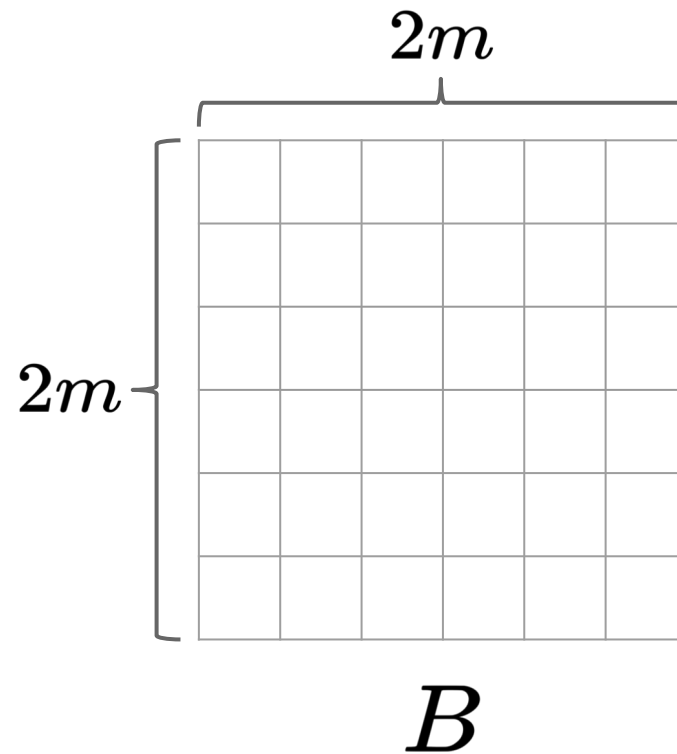
- Vectorization vs. pattern recognition
- Literature review
- Edge embedding with NBM
- Examples

Detour: Non-Backtracking matrix B

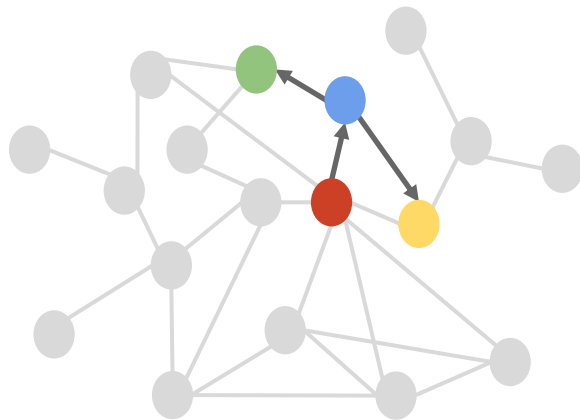


$$G = (V, E)$$

$$|E| = m$$

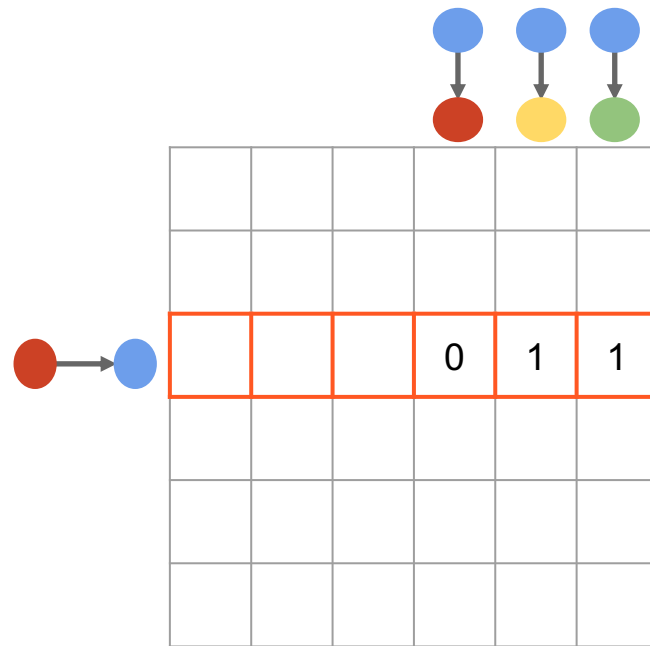


Detour: non-backtracking matrix B



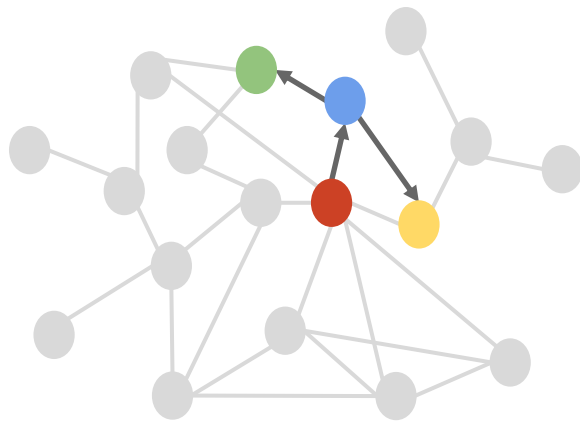
$$G = (V, E)$$

$$|E| = m$$



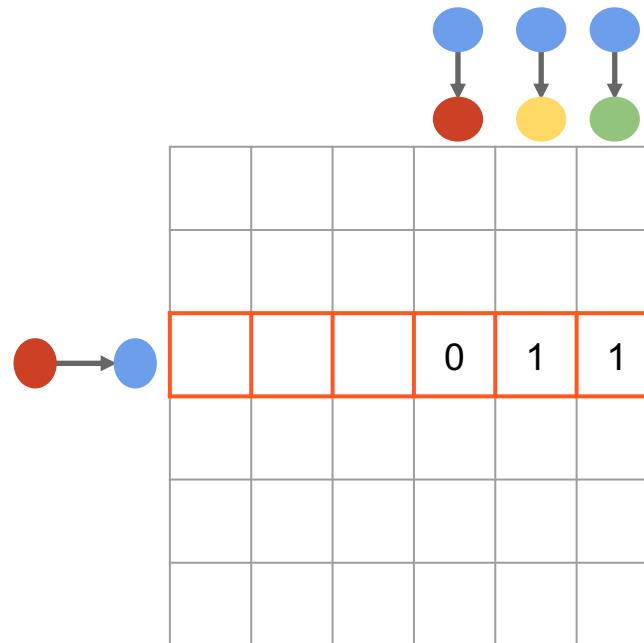
B

Detour: non-backtracking matrix B



$$G = (V, E)$$

$$|E| = m$$



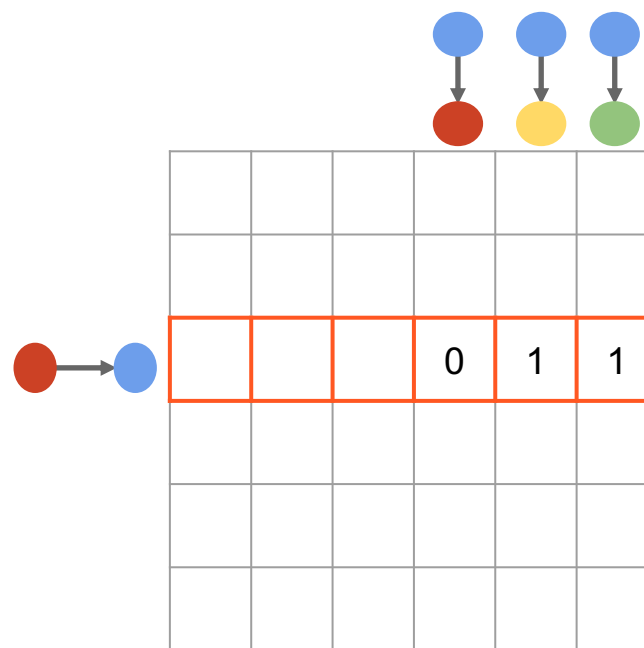
$$B$$

$$B_{k \rightarrow l, u \rightarrow v} = \delta_{vk} (1 - \delta_{ul})$$

$$\delta_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

Detour: non-backtracking matrix B

- Similar to an adjacency matrix of the set of directed edges
- Entries of the powers store the number of non-backtracking walks
- $B_{e_1 e_2}^k = \#$ of non-backtracking walks starting at e_1 and ending at e_2

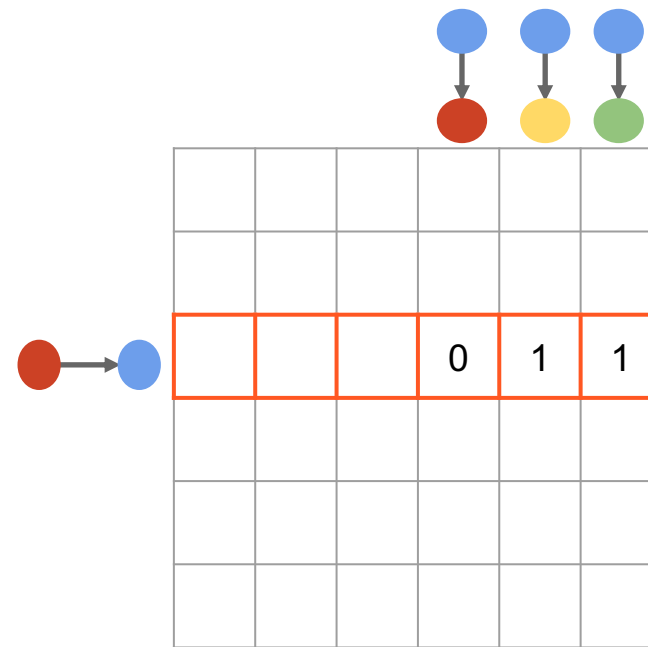


B

$$B_{k \rightarrow l, u \rightarrow v} = \delta_{vk}(1 - \delta_{ul})$$

Detour: non-backtracking matrix B

- Similar to an adjacency matrix of the set of directed edges
- Entries of the powers store the number of non-backtracking walks
- $\sum_i B_{e_i e_i}^k = \#$ of non-backtracking cycles (NBCs) of length k

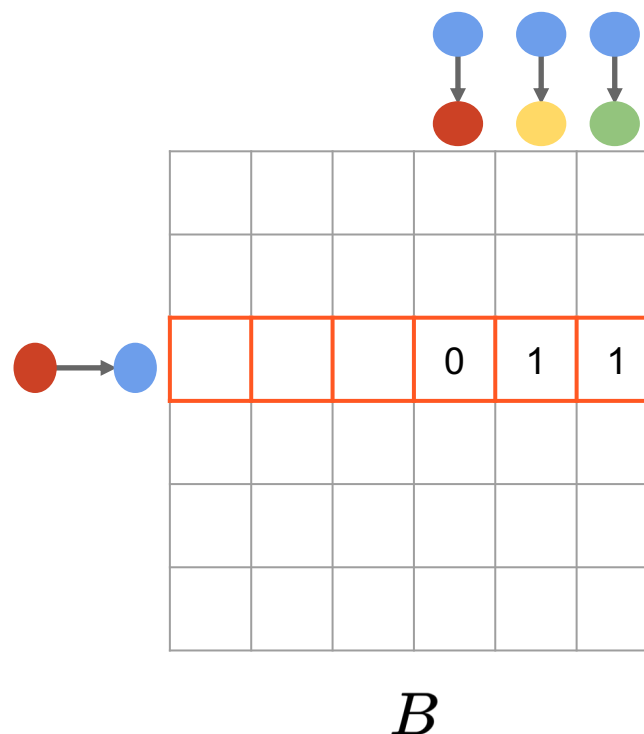


B

$$B_{k \rightarrow l, u \rightarrow v} = \delta_{vk}(1 - \delta_{ul})$$

Detour: non-backtracking matrix B

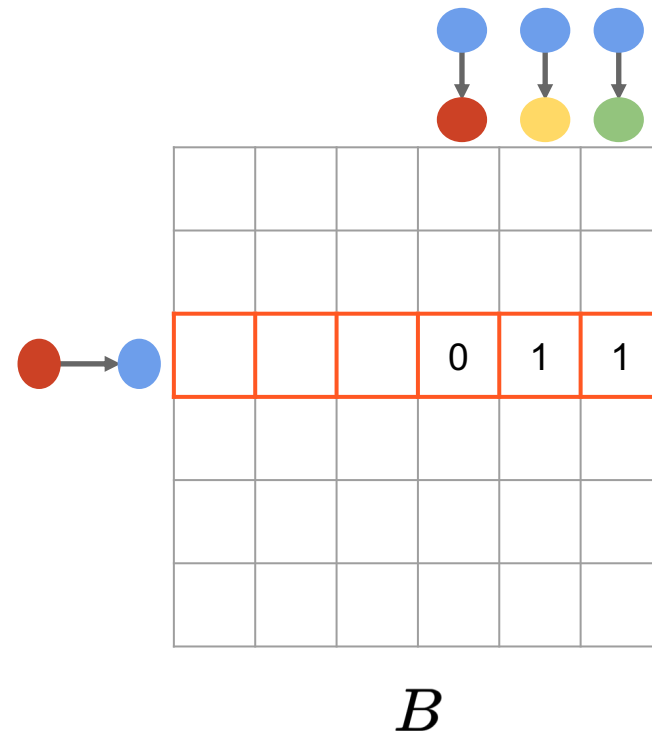
- Similar to an adjacency matrix of the set of directed edges
- Entries of the powers store the number of non-backtracking walks
- $\text{tr}(B^k) = \#$ of NBCs of length k



