Graph based similarity: going beyond personal Page Rank

Yosi Keller

School of Engineering, Bar-Ilan University

Outline

1. Graph based similarity
2. Graph transduction:
   Probability propagation Vs. Diffusion propagation
3. Structural graph similarity
4. Application to shape matching

Motivation

John Donne
“no man is an island”

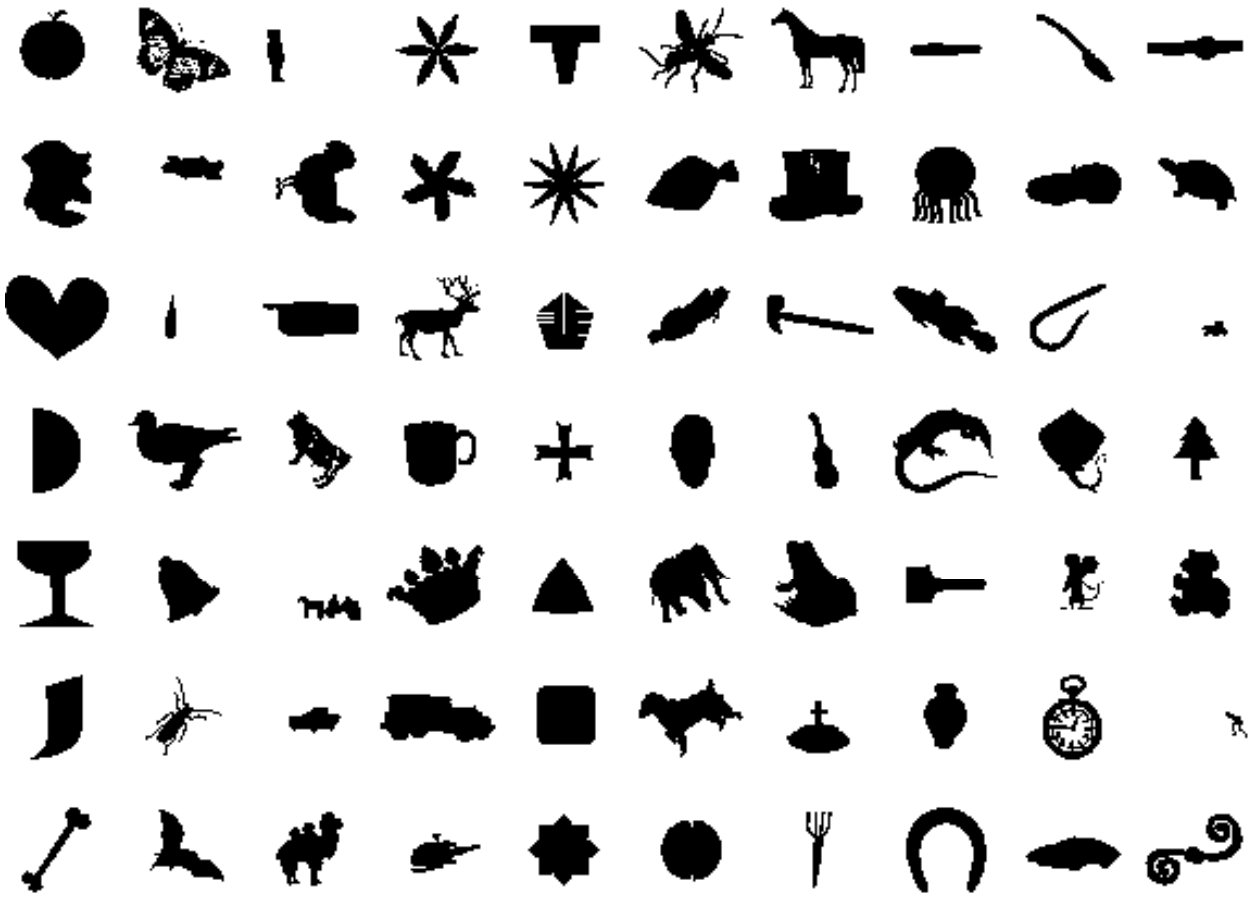
Motivation

John Donne
“no man is an island”

“no data point is an island”

