# Graph similarity

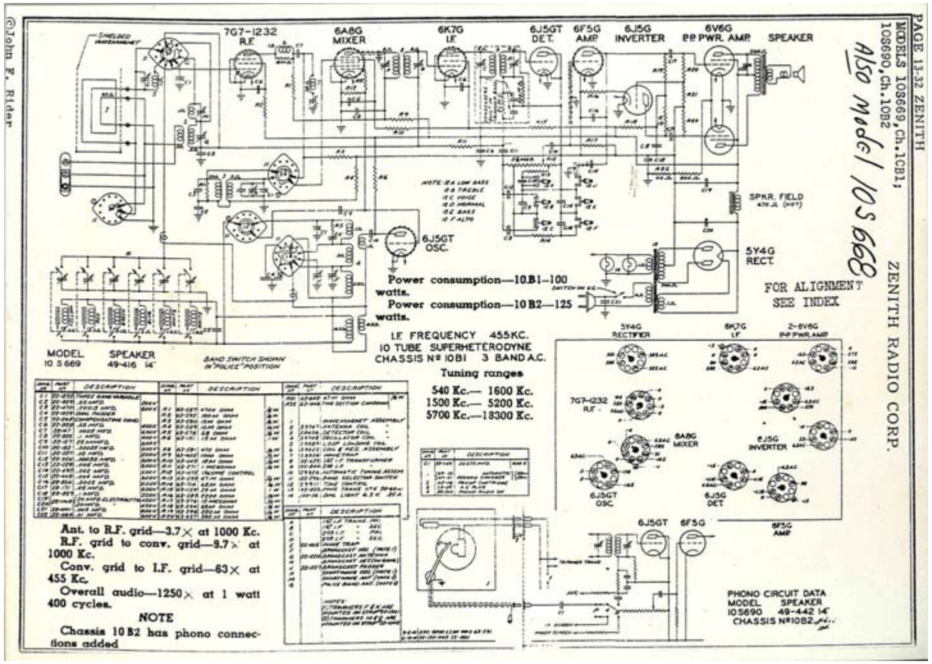
Laura Zager and George Verghese

EECS, MIT

March 2005

# Words you won’t hear today

- *impedance matching*
- *thyristor*
- *oxide layer*
- *VARs*



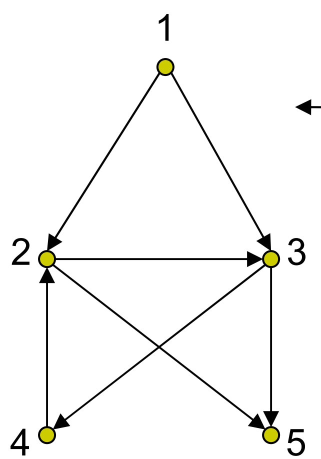
# Some quick definitions

$G(V, E)$ ← a graph **G**

- $V$ ← the set of *vertices* or *nodes*
- $E \subset V \times V$ ← the set of *edges* – can be *directed* or *undirected.*

---

<u>ex</u>

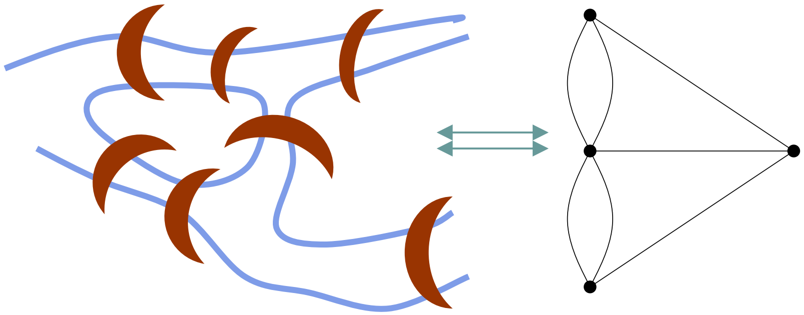



a directed graph and its
*node-node adjacency matrix*

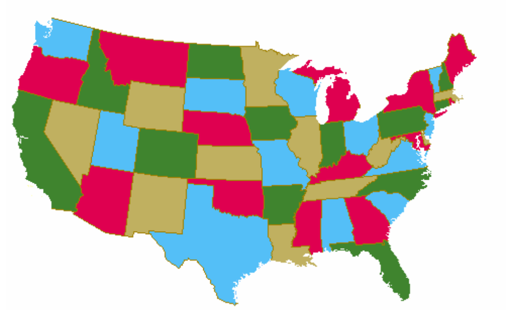
$$\begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

# Graph theory:  some perspective

The Königsberg bridge problem (18th c.)

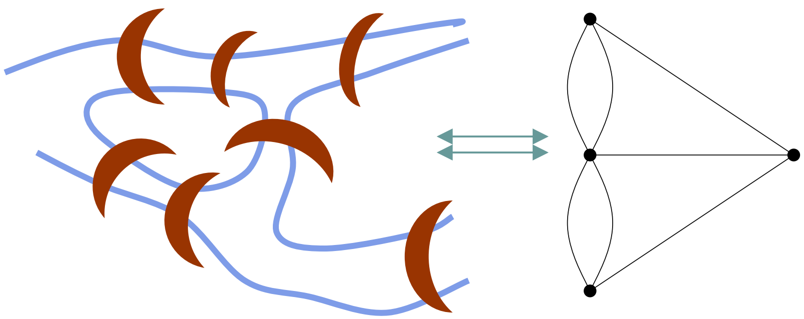


The Four Color Theorem (1976)

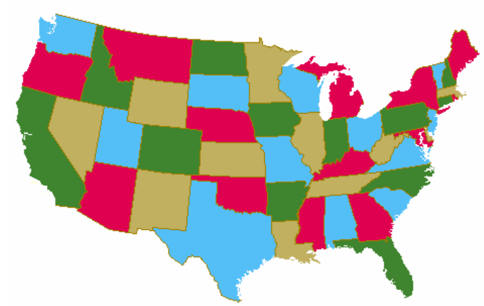


# Graph theory: some perspective

The Königsberg bridge problem
(18th c.)



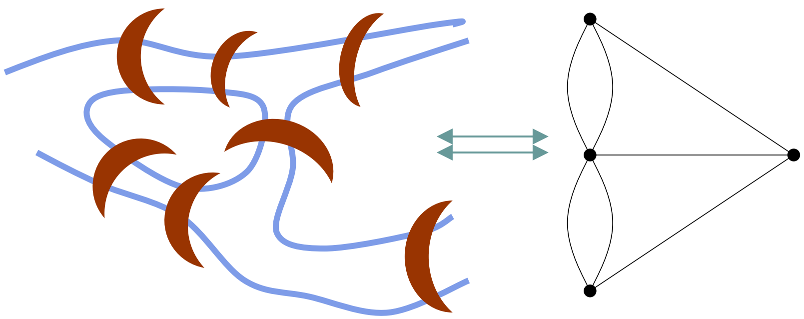
The Four Color Theorem
(1976)



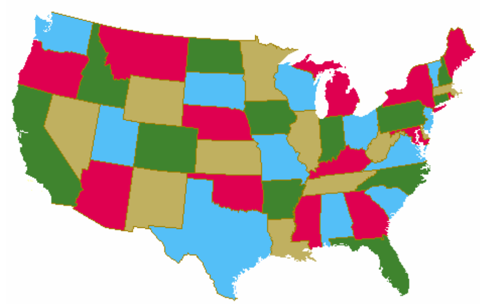
Erdös and Rényi random graph models
(1959)

# Graph theory:  some perspective

The Königsberg bridge problem ($18^{th}$ c.)