$$B_{k \rightarrow l, u \rightarrow v} = \delta_{vk}(1 - \delta_{ul})$$

Detour: non-backtracking matrix B

- Similar to an adjacency matrix of the set of directed edges
- Entries of the powers store the number of non-backtracking walks
- $\sum_i \lambda_i^k = \#$ of NBCs of length k



$$B_{k \rightarrow l, u \rightarrow v} = \delta_{vk}(1 - \delta_{ul})$$

Computing B

- Given $G = (V, E)$ with $|V| = n$ and $|E| = m$,
define

$$Q_{x,u \rightarrow v} = \delta_{xv}$$

$$P_{x,u \rightarrow v} = \delta_{xu}$$

$$C = P^T Q$$

- Q and P are $n \times 2m$ matrices, so C is a
 $2m \times 2m$ matrix

Computing B

- Given $G = (V, E)$ with $|V| = n$ and $|E| = m$,
define

$$Q_{x,u \rightarrow v} = \delta_{xv}$$

$$P_{x,u \rightarrow v} = \delta_{xu}$$

$$C = P^T Q$$

$$C_{k \rightarrow l, u \rightarrow v} = \delta_{vk}$$

Computing B

$$B_{k \rightarrow l, u \rightarrow v} = \delta_{vk}(1 - \delta_{ul})$$

$$C_{k \rightarrow l, u \rightarrow v} = \delta_{vk}$$

$$C_{u \rightarrow v, k \rightarrow l} = \delta_{ul}$$

Computing B

$$B_{k \rightarrow l, u \rightarrow v} = \delta_{vk}(1 - \delta_{ul})$$

$$C_{k \rightarrow l, u \rightarrow v} = \delta_{vk}$$

$$C_{u \rightarrow v, k \rightarrow l} = \delta_{ul}$$

$$B_{k \rightarrow l, u \rightarrow v} = C_{k \rightarrow l, u \rightarrow v}(1 - C_{u \rightarrow v, k \rightarrow l})$$

Computing B

$$B_{k \rightarrow l, u \rightarrow v} = \delta_{vk}(1 - \delta_{ul})$$

$$C_{k \rightarrow l, u \rightarrow v} = \delta_{vk}$$

$$C_{u \rightarrow v, k \rightarrow l} = \delta_{ul}$$

$$B_{k \rightarrow l, u \rightarrow v} = C_{k \rightarrow l, u \rightarrow v}(1 - C_{u \rightarrow v, k \rightarrow l})$$

Algorithm: computeB

Input: a graph G

Output: non-backtracking matrix B of G

```
P, Q ← incidence matrices →  $O(m)$ 
C ←  $P^T \times Q$  →  $O(n\langle k^2 \rangle)$ 
for each positive entry  $C_{k \rightarrow l, u \rightarrow v}$ : →  $O(n\langle k^2 \rangle)$ 
    if  $C_{u \rightarrow v, k \rightarrow l} = 0$ :
         $B_{k \rightarrow l, u \rightarrow v} = 1$ 
```

Runtime
complexity for
computing B
is $O(m+n\langle k^2 \rangle)$

Outline

Part 1.

Graphs as metric spaces

- The length spectrum
- Modifying the length spectrum
- Detour: Non-backtracking matrix (NBM)
- Graph distance
- Properties & examples

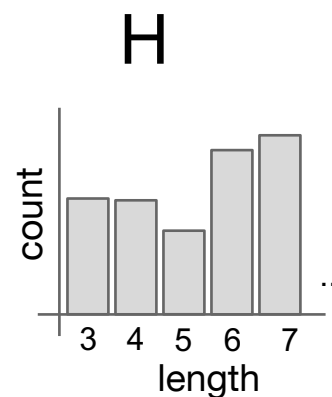
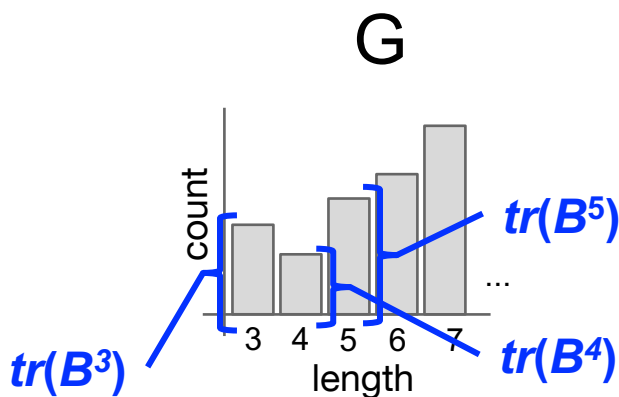
Part 2.

Geometry of graph embeddings

- Vectorization vs. pattern recognition
- Literature review
- Edge embedding with NBM
- Examples

Graph distance

- How should we compare these two histograms?
- Recall that the height of each bar is the number of NBCs of a certain length
- The histograms can be generated using only the eigenvalues of B



The NBD of two graphs

- Given two graphs G and H
- Let λ_k be the k^{th} eigenvalue of G 's non-backtracking matrix
- Let μ_k be the k^{th} eigenvalue of H 's non-backtracking matrix
- Consider the top r eigenvalues of G and H
 - $|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_r|$
 - $|\mu_1| \geq |\mu_2| \geq \dots \geq |\mu_r|$
- Then, the **Non-Backtracking Distance, NBD**, is

$$d(G, H) = \sqrt{\sum_{k=1}^r |\lambda_k - \mu_k|^2}$$

Computing the NBD graph distance

$$d(G, H) = \sqrt{\sum_{k=1}^r |\lambda_k - \mu_k|^2}$$

Algorithm: NBD

Input: two graphs G, H , integer r

Output: real number d , the NBD between G, H

$G', H' \leftarrow \text{shave}(G), \text{shave}(H)$

$B_1, B_2 \leftarrow \text{computeB}(G'), \text{computeB}(H')$

$\lambda, \mu \leftarrow \text{eigs}(B_1, r), \text{eigs}(B_2, r)$

$d \leftarrow \text{Euc}(\lambda, \mu)$

NBD is a pseudo-metric

$$d(G, H) = \sqrt{\sum_{k=1}^r |\lambda_k - \mu_k|^2}$$

[A1] **Identity:** $d(G, G) = 0$

[A2] **Symmetry:** $d(G, H) = d(H, G)$

[A3] **Triangle Inequality:** $d(G, H) \leq d(G, F) + d(F, H)$

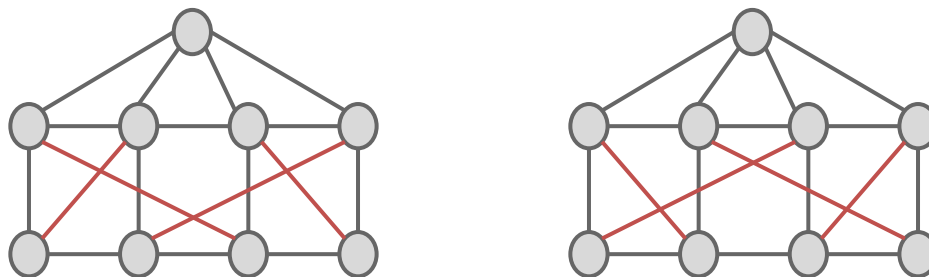
~~[A4]~~ **Id. of indiscernibles:** $d(G, H) = 0 \implies G = H$

[A5] **Divergence*:** $d(K_n, \bar{K}_n) \rightarrow \infty$ as $n \rightarrow \infty$

Koutra, Danai, et al. “DeltaCon: A Principled Massive-Graph Similarity Function.” SIAM SDM 2013.

Cospectrality

- NBD does not satisfy [A4] because of cospectrality: sometimes different graphs have the same eigenvalues
- This can occur w.r.t. the adjacency matrix, the Laplacian, or the non-backtracking matrix



Smallest cospectral graphs w.r.t. the non-backtracking matrix

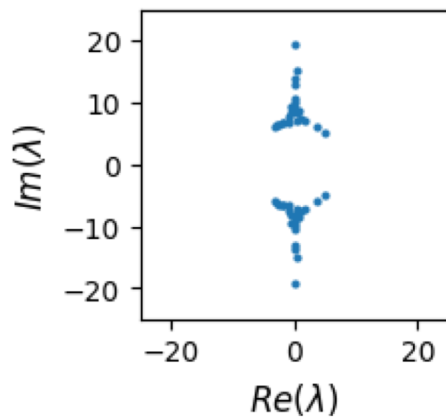
Cospectrality

- **Almost all trees are cospectral.**
 - Schwenk, A. J., Almost all trees are cospectral. In New Directions in the Theory of Graphs (Proc. Third Ann Arbor Conf), pp. 275-307.
- **Open problem:** How many graphs are cospectral?
 - Godsil, C. D., McKay, B. D. Constructing cospectral graphs. Aequationes Math. 25 (1982), no. 2-3, 257–268.
- **Conjecture:** The number of graphs that are cospectral goes to 0 as the number of nodes goes to infinity.
 - Durfee, C., Martin, K., Distinguishing graphs with zeta functions and generalized spectra. Linear Algebra Appl. 481 (2015), 54–82.