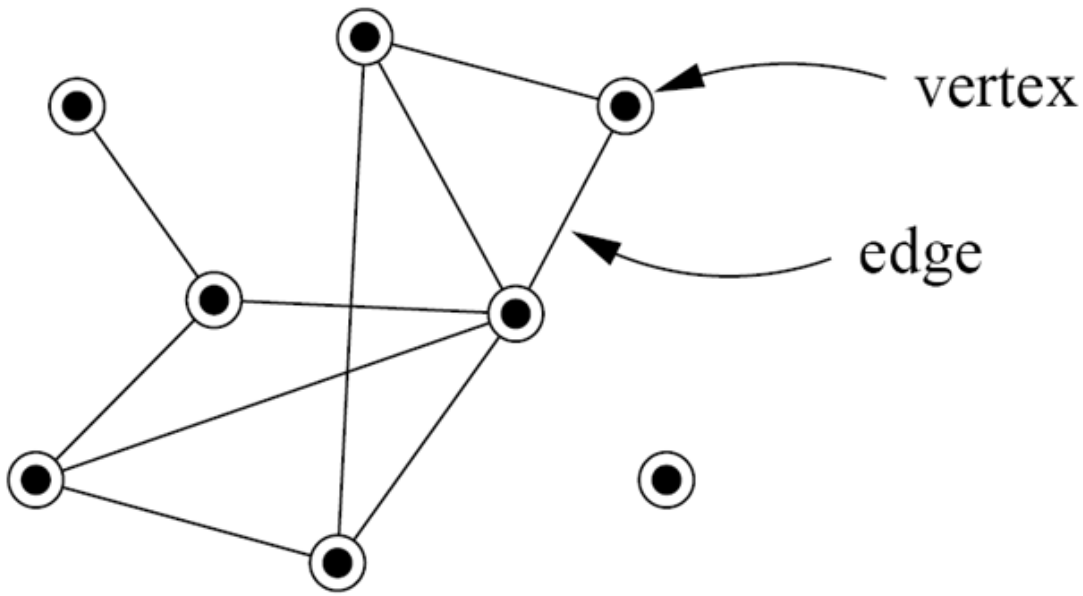
“No data point is an island”

In most applications data points are part of a set/manifold



Data manifolds can be represented by graphs

Data sets as graphs



Nodes $\{x_i\}_1^n$

Edges $e_{ij} = d(x_i, x_j)$

$d(\cdot, \cdot)$ is an application specific distance/affinity measure

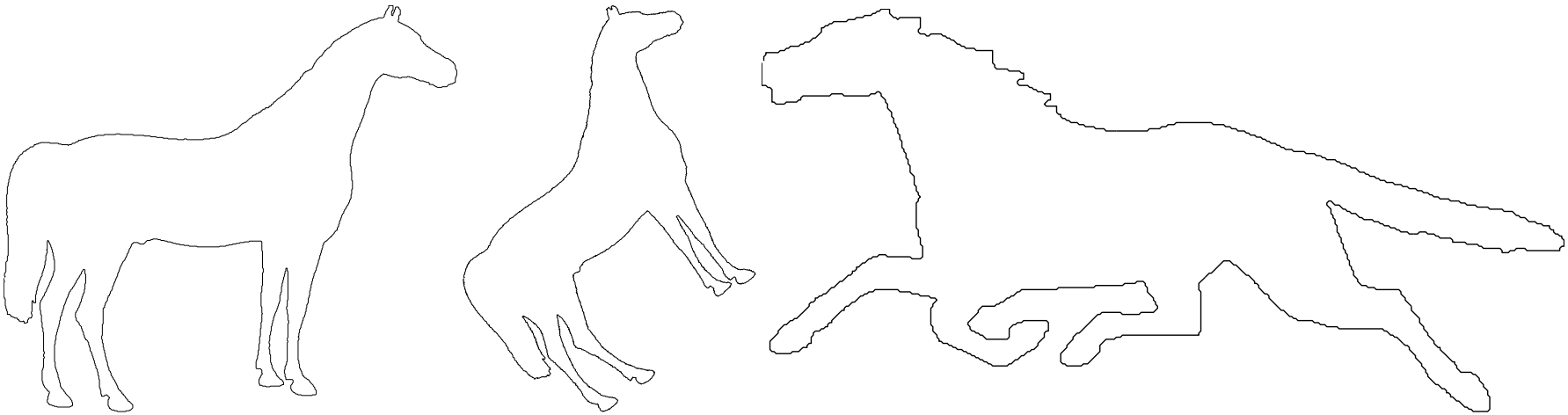
Recognition I

Given a query sample y and a set $\{x_i\}_1^n$, find

$$i^* = \arg \min_i d(y, x_i)$$

The problem:

- we can only approximate $d(\cdot, \cdot)$
- $d(\cdot, \cdot)$ might be corrupted by noise



Recognition II

- Useful for a gamut of applications:
  - Shape and object recognition
  - Target acquisition
  - HMM (state space) modeling
  - Soft Vs. Hard recognition
  - Can be part of a multi-layer system

Low level one-to-one shape matching

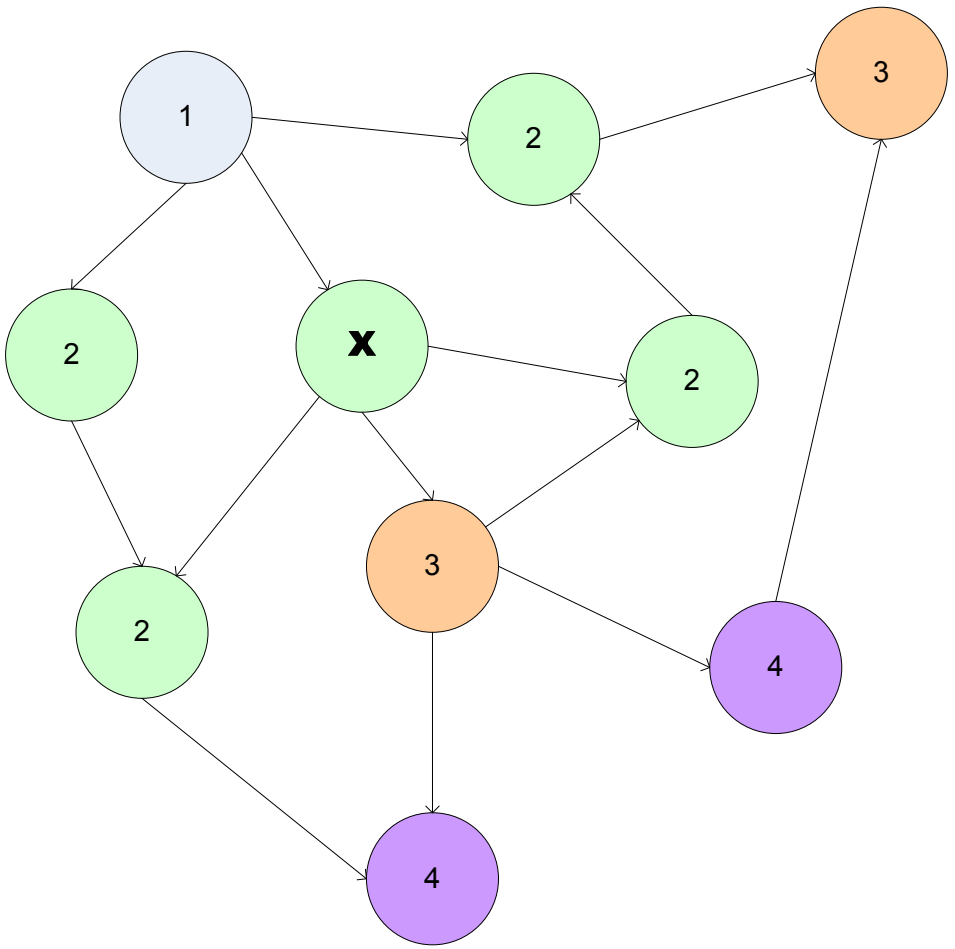
Algorithm	Performance
CSS Mokhtarian96	75.44%
Visual Parts Latecki00	76.45%
Shape Contexts Belongie03	76.51%
Curve Edit Distance Sebastian03	78.17%
MDS+SC+DP Ling07	84.35%
Planar Graph cuts Schmidt-cvpr09	85%
IDSC+DP Ling07	85.40%
IDSC+DP+EMD-L_1	86.56%
Shape-tree	87.7%
GM+IDSC	87.47%
GM+SC	88.11%
Contour Flexibility	89.31%