The Four Color Theorem (1976)



Erdös and Rényi random graph models (1959)

**present and future:**

graphs that arise in the natural world

# Applications

- Comparing biological networks
  - Deriving phylogenetic trees from metabolic pathway data [Heymans, Singh, 2003].
- Social network mapping
  - Small world phenomena [Milgram, 1967; Watts, 1999].
- Web searching
  - Improving searching results using WWW structure [Kleinberg, 1999].
- Chemical structure matching
  - Finding similar structures in a chemical database [Hattori et al., 2003].

# Applications

- Comparing biological networks
  - Deriving phylogenetic trees from metabolic pathway data [Heymans, Singh, 2003].
- Social network mapping
  - Small world phenomena [Milgram, 1967; Watts, 1999].
- Web searching
  - Improving searching results using WWW structure [Kleinberg, 1999].
- Chemical structure matching
  - Finding similar structures in a chemical database [Hattori et al., 2003].

one common thread: similarity

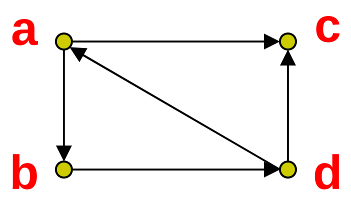
# Notions of similarity

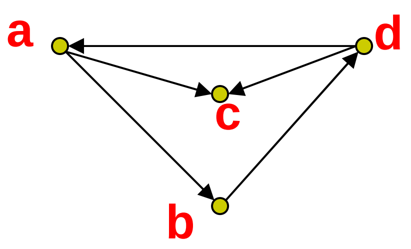
- Isomorphism – identifying a *bijection* between the nodes of two graphs which preserves (directed) adjacency.

  - Corneil & Gotlieb, *Journal of the ACM*, 1970.
  - Pelillo, *Neural Computation*, 1999.
  - Ullman, *Journal of the Assoc. of Computing Machinery*, 1976.

# Notions of similarity

- Isomorphism – identifying a *bijection* between the nodes of two graphs which preserves (directed) adjacency.

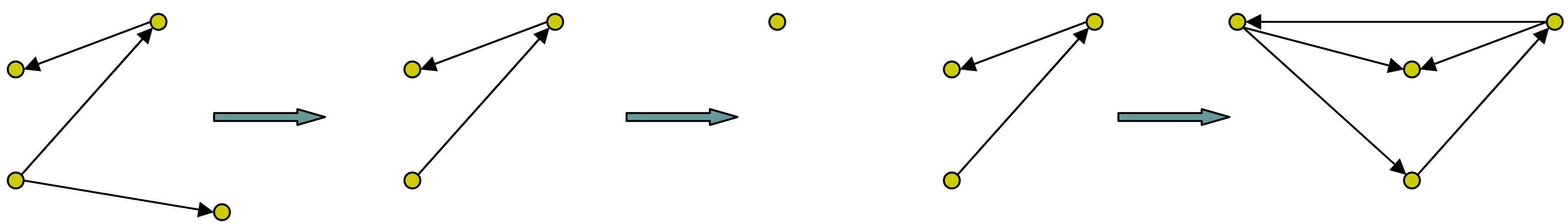



  - Corneil & Gotlieb, *Journal of the ACM*, 1970.
  - Pelillo, *Neural Computation*, 1999.
  - Ullman, *Journal of the Assoc. of Computing Machinery*, 1976.

# Notions of similarity

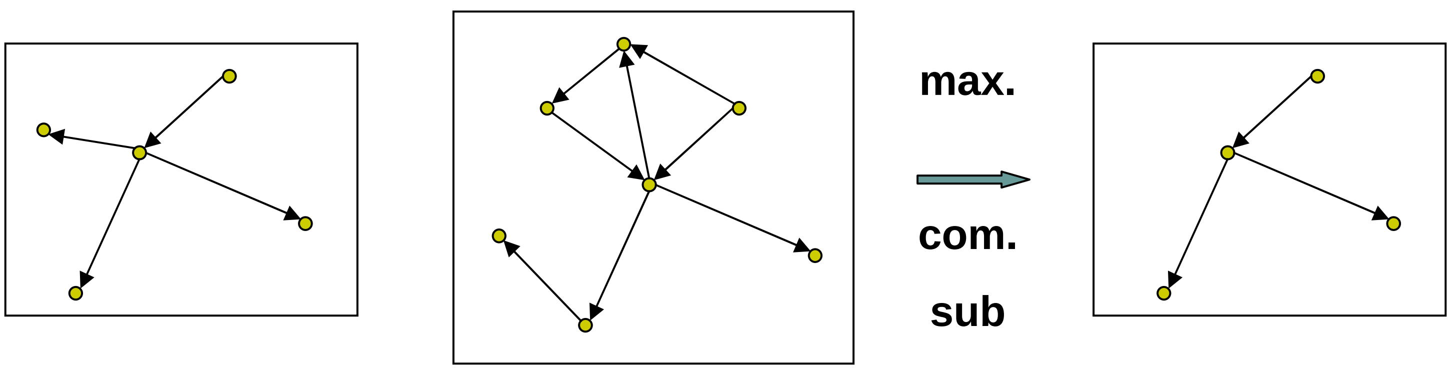

- Edit distance – given a cost function on *edit operations* (e.g. addition/deletion of nodes and edges), determine the minimum cost transformation from one graph to another.



  - Bunke, *IEEE Trans. Pattern Analysis and Machine Int.,* 1999.
  - Messmer & Bunke, *IEEE Trans. Pattern Analysis and Machine Int.,* 1998.

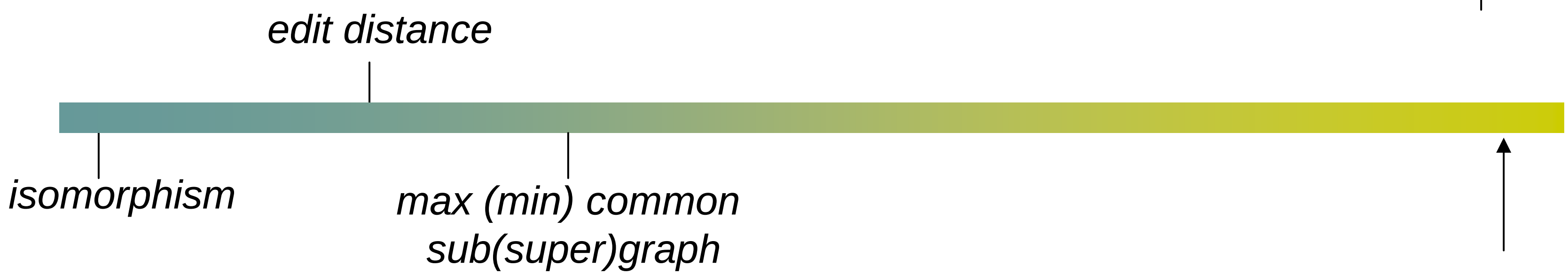
# Notions of similarity


- Maximum common subgraph – identifying the `largest' isomorphic subgraphs of two graphs.
- Minimum common supergraph – identifying the `smallest' graph that contains both graphs.