Properties of B's spectrum: hubs

Configuration Model with power law deg. dist.

$$N = 10k, \quad \langle k \rangle = 10, \quad \gamma = 2.1$$

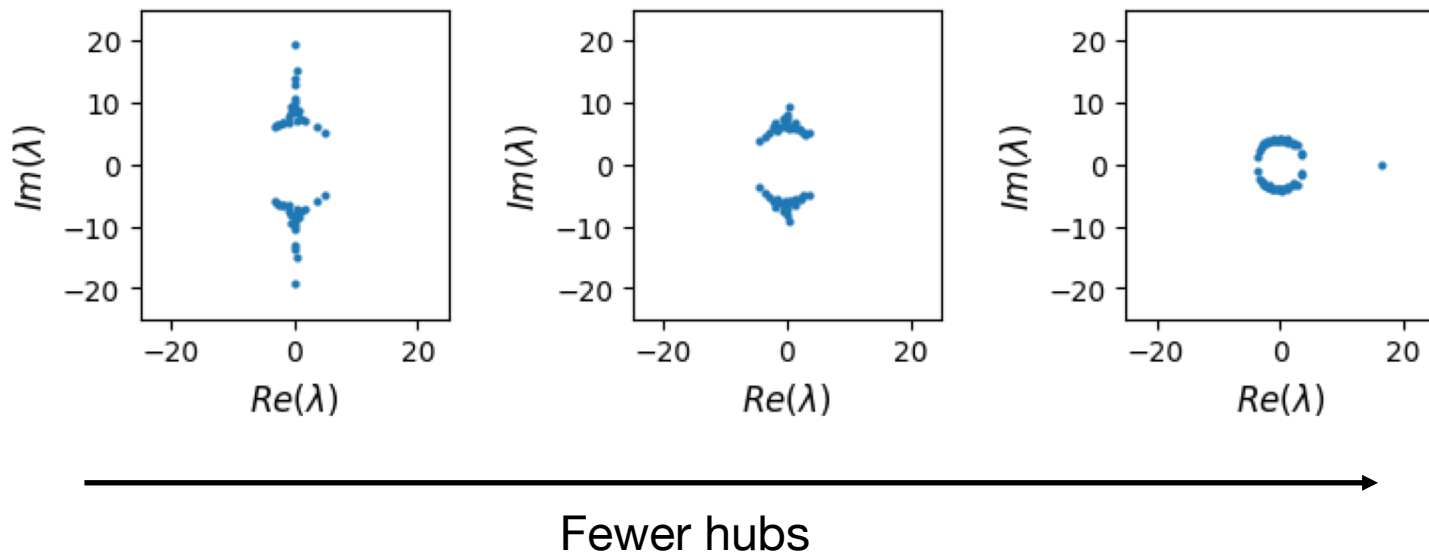


Fewer hubs

Properties of B's spectrum: hubs

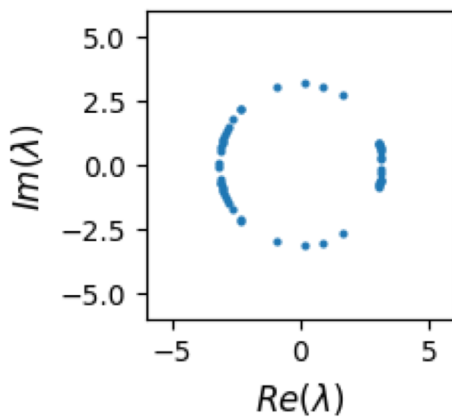
Configuration Model with power law deg. dist.

$N = 10k$, $\langle k \rangle = 10$, $\gamma = 2.1$



Properties of B's spectrum: triangles

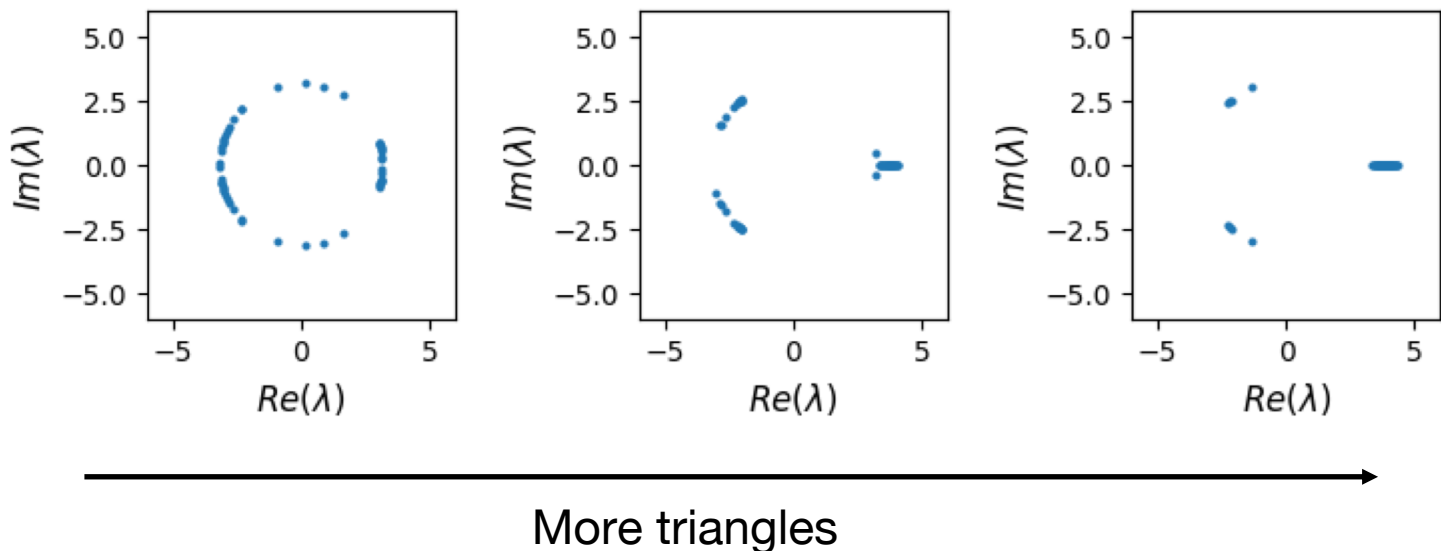
Erdos-Renyi Graph
 $N = 10k$, $\langle k \rangle = 10$, $\gamma = 2.1$



More triangles

Properties of B's spectrum: triangles

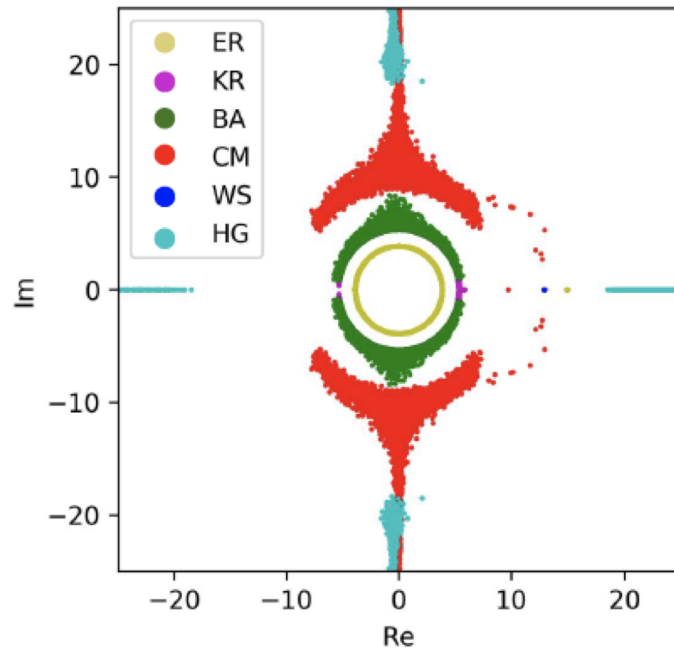
Erdos-Renyi Graph
 $N = 10k$, $\langle k \rangle = 10$, $\gamma = 2.1$



Fine tuning B's spectrum

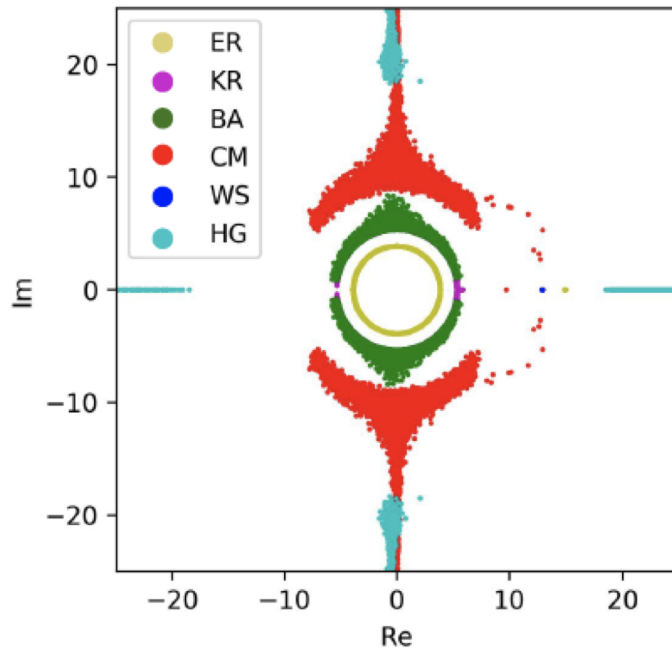
- The original NBD uses true eigenvalues
 - $d(G, H) = \sqrt{\sum_{k=1}^r |\lambda_k - \mu_k|^2}$
 - $\lambda_k = a_k + ib_k$
- NBD can be fine-tuned for triangles
 - $\lambda'_k = \sigma a_k + \frac{ib_k}{\sigma}, \sigma > 1$
- NBD can be fine-tuned for degrees
 - $\lambda''_k = |\lambda_k|^\eta (a_k + ib_k), \eta > 0$

Application: clustering

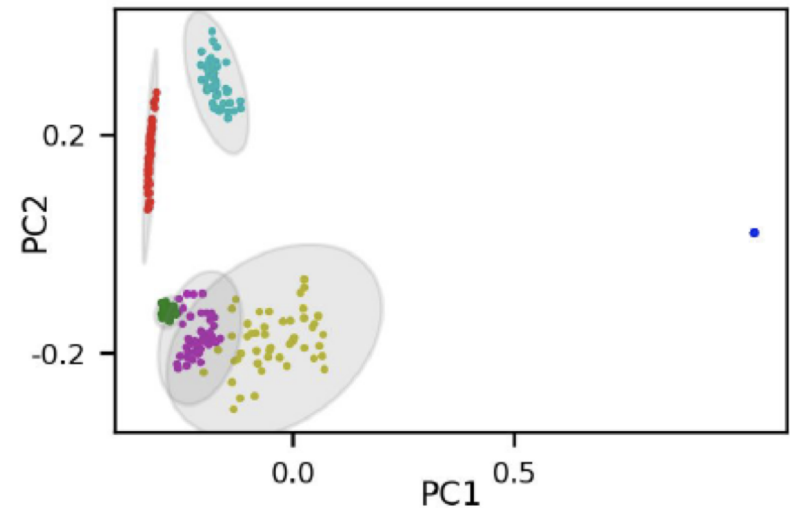


- 1 dot = 1 eigenvalue
- $r = 200$; 50 graphs per model

Application: clustering

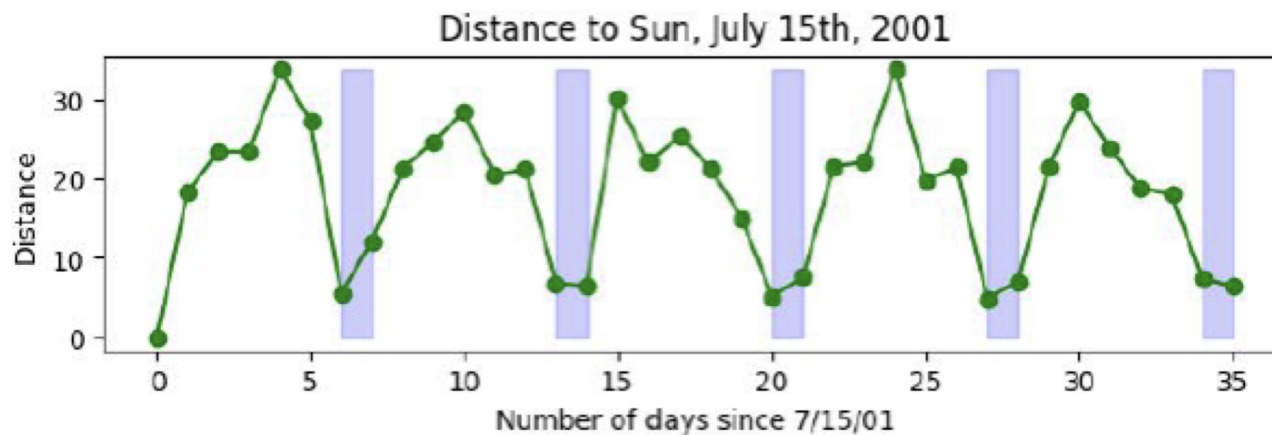


- 1 dot = 1 eigenvalue
- $r = 200$; 50 graphs per model



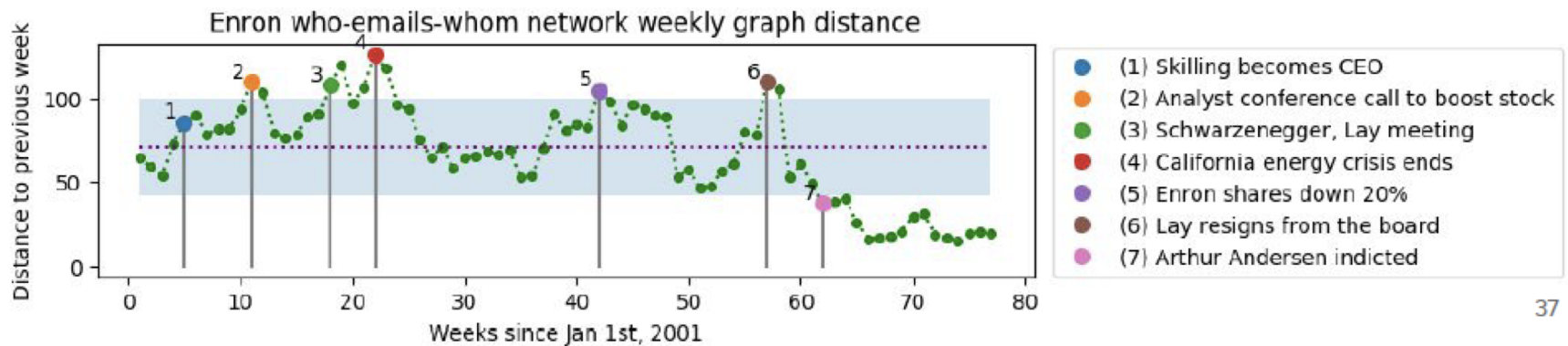
- 1 dot = 1 graph
- 98.66% accuracy
- Errors are when KR gets misclassified as BA