- Application specific
- Mediocre accuracy

Graph based recognition

Basic idea

My friend's friend is my friend



Mathematical formulation

The graph is analyzed by simulating a Markov random walk and analyzing its properties:

1. Represent the graph by an affinity matrix A
2. Row-normalize A to compute the Markov matrix $M = D^{-1}A$, where $d_{ii} = \sum_j a_{ij}$

$\mathbf{x}^T M = \lambda \mathbf{x}^T$ $\mathbf{x}$ is the asymptotic Markov state *probability*

$M\mathbf{x} = \lambda \mathbf{x}$ $\mathbf{x}$ a *diffusion* vector

Probability propagation

Personalized PageRank (Brin,Page,'98)

$$\mathbf{x}_{n+1}^T = (1-c)\,\mathbf{x}_n^T M + c\mathbf{r},\ c \in [0,1]$$

Fast Random Walk with Restarts (C. Faloutsos's group , CMU)

repeat the above N times

Theoretical analysis Fan Chung (UCSD) and Dan Spielman (Yale)

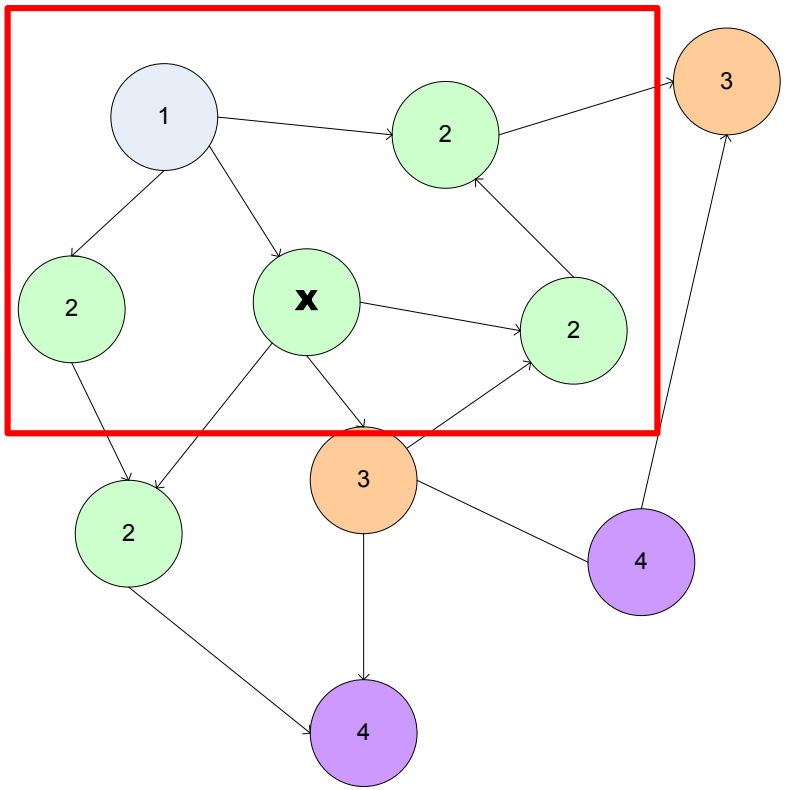
Diffusion propagation I

Xiang *et al.* , *Learning Context Sensitive Shape Similarity by Graph Transduction*, PAMI2009.

Algorithm	Performance
IDSC+DP+EMD-L_1	86.56%
Graph Transduction	92.3%

The algorithm

$$\mathbf{x}_{n+1} = M\mathbf{x}_n$$
$$\mathbf{x}_{n+1}(i_0) = 1$$



Xiang *et al*. actually compute the leading *constrained* eigenvector of the sub-graph.