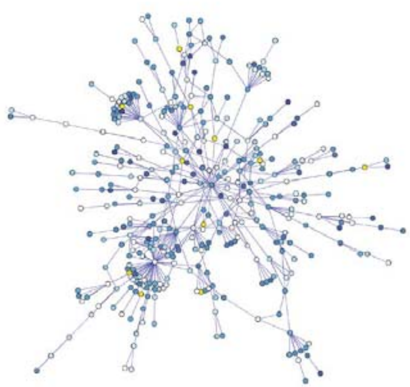


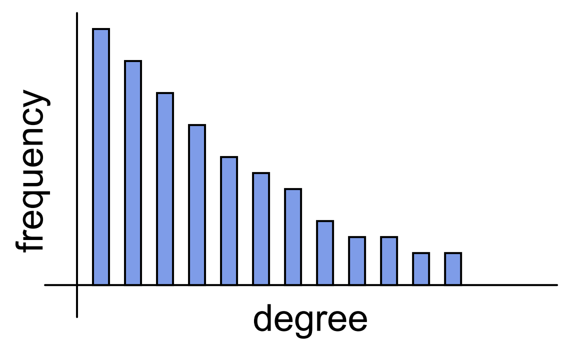

- Fernandez & Valiente, *Pattern Recognition Letters*, 2001.
- Bunke, Jiang & Candel, *Computing*, 2000.

# Notions of similarity




- Statistical methods – assessing *aggregate measures* of graph structure (e.g. degree distribution, diameter, betweenness measures).






  - Albert, Barabasi, *Reviews of Modern Physics*, 2002
  - Dill, Kumar, et al., *ACM Transactions on Internet Technology*, 2002.
  - Watts, Small Worlds, 1999.

# Notions of similarity

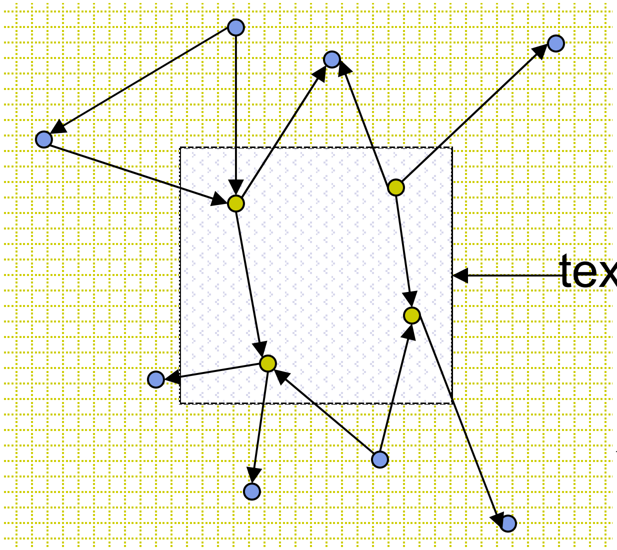

- Iterative methods:

  Two graph elements (e.g., edges or nodes) are similar if their neighborhoods are similar.

  - Kleinberg, *Journal of the ACM*, 1999.
  - Blondel, Van Dooren, et al., *SIAM Review*, 2004.
  - Jeh & Widom, *8th Intl. Conf. on Knowledge Discovery and Data Mining*, 2002.
  - Melnik, Garcia-Molina, *18th Intl. Conf. on Data Engineering*, 2002.
  - Heymans & Singh, *Bioinformatics*, 2003.

# Kleinberg, 1999*

- Motivated by demands of web searching
  - Step 1: Use text-based search methods to identify a candidate graph containing relevant websites and their neighbors.



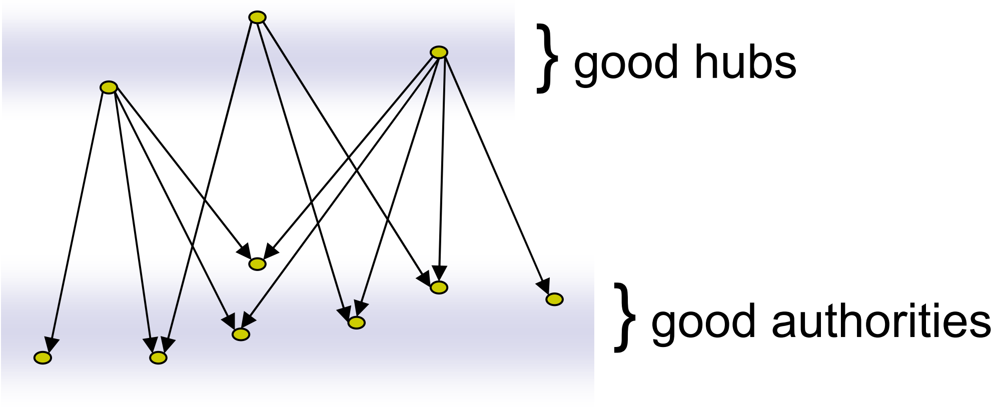
text-based search results

*plus* the local neighborhood

**Kleinberg, J.M. Authoritative sources in a hyperlinked environment. *Journal of the ACM*. 1999**

# Kleinberg, 1999

- Relevant search results might be:
  - Hubs – pages which *point to* many good authorities
  - Authorities – pages which *are pointed to* by many good hubs

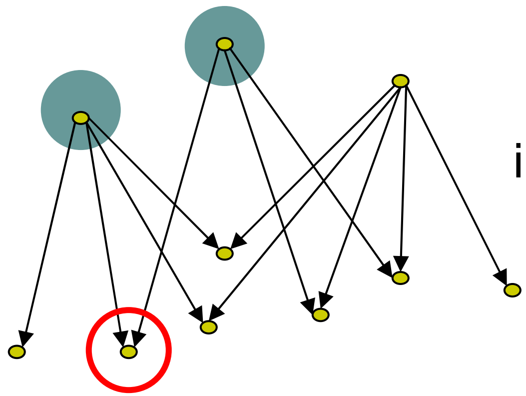



- Step 2: Compute hub and authority scores for every node in the candidate graph.

# Kleinberg, 1999

- Denote:
  - $x_{1p}(k)$ = hub score of node *p* at iteration *k*
  - $x_{2p}(k)$ = authority score of node *p* at iteration *k*

- Update rule:



$$x_{2p}(k+1) = \sum_{q:(q,p)\in E} x_{1q}(k)$$

i.e. the sum of hub scores of nodes that <u>point to</u> node *p*

$$x_{1p}(k+1) = \sum_{q:(p,q)\in E} x_{2q}(k)$$

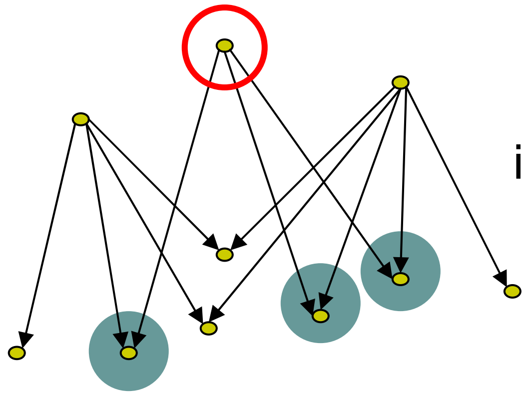
i.e. the sum of authority scores of nodes that <u>are pointed to</u> by node p

- Normalize the scores so that $\sum_p x_{ip}$ = 1 and repeat.