Application: pattern recognition



- Enron email network
- 1 network per day
- Comparing all other days to day 0 (Sunday)

Application: anomaly detection

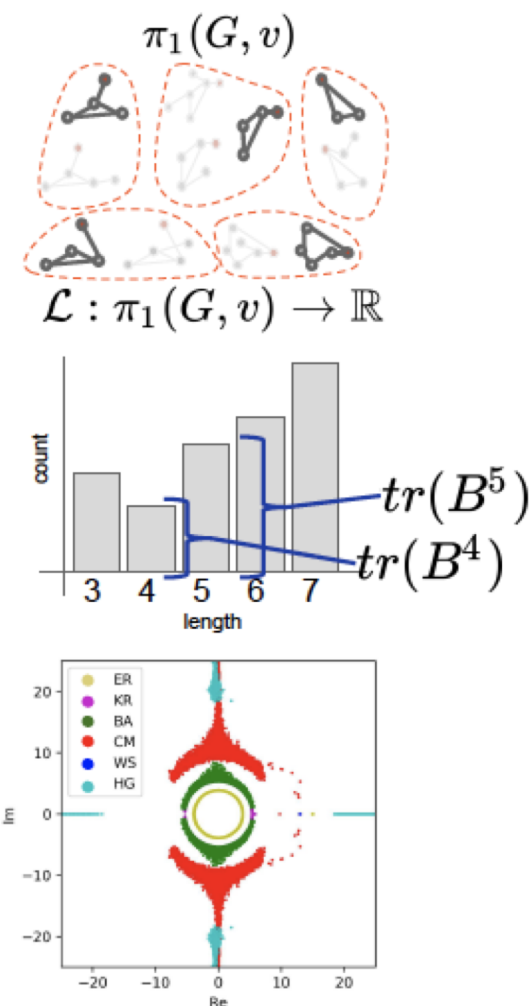


37

- Enron email network
- 1 network per week
- Comparing each week to the previous week

Summary of part 1: graphs as metric spaces

- The length spectrum characterizes the 2-core of a graph
- The eigenvalues of B account for the image of the length spectrum
- NBD is a pseudo-metric, which can be interpreted in terms of triangles & degrees
- Applications: cluster graphs, pattern recognition, anomaly detection
- Paper: [arXiv:1807.09592](https://arxiv.org/abs/1807.09592)
- Code: <https://github.com/leotrs/sunbeam>



Outline

Part 1.

Graphs as metric spaces

- The length spectrum
- Modifying the length spectrum
- Detour: Non-backtracking matrix (NBM)
- Graph distance
- Properties & examples

Part 2.

Geometry of graph embeddings

- Vectorization vs. pattern recognition
- Literature review
- Edge embedding with NBM
- Examples

Vectorization vs. pattern recognition

Vectorization

- Given a graph $G=(V, E)$, find a d -dimensional feature vector for each node or edge
- **Geometry:** similar nodes are **close** to each other
- Useful for link prediction, node classification, *etc.*

Vectorization vs. pattern recognition

Vectorization

- Given a graph $G=(V, E)$, find a d -dimensional feature vector for each node or edge
- **Geometry:** similar nodes are **close** to each other
- Useful for link prediction, node classification, *etc.*

Pattern Recognition

- Given a graph $G=(V, E)$, find a d -dimensional location for each node/edge
- **Geometry:** **shape** reveals relationships between nodes
- Useful for visualization, anomaly detection, *etc.*

Example of pattern recognition

“Perhaps the most famous example is that the embedded representation of the word *queen* can be roughly recovered from the representations of *king*, *man*, and *woman*.”

$$\textit{queen} \approx \textit{king} - \textit{man} + \textit{woman}$$

- Omer Levy, Yoav Goldberg, **Linguistic Regularities in Sparse and Explicit Word Representations**. CoNLL 2014: 171-180.

Literature review

- **Graph embedding** (from graph mining): need a low rank representation of the nodes; they use the distance, but don't use the geometry. See Tutorial T6/44 this afternoon.
- **Isometric embedding**: Given a distance on a graph (e.g. shortest path), find an embedding in Euclidean space that exactly preserves distances.
 - Graham, R. L., Winkler, P. M., *Isometric embeddings of graphs*. PNAS U.S.A. 81 (1984), no. 22, Phys. Sci., 7259–7260.
 - **Theorem**: If G is complete, and has weights on edges that satisfy triangle inequality, then there's a necessary and sufficient characterization for isometric embeddings – namely, a quadratic inequality on linear combinations of edge weights.
 - Deza, M. M., Laurent, M., *Geometry of cuts and metrics*. Algorithms and Combinatorics, 15. Springer, Heidelberg, 2010.

Literature review

- **Embedding with distortion** (a.k.a. **nearly isometric embedding**): an embedding that is “almost” isometric, up to some distortion in distances.
 - Linial, N., London, E., Rabinovich, Y., *The geometry of graphs and some of its algorithmic applications*. Combinatorica 15 (1995), no. 2, 215–245.
- **Simplex geometry**: exact correspondence between the n nodes of a graph and the vertices of an $(n-1)$ dimensional simplex in Euclidean space.
 - Devriendt, K., Piet Van M., *The Simplex Geometry of Graphs*. preprint arXiv:1807.06475 (2018).
- **Graph planarity**: Place nodes on a plane such that edges don’t cross; preserves adjacency but not distance.
 - https://en.wikipedia.org/wiki/Planarity_testing

Literature review table

Name	Definition	Common Approach	Properties	Unique solution?
Graph embedding	Low rank representation of the nodes	Matrix factorization	Distance	No
Isometric embedding	Embedding preserves distances exactly	Randomized algorithm	Distance	No
Nearly isometric embedding	Embedding preserves distances with global distortion	Randomized algorithm	Distance	No
Simplex embedding	Embed nodes based on the eigenvectors of the Laplacian	Eigen decomposition	Vector space (convexity, linear indep., ...)	Yes
Graph planarity	Place nodes on a plane such that edges don't cross	Edge additions	Adjacency (relaxed distance)	No

Edge embedding with the eigenvectors of the non-backtracking matrix

- Problem
 - Given a graph $G = (V, E)$, find a low-dimensional representation for each directed edge, $\{p_e\}_{e \in E}$
- Approach
 - For each directed edge e , assign $p_e = (\vec{v}_1^e, \vec{v}_2^e)$, where \vec{v}_1 and \vec{v}_2 are the first and second eigenvectors of G 's non-backtracking matrix

Why the eigenvectors of the non-backtracking matrix?

- Edge embedding
 - Node embedding by aggregating edges incident to the same node
 - Distinction between source and target nodes
- The first and second eigenvectors are interpretable
 - 1st eigenvector captures edge centrality
 - 2nd eigenvector captures community structure
- Deterministic, not stochastic
 - Always produces the same embedding (save for signs)

Non-backtracking embedding of edges

- Consider a graph $G = (V, E)$
- For each directed edge, there are two corresponding rows in the non-backtracking matrix B , one for each orientation
 - Each eigenvector has two entries for each directed edge
- Let λ_1, λ_2 be the largest two eigenvalues of B
- Let \vec{v}_1, \vec{v}_2 be their corresponding eigenvectors
- For each directed edge e , define $p_e = (\vec{v}_1^e, \vec{v}_2^e)$

Non-backtracking embedding of edges

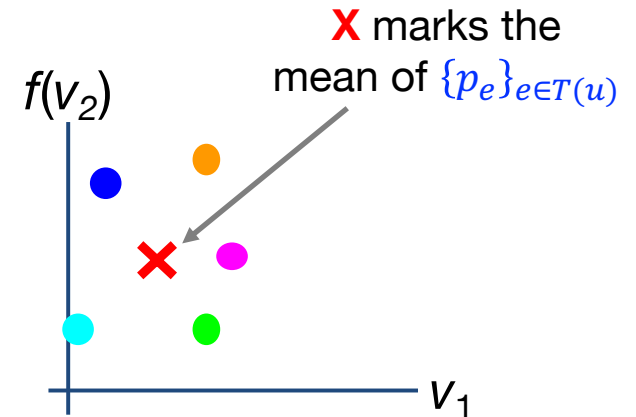
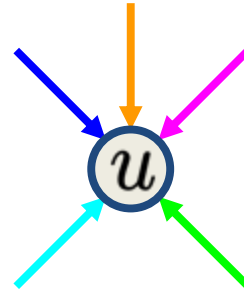
- For each directed edge e , define $p_e = (\vec{v}_1^e, \vec{v}_2^e)$
- **Issue:** \vec{v}_1^e is always a real number, but \vec{v}_2^e may be complex
- **Solution:** Use $f(\vec{v}_2^e) = \text{Re}(\lambda_2)\text{Re}(\vec{v}_2^e) - \text{Im}(\lambda_2)\text{Im}(\vec{v}_2^e)$
 - $f(\vec{v}_2^e)$ is proportional to \vec{v}_2^e when λ_2 is real
 - $f(\vec{v}_2^e)$ is a real linear combination of the real and imaginary parts of \vec{v}_2^e when λ_2 is complex
 - $f(\vec{v}_2^e)$ is always a real number

Non-backtracking embedding of nodes

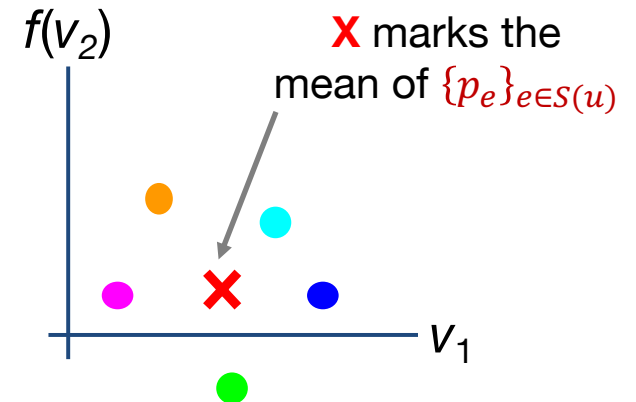
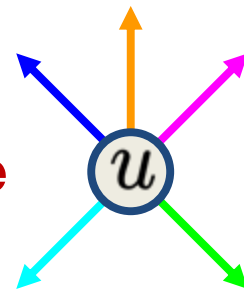
- Given the directed edge embeddings $\{p_e\}_{e \in E}$, we build two distinct node embeddings
- For node u ,
 - $S(u)$ = set of all edges with u as **source**
 - $T(u)$ = set of all edges with u as **target**
- The embedding of u as a **source** is defined as the **mean** of $\{p_e\}_{e \in S(u)}$
- The embedding of u as a **target** is defined as the **mean** of $\{p_e\}_{e \in T(u)}$

From edge to node embedding

Node u as a **target**



Node u as a **source**



Outline

Part 1.