Commute time/diffusion distance

Commute time/diffusion distance

$C(u,v)$ is the expected time for a random walk to travel between u and v and back

Commute time: Qiu and Hancock PAMI2007
Diffusion distance: Lafon and Coifman ACHA2006

$$C(u,v) = \|\Psi_t(u) - \Psi_t(v)\|_{L_2}^2 = \sum_{l=0}^{n-1} \lambda_l^{2t}(\psi_l(u) - \psi_l(v))^2$$

Our approach: structural similarity

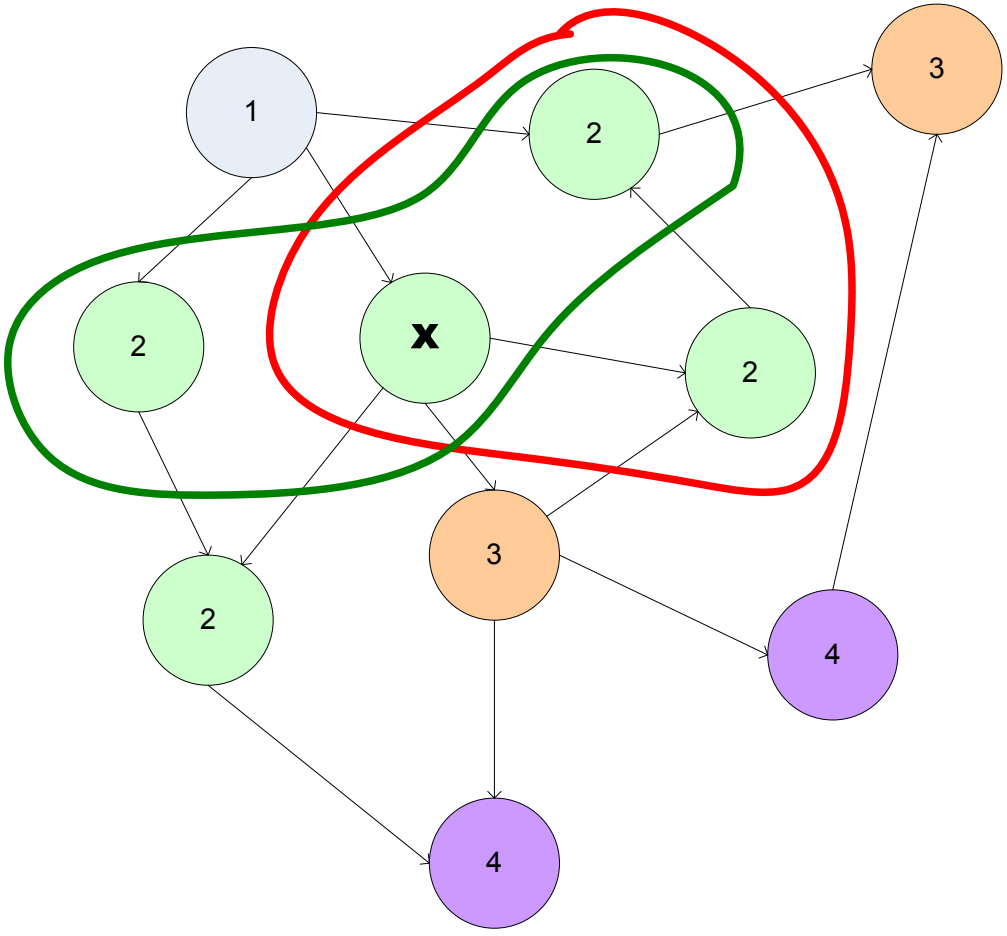
Instead of node similarity

Our approach: structural similarity

Instead of node similarity
use structural similarity

Our approach: structural similarity

Instead of node similarity
use structural similarity



Formally

Let $G_1 \subseteq G$ and $G_2 \subseteq G$, where G_1 and G_2 are $K-NN$ graphs. We aim to compute $d(G_1, G_2)$.