# Kleinberg, 1999

- Denote:
  - $x_{1p}(k)$ = hub score of node *p* at iteration *k*
  - $x_{2p}(k)$ = authority score of node *p* at iteration *k*

- Update rule:



$$x_{2p}(k+1) = \sum_{q:(q,p)\in E} x_{1q}(k)$$

i.e. the sum of hub scores of nodes that <u>point to</u> node *p*

$$\boxed{x_{1p}(k+1) = \sum_{q:(p,q)\in E} x_{2q}(k)}$$

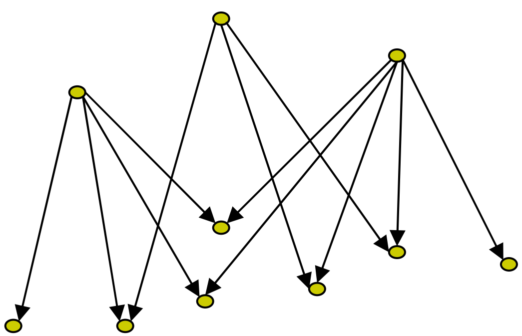
i.e. the sum of authority scores of nodes that <u>are pointed to</u> by node p

- Normalize the scores so that $\sum_p x_{ip}$ = 1 and repeat.

# Kleinberg, 1999

- Denote:
  - $x_{1p}(k)$ = hub score of node *p* at iteration *k*
  - $x_{2p}(k)$ = authority score of node *p* at iteration *k*

- Update rule:



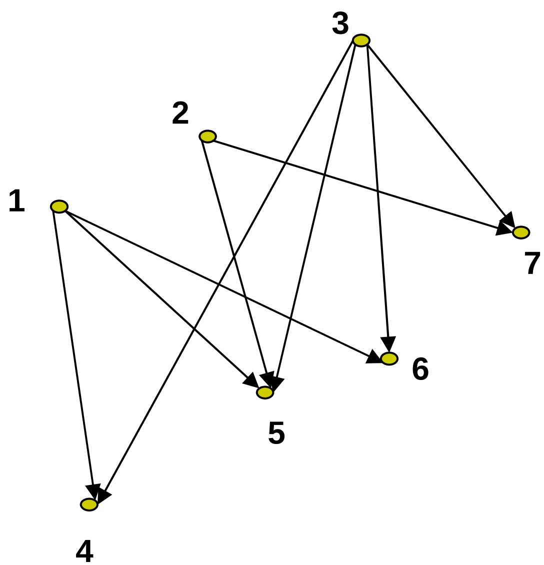
- Stack the scores $x_{1p}(k)$ into a vector $[x_1]_k$, then stack $[x_1]_k$ and $[x_2]_k$.

- Let B be the *node-node adjacency matrix* of the candidate graph. Then:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_{k+1} = \begin{bmatrix} 0 & B \\ B' & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_k$$

# Kleinberg, 1999

Ex.

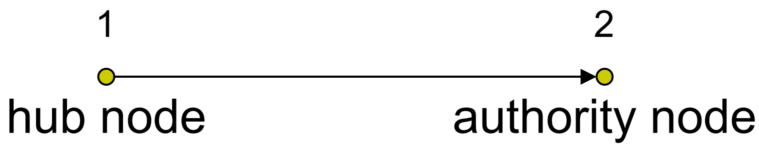



| nodes | $x_1$ hub scores | $x_2$ authority scores |
|---|---|---|
| **1** | 0.374 | 0 |
| **2** | 0.242 | 0 |
| **3** | 0.467 | 0 |
| **4** | 0 | 0.365 |
| **5** | 0 | 0.467 |
| **6** | 0 | 0.365 |
| **7** | 0 | 0.308 |

- for a good read, see “The Ongoing Search for Efficient Web Search Algorithms,” SIAM News, November 2004.

# Blondel, Van Dooren, et al., 2004*

- Views Kleinberg’s iteration as a comparison between the web graph and a *hub-authority graph:*




$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

- Observe that the matrix form of Kleinberg’s update can be written as follows:

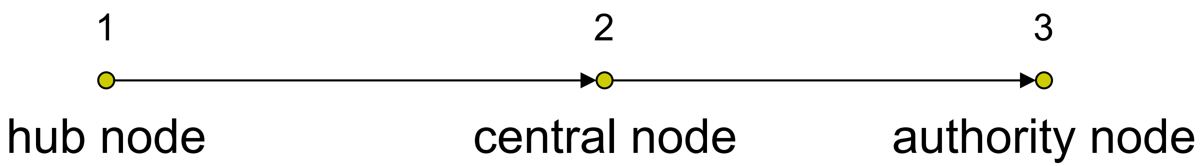
$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_{k+1} = \begin{bmatrix} 0 & B \\ B' & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_{k} = \left( A \otimes B + A' \otimes B' \right) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_{k}$$

- Is this generalizable to any two graphs $G_A$ and $G_B$?

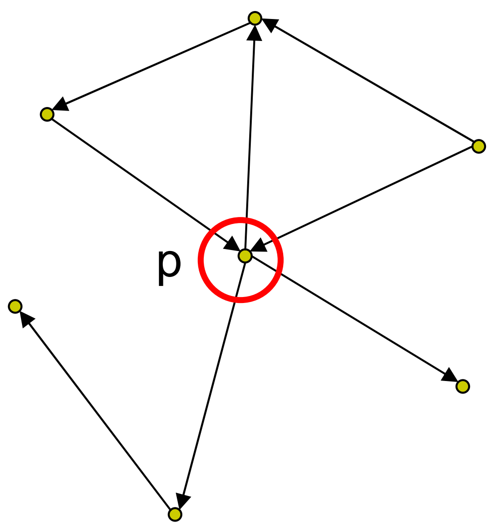
*Blondel, V., Gajardo, A., Heymans, M., Senellart, P., Van Dooren, P. A measure of similarity between graph vertices: applications to synonym extraction and web searching. *SIAM Review*, v. 46(4), 647-666. 2004.