Graphs as metric spaces

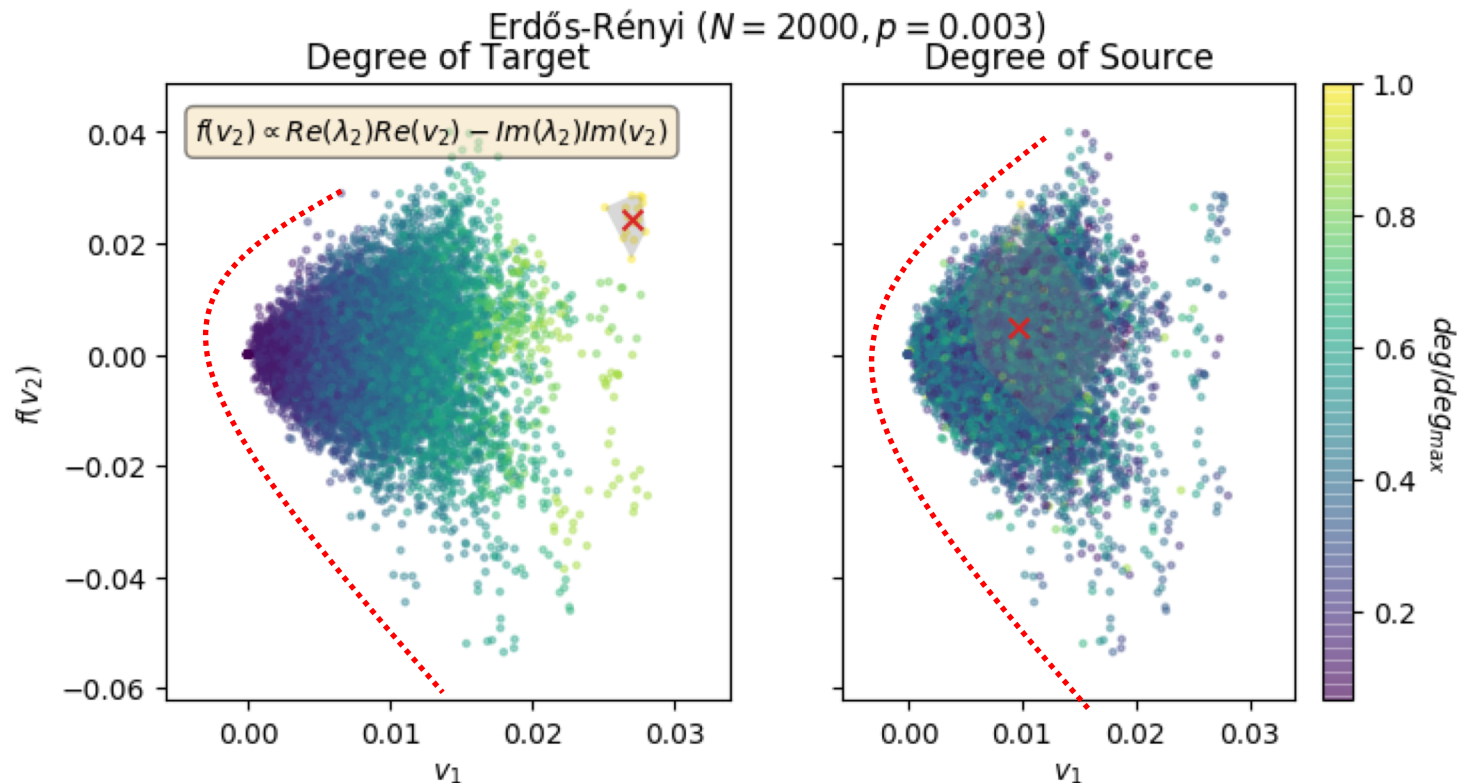
- The length spectrum
- Modifying the length spectrum
- Detour: Non-backtracking matrix (NBM)
- Graph distance
- Properties & examples

Part 2.

Geometry of graph embeddings

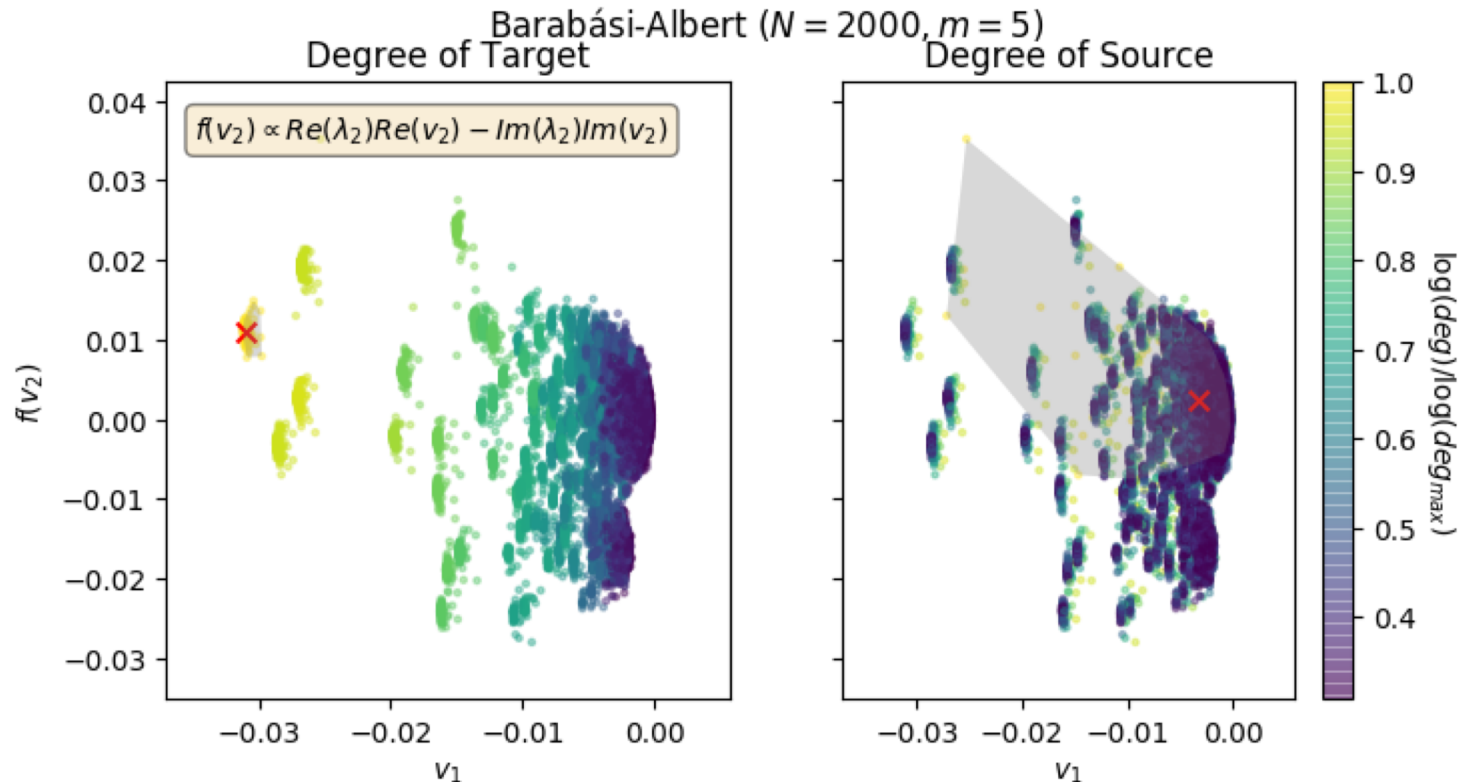
- Vectorization vs. pattern recognition
- Literature review
- Edge embedding with NBM
- Examples

Erdos-Renyi graphs



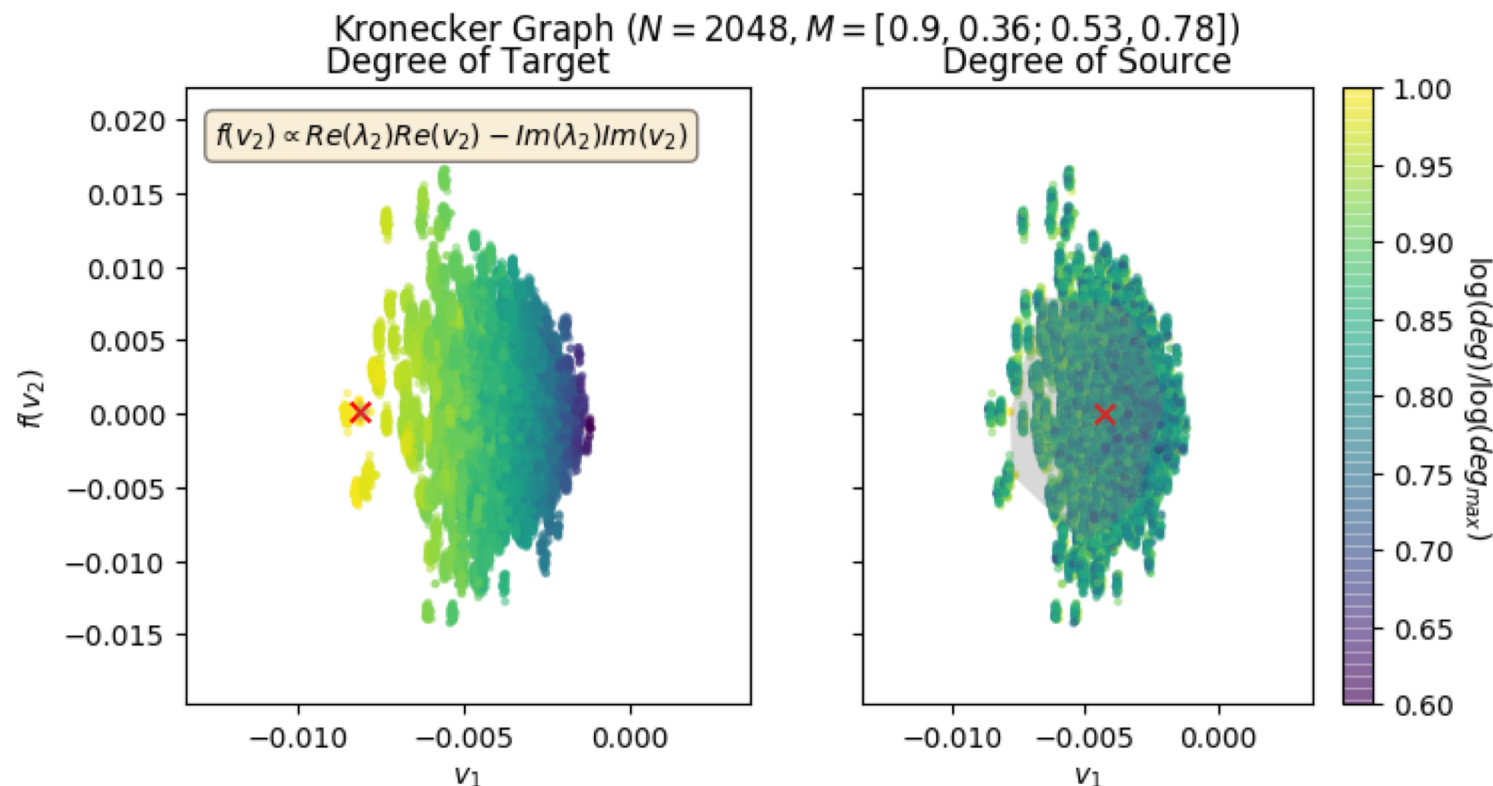
The overall structure is a noisy point cloud,
but the dots seem to have a fan shape. Why?