First idea: K-NN graph intersection

$$d(G_1, G_2) = |G_1 \cap G_2|$$

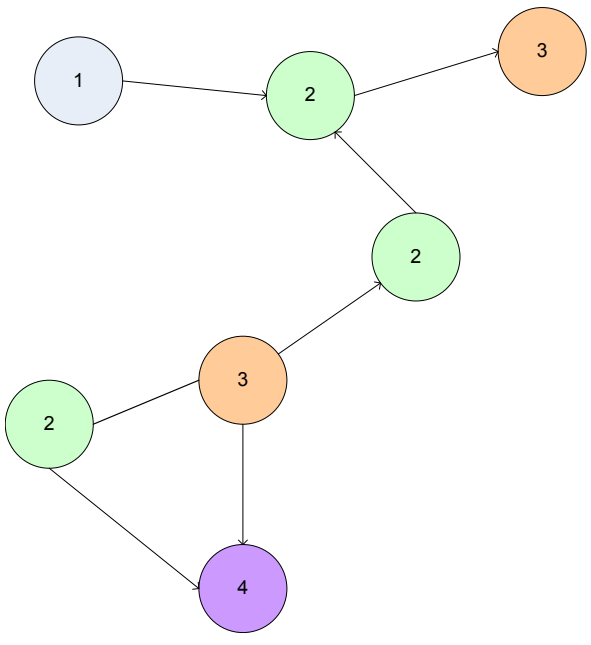
A. Egozi and Y. Keller2009

Algorithm	Performance
IDSC+DP+EMD-L_1	86.56%
Graph Transduction	92.3%
K-NN intersection	93.1%

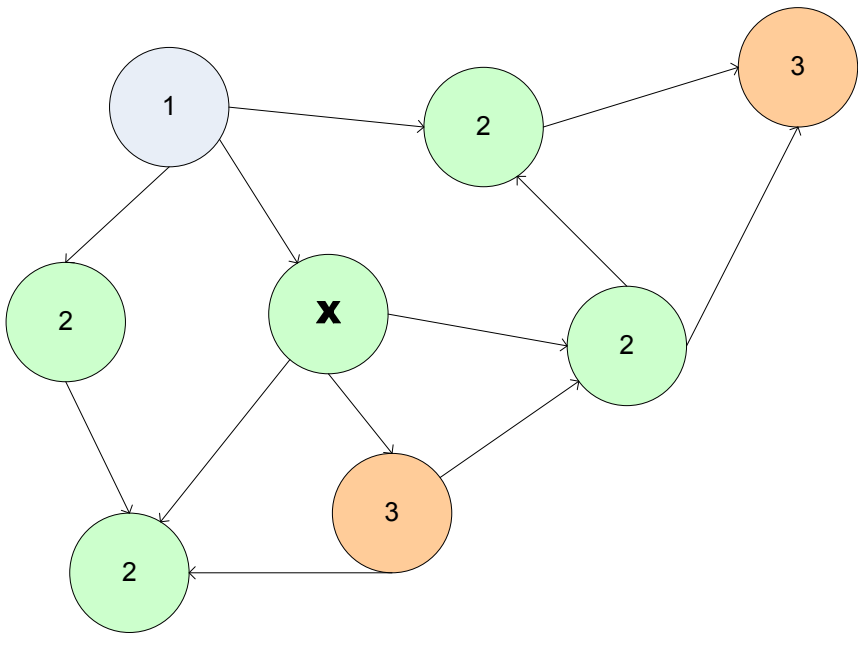
The mixing time of Markov chains I

Mixing time of a Markov chain

The time a Markov chain takes to converge to its stationary distribution



τ_1



τ_2

The mixing time of Markov chains II

$$\tau_1 \gg \tau_2$$

Second idea: use the mixing time of $G_i \cup G_j$

- Compute $\tau_{ij} = \tau\left(G_i \cup G_j\right)$
- $\tau_{ij} \approx \tau_{ii}$: x_i and x_j are similar
- $\tau_{ij} \gg \tau_{ii}$: x_i and x_j are dissimilar

Computing the mixing time

Given a graph G and a corresponding Markov matrix M, the mixing time $\tau(G)$ can be approximated by

$$\tau(G) = \frac{1}{1-\lambda_1}$$

where $\lambda_0 = 1 \geq \lambda_1 \geq \lambda_2$ are the eigenvalues of M.

Algorithm	Performance
IDSC+DP+EMD-L_1	86.56%
Graph Transduction	92.3%
K-NN intersection	93.1%
K-NN mixing time	95.2%

Conclusions

- Many data sources are derived from data manifolds
- Manifolds can be represented by graphs
- The graph structure can be used to improve general-purpose recognition
- Graphs can be analyzed by Markov walk theory
- Probability and diffusion can be propagated
- Intrinsic Markov walk properties such as the commute and mixing times can be used

Thanks You!