# Blondel, Van Dooren, et al., 2004

- A first step toward generalizing Kleinberg's approach: consider comparing the graph $G_B$ to the following graph using a similar update:




$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$



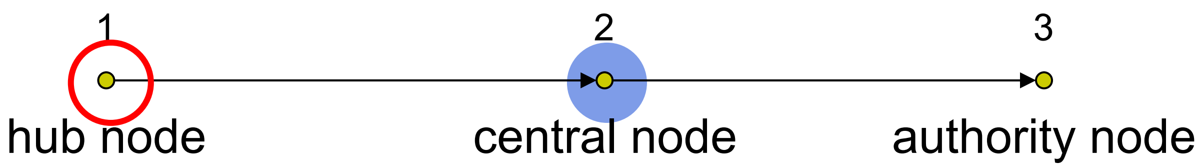

$$x_{1p}(k+1) = \sum_{q:(p,q)\in E} x_{2q}(k)$$

$$x_{2p}(k+1) = \sum_{q:(q,p)\in E} x_{1q}(k) + \sum_{q:(p,q)\in E} x_{3q}(k)$$

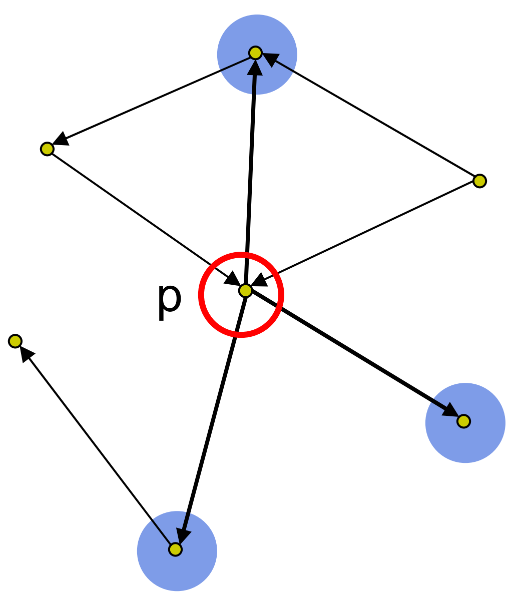
$$x_{3p}(k+1) = \sum_{q:(q,p)\in E} x_{2q}(k)$$

# Blondel, Van Dooren, et al., 2004

- A first step toward generalizing Kleinberg's approach: consider comparing the graph $G_B$ to the following graph using a similar update:




$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$



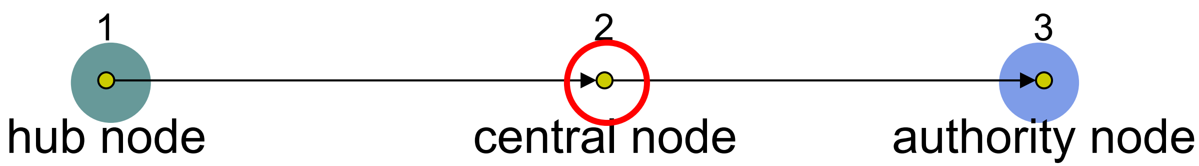

$$\boxed{x_{1p}(k+1) = \sum_{q:(p,q)\in E} x_{2q}(k)}$$

$$x_{2p}(k+1) = \sum_{q:(q,p)\in E} x_{1q}(k) + \sum_{q:(p,q)\in E} x_{3q}(k)$$

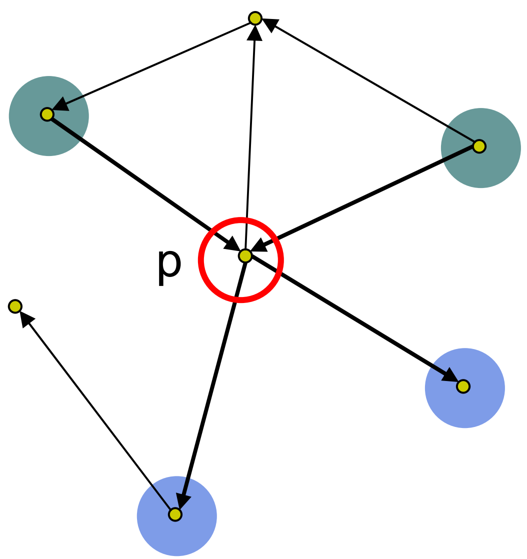
$$x_{3p}(k+1) = \sum_{q:(q,p)\in E} x_{2q}(k)$$

# Blondel, Van Dooren, et al., 2004

- A first step toward generalizing Kleinberg's approach: consider comparing the graph $G_B$ to the following graph using a similar update:




$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$



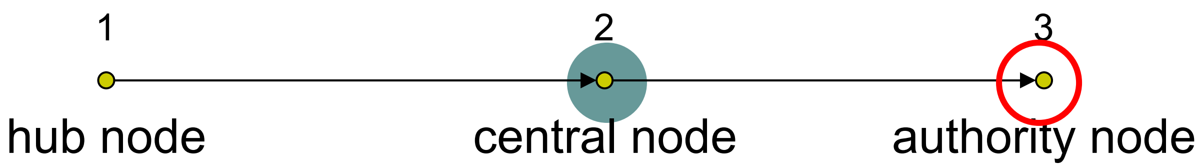

$$x_{1p}(k+1) = \sum_{q:(p,q)\in E} x_{2q}(k)$$

$$\boxed{x_{2p}(k+1) = \sum_{q:(q,p)\in E} x_{1q}(k) + \sum_{q:(p,q)\in E} x_{3q}(k)}$$

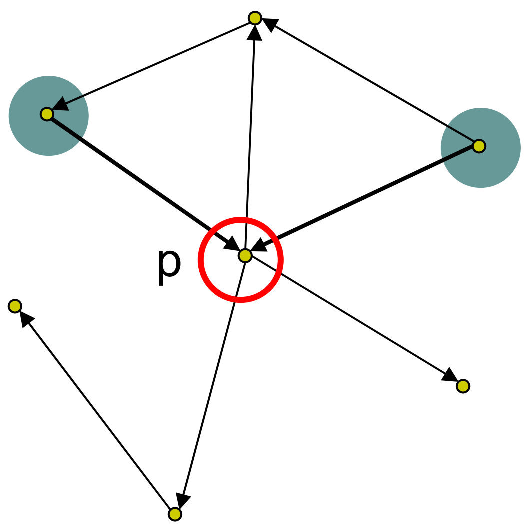
$$x_{3p}(k+1) = \sum_{q:(q,p)\in E} x_{2q}(k)$$

# Blondel, Van Dooren, et al., 2004

- A first step toward generalizing Kleinberg's approach: consider comparing the graph $G_B$ to the following graph using a similar update:




$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$



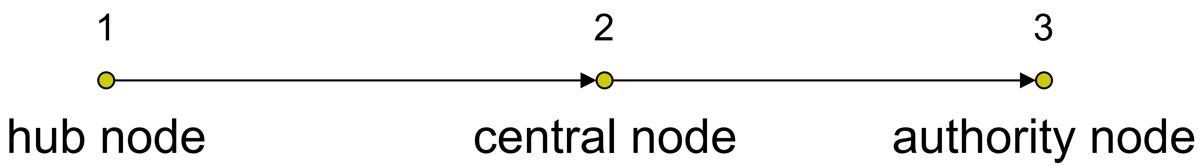

$$x_{1p}(k+1) = \sum_{q:(p,q)\in E} x_{2q}(k)$$

$$x_{2p}(k+1) = \sum_{q:(q,p)\in E} x_{1q}(k) + \sum_{q:(p,q)\in E} x_{3q}(k)$$

$$\boxed{x_{3p}(k+1) = \sum_{q:(q,p)\in E} x_{2q}(k)}$$

# Blondel, Van Dooren, et al., 2004

- A first step toward generalizing Kleinberg's approach: consider comparing the graph $G_B$ to the following graph using a similar update:




$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_{k+1} = \begin{bmatrix} 0 & B & 0 \\ B' & 0 & B \\ 0 & B' & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_k = \underline{\underline{\left( A \otimes B + A' \otimes B' \right)}} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_k$$

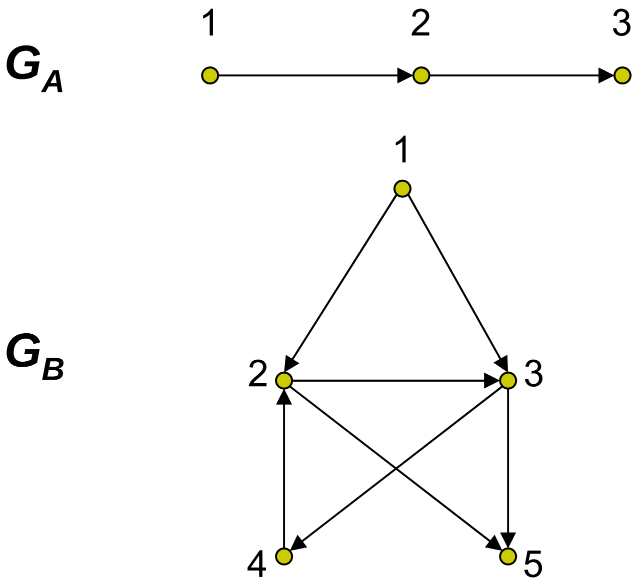
*(use this construction for automatic synonym extraction)*

# Blondel, Van Dooren, et al., 2004

- In general, the nodes of two graphs $G_A$ and $G_B$ can be compared via the following update:

$$\overline{x}_{k+1} = \left(A \otimes B + A' \otimes B'\right)\overline{x}_k$$

Ex.




**similarity scores**

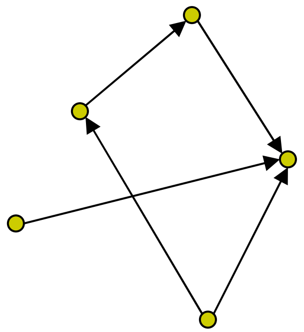
| nodes | **1** | **2** | **3** |
|---|---|---|---|
| **1** | 0.443 | 0.104 | 0 |
| **2** | 0.280 | 0.396 | 0.086 |
| **3** | 0.086 | 0.396 | 0.280 |
| **4** | 0.222 | 0.049 | 0.222 |
| **5** | 0 | 0.104 | 0.443 |

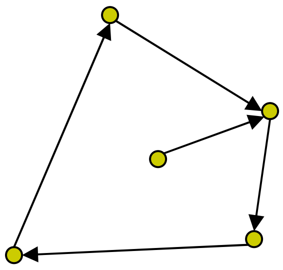
# Coupled edge and node scoring

- Idea: use this iterative approach to assign *edge similarity scores* as well as *node similarity scores*.
- Couple the definitions in the following manner:

$x_{ij}$ = similarity between node *i* in $G_B$ and node *j* in $G_A$
= sum of pairwise similarities between adjacent edges

$y_{ij}$ = similarity between edge *i* in $G_B$ and edge *j* in $G_A$.
= sum of similarities of source and terminal nodes





# Coupled edge and node scoring

- Idea: use this iterative approach to assign *edge similarity scores* as well as *node similarity scores*.
- Couple the definitions in the following manner:

$x_{ij}$ = similarity between node *i* in $G_B$ and node *j* in $G_A$
= sum of pairwise similarities between adjacent edges

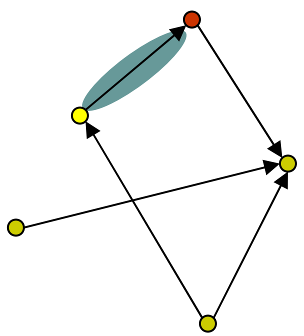
$y_{ij}$ = similarity between edge *i* in $G_B$ and edge *j* in $G_A$.
= sum of similarities of source and terminal nodes

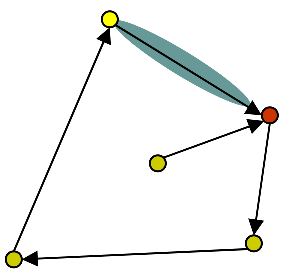
# Coupled edge and node scoring

- Idea: use this iterative approach to assign *edge similarity scores* as well as *node similarity scores*.
- Couple the definitions in the following manner:

$x_{ij}$ = similarity between node *i* in $G_B$ and node *j* in $G_A$
= sum of pairwise similarities between adjacent edges

$y_{ij}$ = similarity between edge *i* in $G_B$ and edge *j* in $G_A$.
= sum of similarities of source and terminal nodes





# Coupled edge and node scoring

- Idea: use this iterative approach to assign *edge similarity scores* as well as *node similarity scores*.
- Couple the definitions in the following manner:

$x_{ij}$ = similarity between node $i$ in $G_B$ and node $j$ in $G_A$
= sum of pairwise similarities between adjacent edges

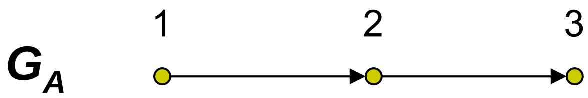
$y_{ij}$ = similarity between edge $i$ in $G_B$ and edge $j$ in $G_A$.
= sum of similarities of source and terminal nodes

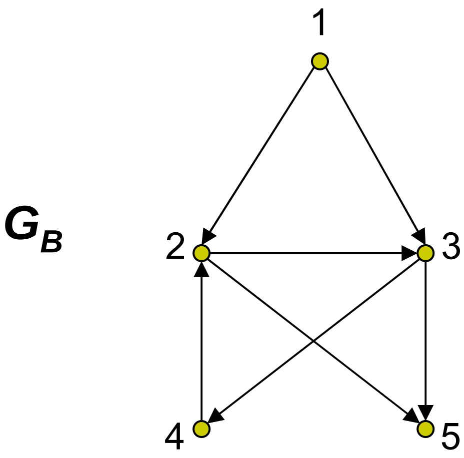
$$\bar{x}_{k+1} = \left[A_S \otimes B_S + A_T \otimes B_T\right]\bar{y}_k$$

$$\bar{y}_{k+1} = \left[A_S{}' \otimes B_S{}' + A_T{}' \otimes B_T{}'\right]\bar{x}_k$$

$$\left[A_S\right]_{ij} = \begin{cases} 1 & s(j) = i \\ 0 & else \end{cases} \qquad \left[A_T\right]_{ij} = \begin{cases} 1 & t(j) = i \\ 0 & else \end{cases}$$

# Example




**Blondel, Van Dooren, et al.**
**similarity scores**

| nodes | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0.443 | 0.104 | 0 |
| 2 | 0.280 | 0.396 | 0.086 |
| 3 | 0.086 | 0.396 | 0.280 |
| 4 | 0.222 | 0.049 | 0.222 |
| 5 | 0 | 0.104 | 0.443 |