Barabasi-Albert graphs



- Each cluster is made up of all edges that are incident to the same node.
- **Why should incident edges cluster together in this space?**
- Every cluster has dark dots to the left and light dots to the right.
- **Why does every cluster have the same internal structure?**

Kronecker graphs



- The overall structure is a noisy cloud point but clusters are present.
- It looks like a mix between the ER and BA.
- **In what way is KR “a mix” between ER and BA?**

Summary of part 2: geometry of graph embeddings

- The non-backtracking matrix allows for deterministic embedding of nodes and edges
- These embeddings can be interpreted in terms of their shape
- Useful for pattern recognition applications such as visualization, anomaly detection, *etc.*



Summary

- **Geometric data analysis** of graphs:
graphs as metric spaces, metric spaces of graphs, metric embedding spaces
- **Topological data analysis** of graphs:
the length spectrum, the non-backtracking matrix, its eigenvalues and eigenvectors

General Outline for T21: Graph Metric Spaces

8:00 - 9:00	Jose Bento	Part 1: Introduction to Graph Distances
9:00 - 9:30	Stratis Ioannidis	Part 2a: A Family of Tractable Graph Metrics
9:30 - 10:00	Coffee break	ICC Capital Suite Foyer (Level 3)
10:00 - 10:30	Stratis Ioannidis	Part 2b: A Family of Tractable Graph Metrics
10:30 - 11:30	Tina Eliassi-Rad	Part 3: Non-backtracking Matrix, Graph Distance, and Metric Embedding
11:30 - 12:00	Q&A	

Slides available at <https://neu-spiral.github.io/GraphMetricSpaces/>





Grants IIS-1741197 & IIS-1741129



Northeastern



BOSTON COLLEGE

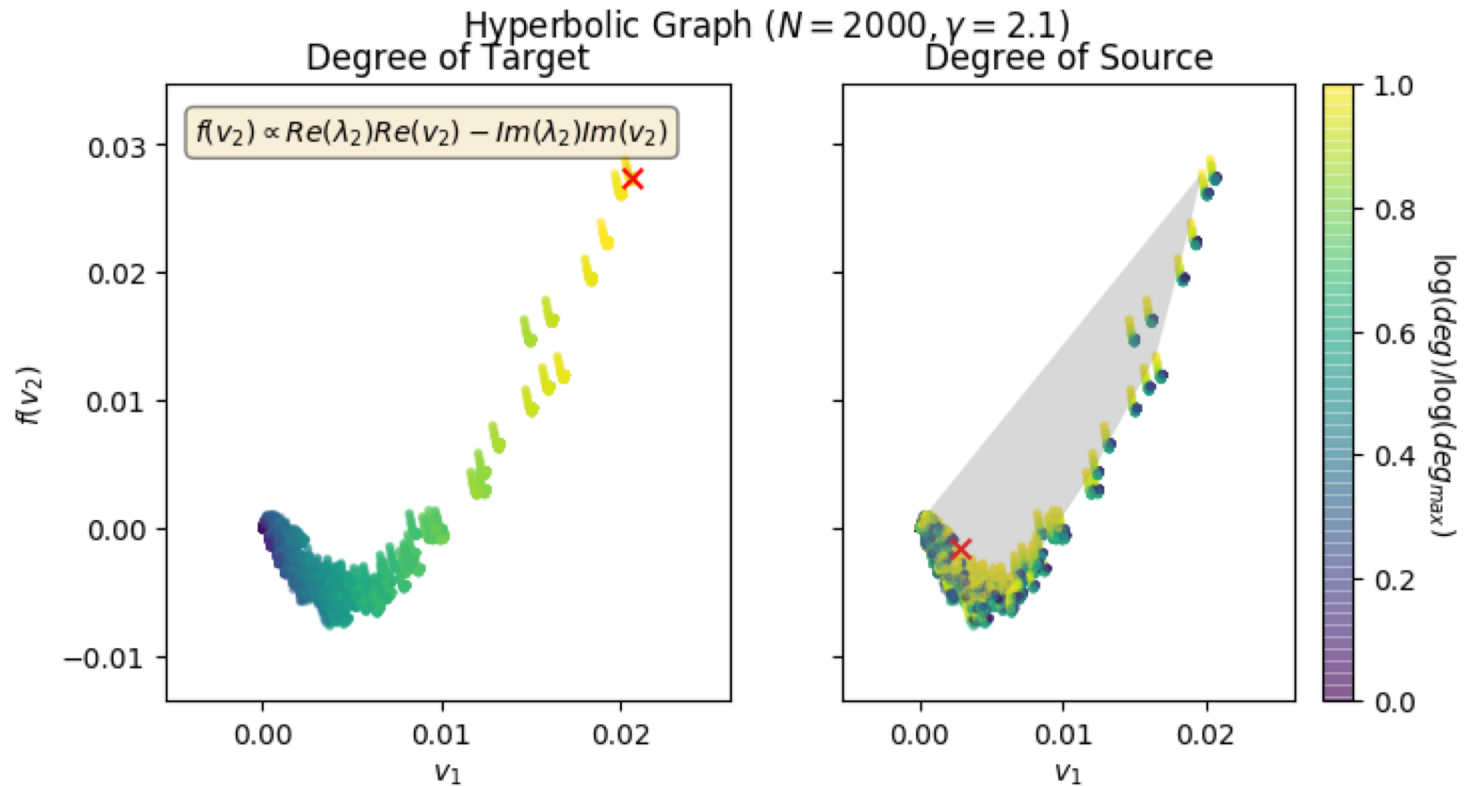
Thank you!