**Coupled model**
**similarity scores**

| nodes | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0.324 | 0.054 | 0 |
| 2 | 0.177 | 0.587 | 0.018 |
| 3 | 0.018 | 0.587 | 0.177 |
| 4 | 0.127 | 0.010 | 0.127 |
| 5 | 0 | 0.054 | 0.324 |

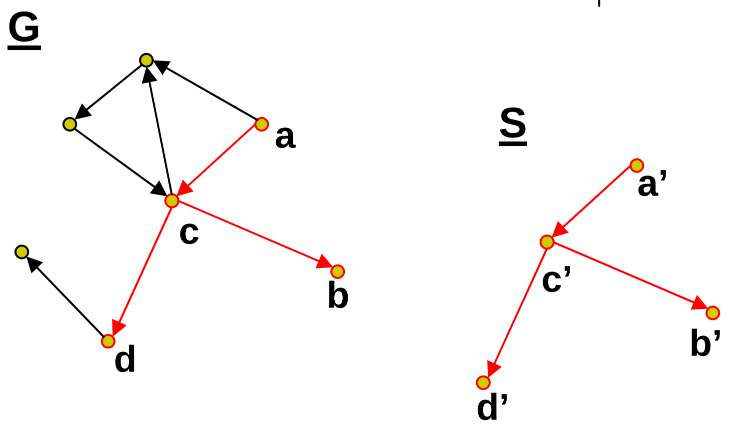
# Application: Graph Matching

- Assign a correspondence between nodes and/or edges of each graph to maximize some performance criteria.
  - The Approach: apply *Hungarian algorithm* to node similarity matrix to maximize the sum of matched scores.

| | | |
|---|---|---|
| 1 | 3 | 7 |
| 3 | 2 | 4 |
| 4 | 8 | 3 |

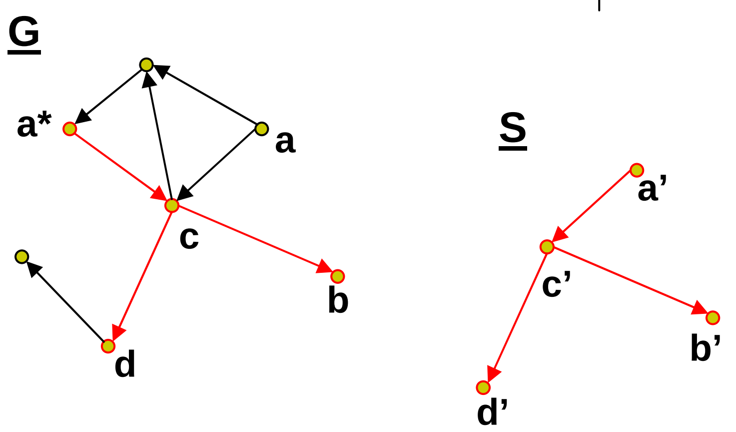
| | | |
|---|---|---|
| 1 | 3 | 7 |
| 3 | 2 | 4 |
| 4 | 8 | 3 |

# Application: Graph Matching




- Task: subgraph matching
  - Generate a random graph, *G.*
  - Select a subgraph, *S.*
  - Compute the node similarity matrices between *G* and *S.*
  - Apply the Hungarian algorithm to \`best’ match the nodes of *S* to those in *G* by finding a matching the <u>maximizes the sum of matched scores</u>.
  - Record successes for nodes that are matched with their original identifier.

# Application: Graph Matching

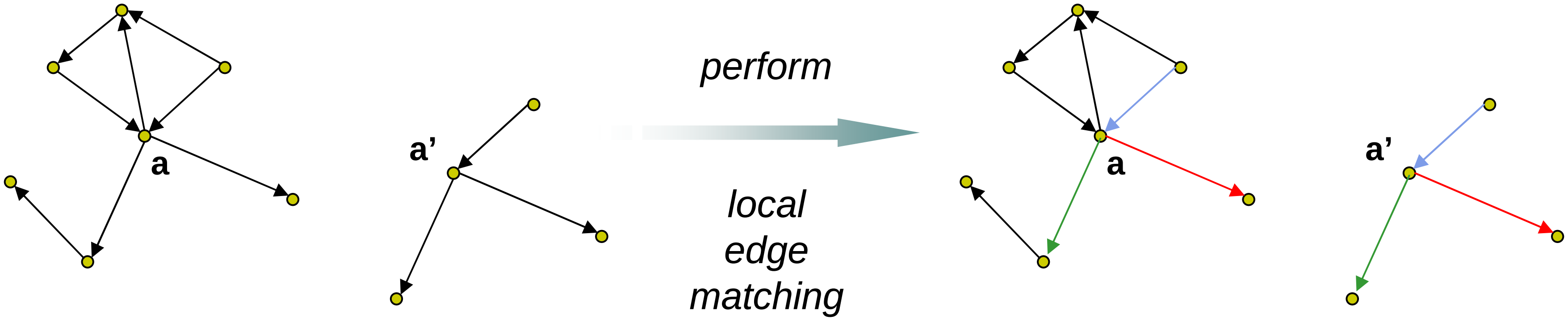



- Task: subgraph matching
  - Generate a random graph, *G*
  - Select a subgraph, *S*
  - Compute the node similarity matrices between *G* and *S*
  - Apply the Hungarian algorithm to `best' match the nodes of *S* to those in *G* by finding a matching the maximizes the sum of matched weights.
  - Record successes for nodes that are matched with their original identifier

→ **Yields a lower bound on the success of the matching process**

# Application: Graph Matching

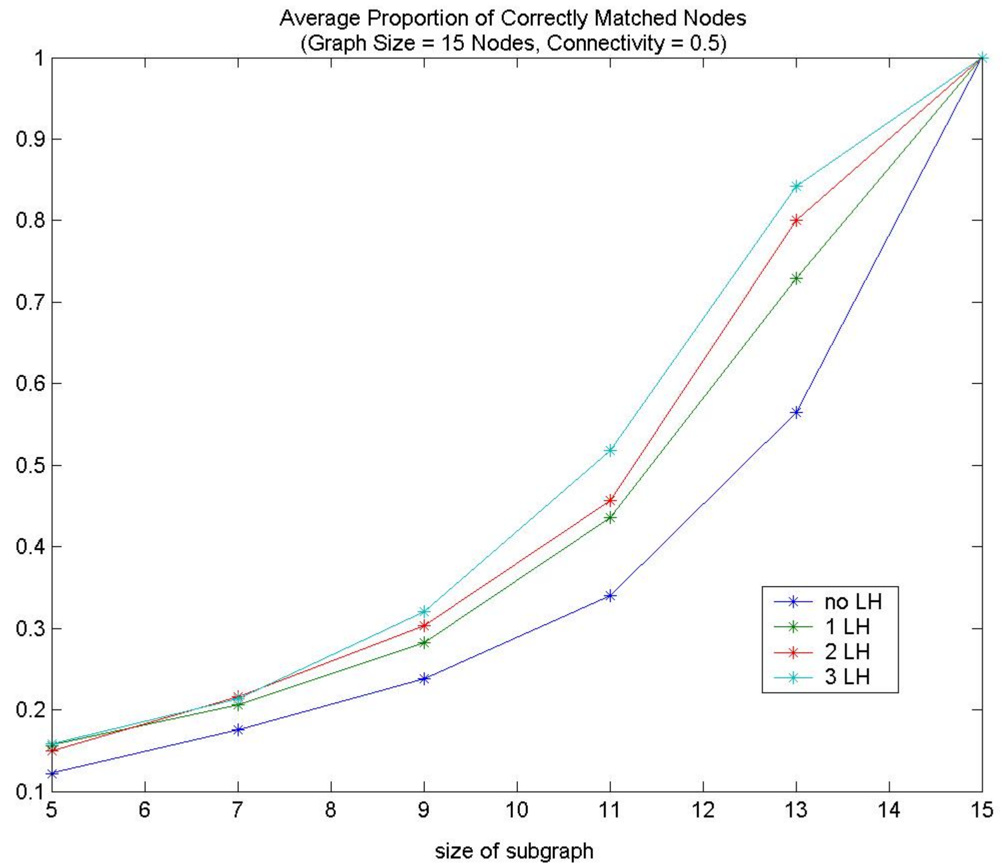
- Using local edge similarity to improve scores:




$x_{aa'}$

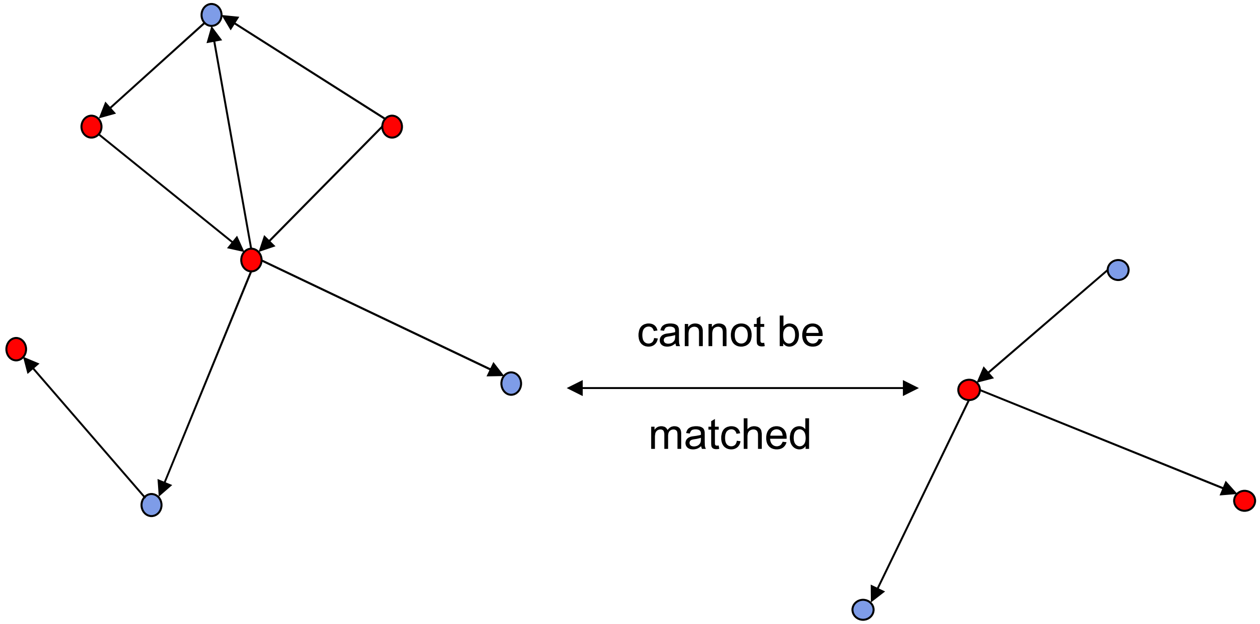
$x_{aa'}^* = x_{aa'} + m_{aa'}$

# Application: Graph Matching

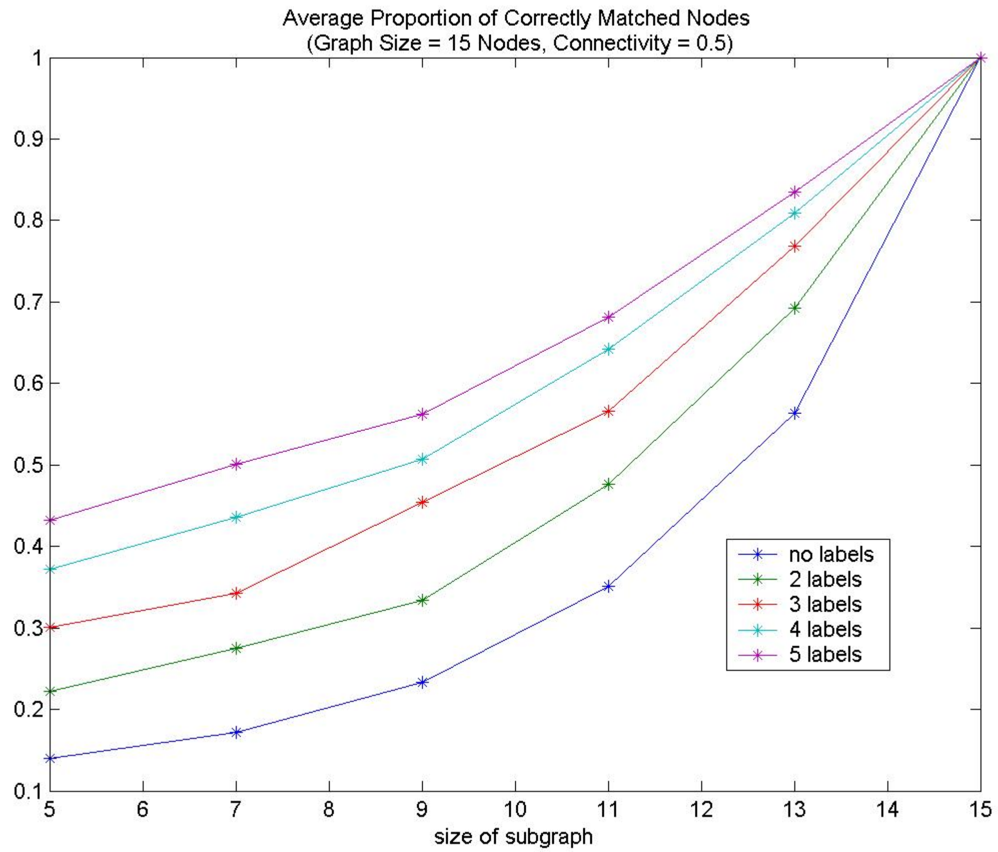


# Application: Graph Matching

- Exploring the impact of node labeling:



# Application: Graph Matching



# Current/future work

- How does graph structure (e.g., cycles, paths, completeness) impact similarity scores?
- What can be inferred about a pair of graphs from a similarity measurement?
- What kinds of tasks is this measure appropriate for?

# Acknowledgments


- George Verghese, MIT
- Sandip Roy, WSU
- Paul Van Dooren, Université catholique de Louvain

Work supported by a NSF Graduate Research Fellowship.