Graph Metric Spaces

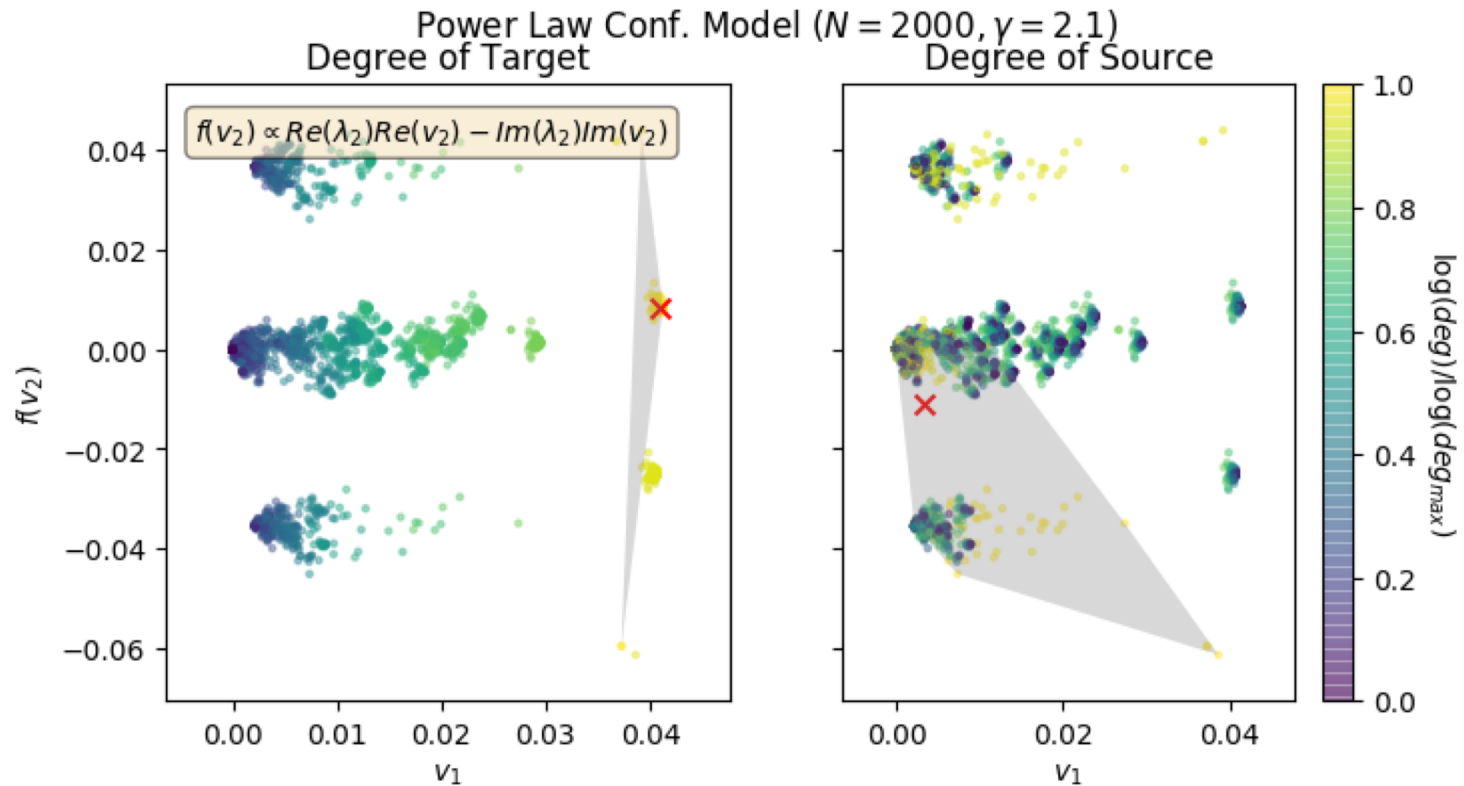
Tutorial @ KDD 2018

J. Bento, T. Eliassi-Rad / L. Torres, and S. Ioannidis

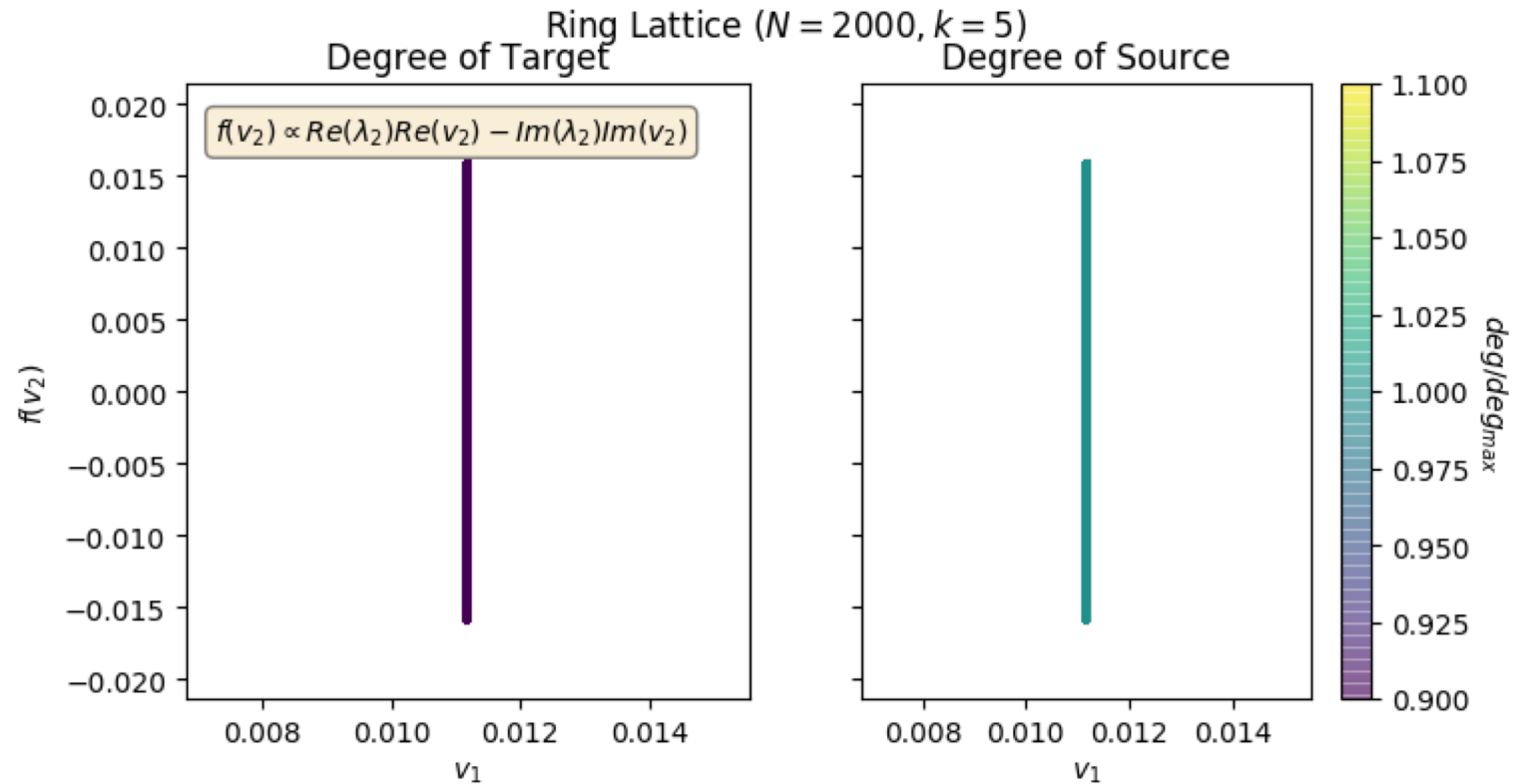
Hyperbolic graphs



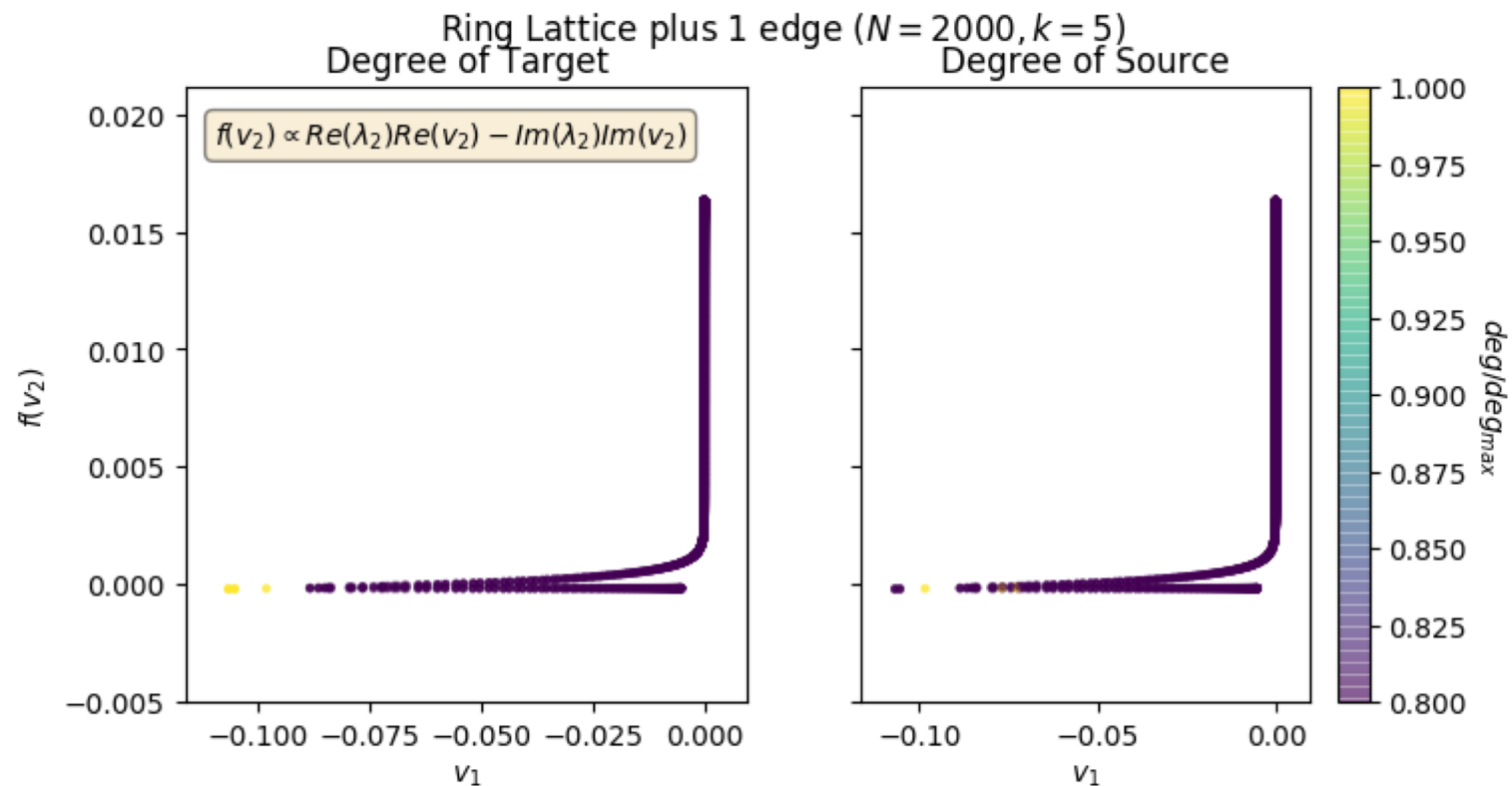
Configuration model graphs



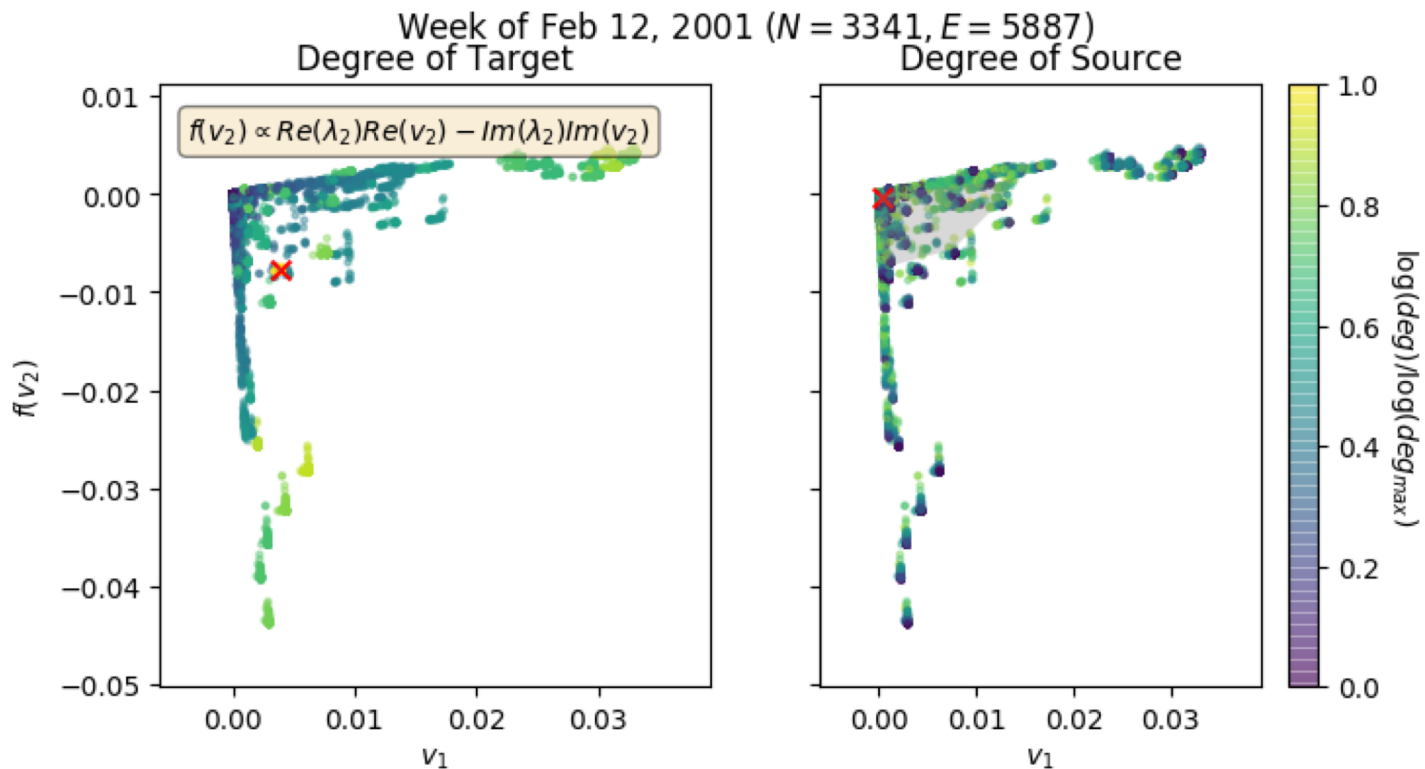
Ring lattice graphs



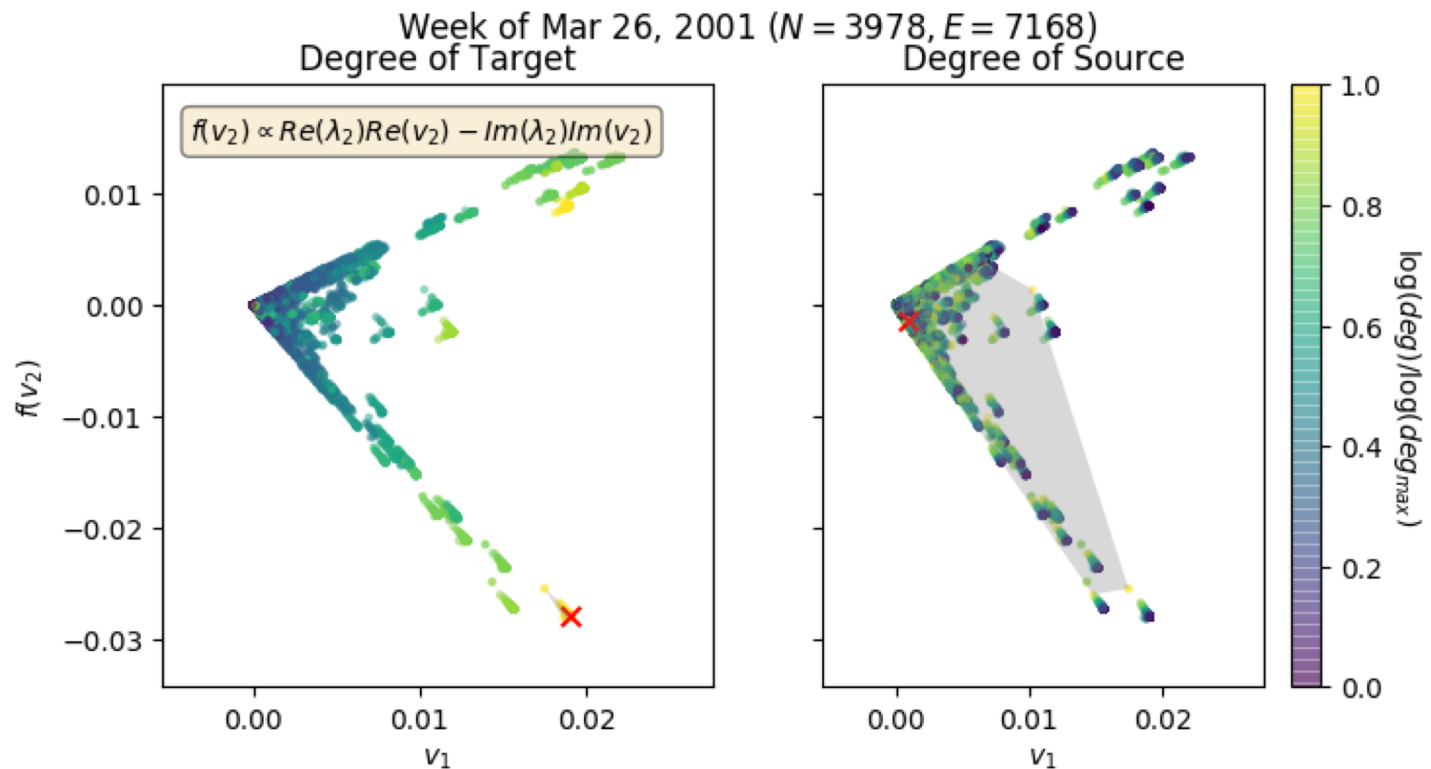
Ring lattice graphs



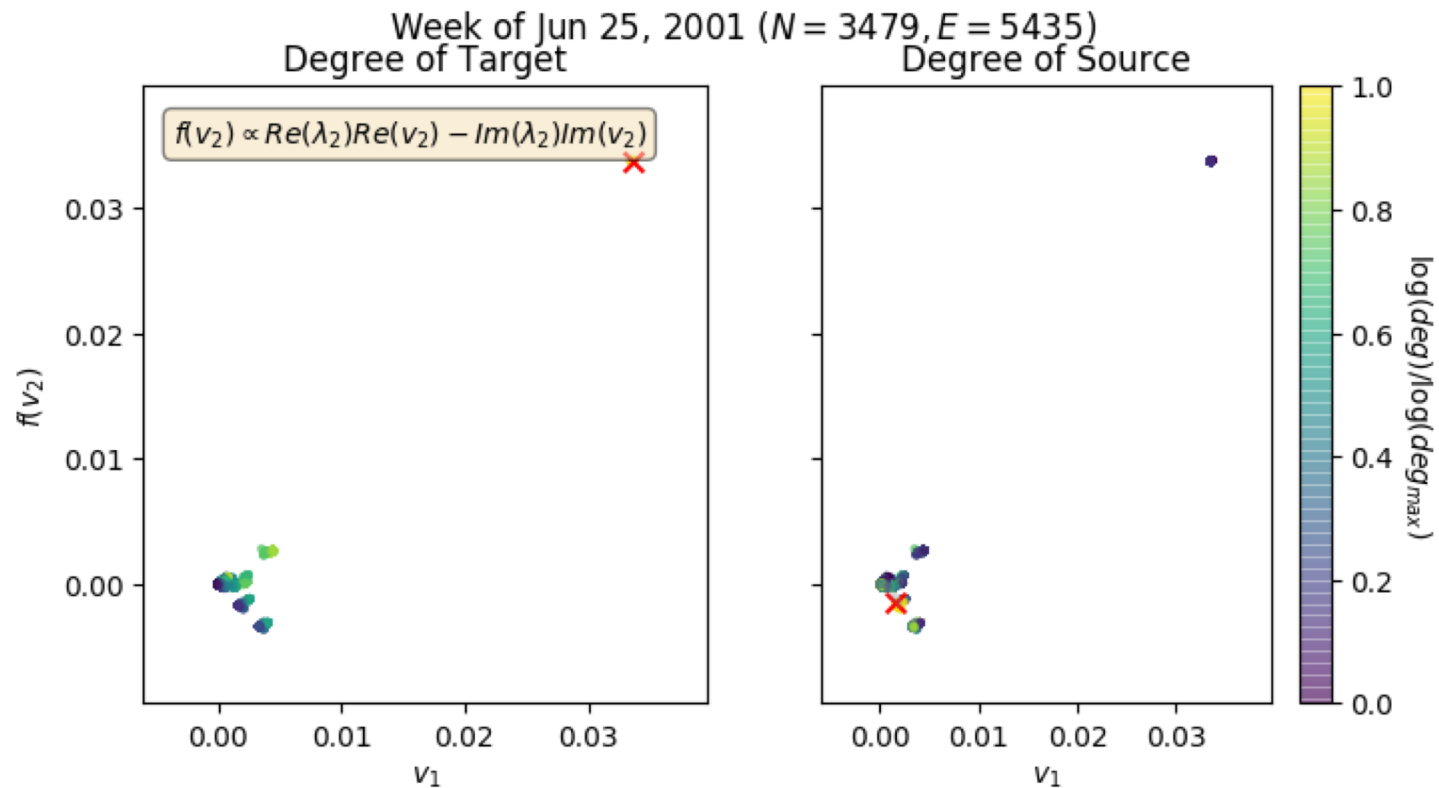
Enron email network: Jeff Skilling becomes Enron CEO



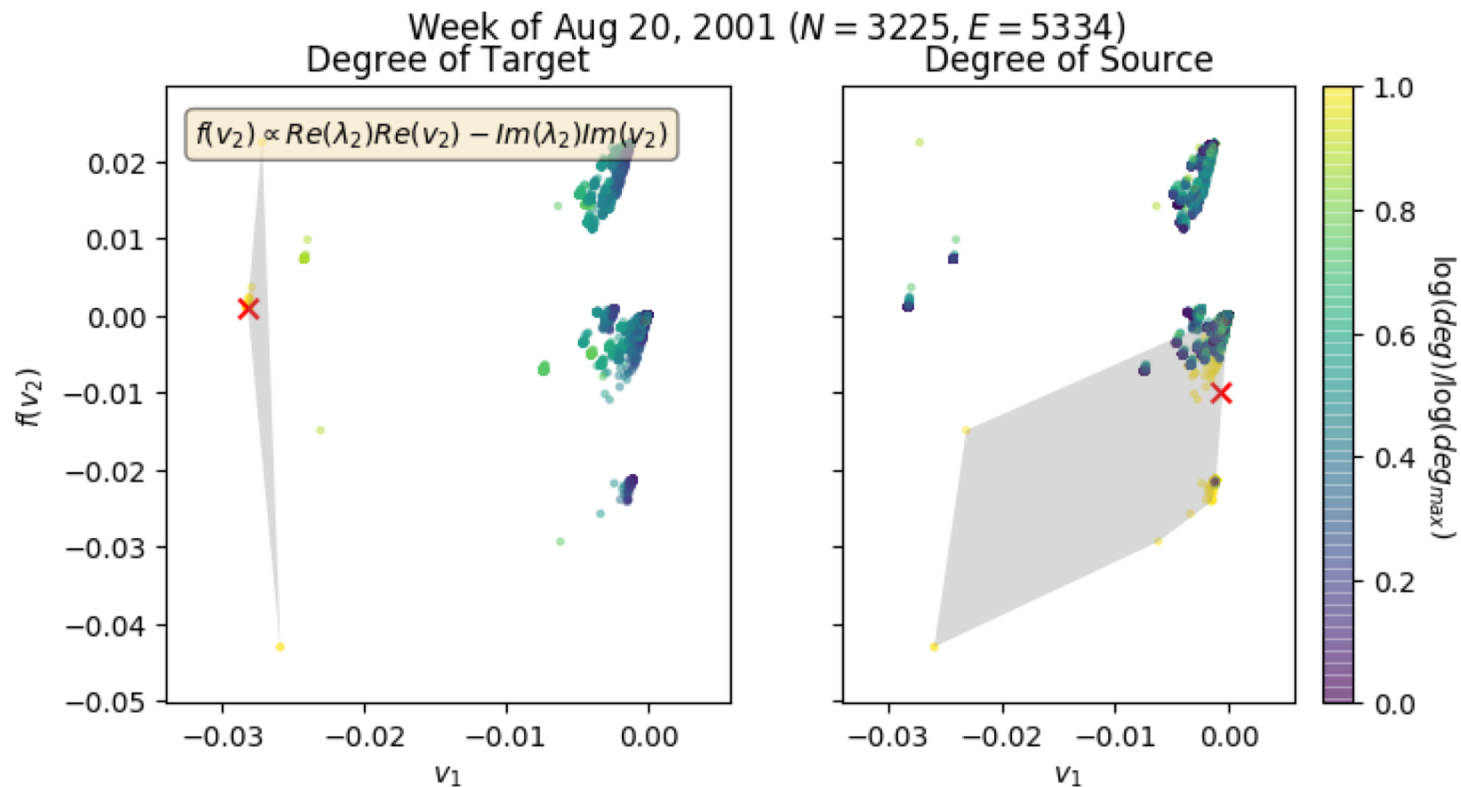
Enron email network: Analyst call to boost stock



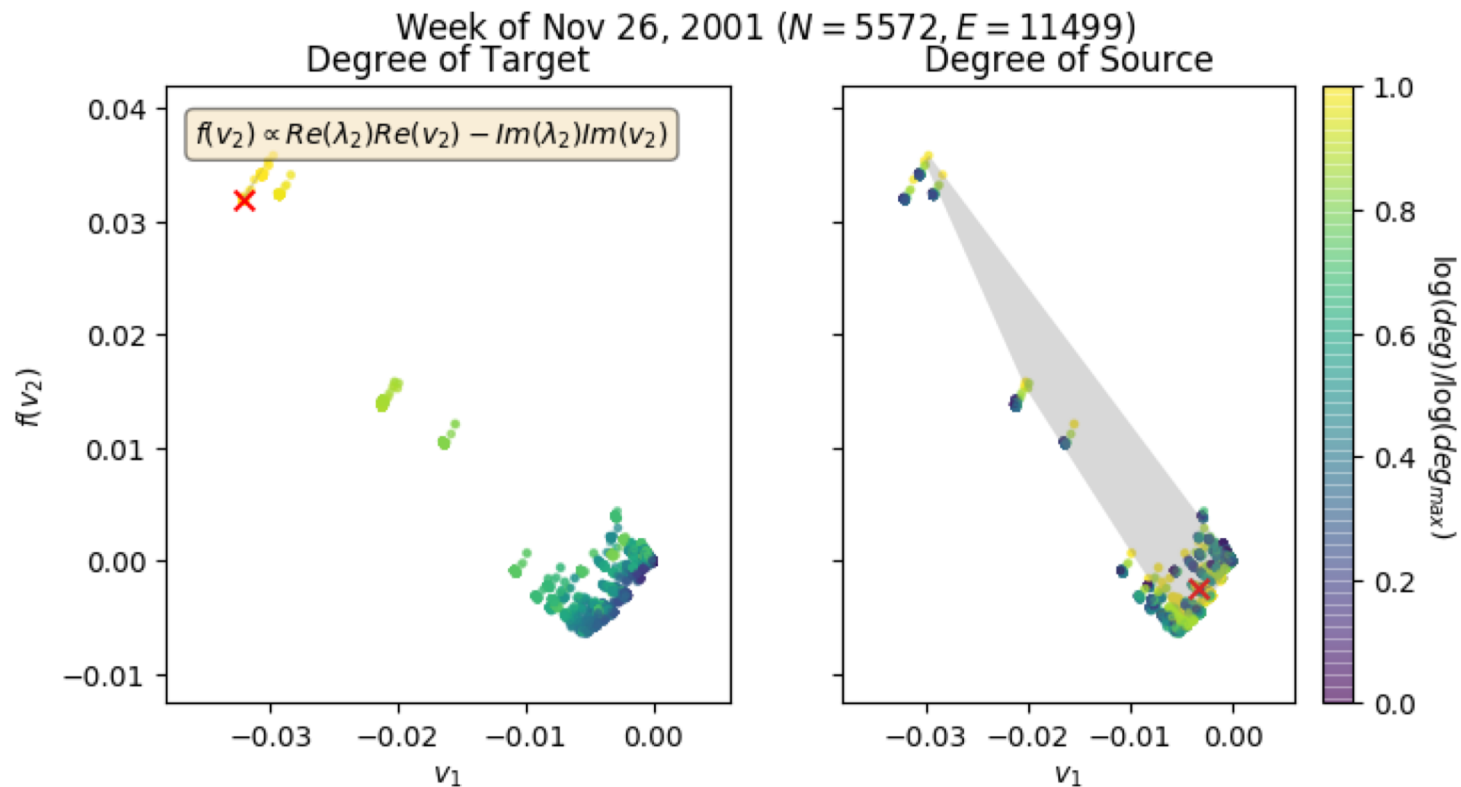
Enron email network: California energy crisis ends



Enron email network: One week after Skilling resigns



Enron email network: Enron stocks plunge below \$1



Enron email network: Enron goes bankrupt.

