Graph-based Algorithms for Information Retrieval and Natural Language Processing

Tutorial at HLT/NAACL 2006
June 4, 2006

Rada Mihalcea
University of North Texas
rada@cs.unt.edu

Dragomir Radev
University of Michigan
radev@umich.edu

Goal of the Tutorial

- Motivation
  - Graph-theory is a well studied discipline
  - So are the fields of Information Retrieval (IR) and Natural Language Processing (NLP)
  - Often perceived as completely different disciplines
- Goal of the tutorial: provide an overview of methods and applications in IR and NLP that rely on graph-based algorithms, e.g.
  - Graph-based algorithms: graph traversal, min-cut algorithms, random walks
  - Applied to: Web search, text understanding, text summarization, keyword extraction, text clustering

Outline

- Graph-based algorithms
  - Traversal, min-cut/max-flow, matching (Rada)
  - Centrality/ranking, clustering, learning (Drago)
- Information Retrieval applications
  - Web search and text classification (Drago)

BREAK

- Natural Language Processing applications
  - Semantics, subjectivity analysis (Rada)
  - Parsing, summarization (Drago)

Part I

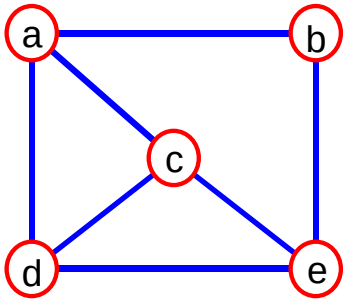
Graph-based Representations and Algorithms
Traversal, min-cut/max-flow, matching

Outline

- graph representations and notations
- algorithms for graph traversal and path finding
- minimum spanning trees
- min-cut/max-flow algorithms
- graph-matching algorithms

What is a Graph?

- A graph G = (V,E) is a data structure composed of:
  V: set of vertices
  E: set of edges connecting the vertices in V
- An edge e = (u,v) is a pair of vertices
- Example:



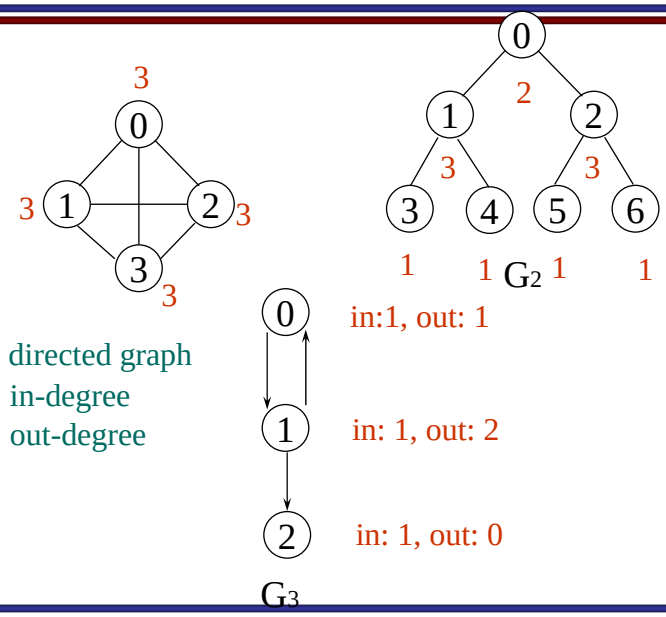
V= {a,b,c,d,e}

E= {(a,b),(a,c),(a,d),
(b,e),(c,d),(c,e),
(d,e)}

Terminology: Adjacent, Incident, Degree

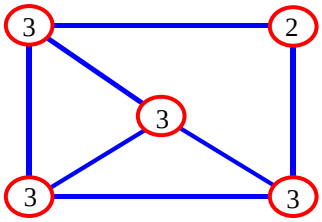
- If (v_0, v_1) is an edge in an undirected graph,
  - v_0 and v_1 are adjacent
  - The edge (v_0, v_1) is incident on vertices v_0 and v_1
- If <v_0, v_1> is an edge in a directed graph
  - v_0 is adjacent to v_1, and v_1 is adjacent from v_0
  - The edge <v_0, v_1> is incident on v_0 and v_1
- The degree of a vertex is the number of edges incident to that vertex
- For directed graphs
  - the in-degree of a vertex *v* is the number of edges that have *v* as the head
  - the out-degree of a vertex *v* is the number of edges that have *v* as the tail

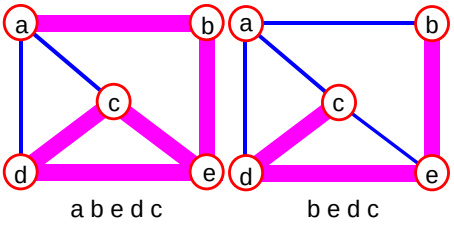
Examples



Terminology: Path

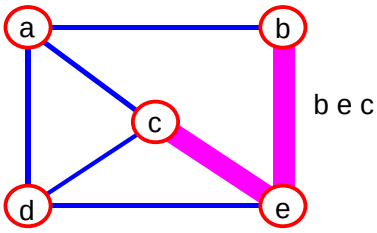
- path: sequence of vertices $v_1, v_2, \ldots v_k$ such that consecutive vertices v_i and v_{i+1} are adjacent.



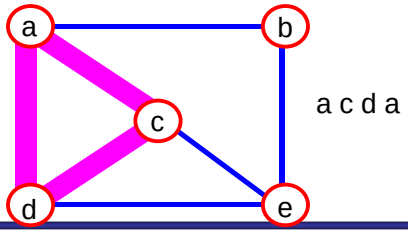


Terminology: Simple Path, Cycle

- simple path: no repeated vertices

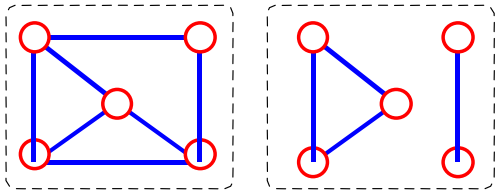


- cycle: simple path, except that the last vertex is the same as the first vertex



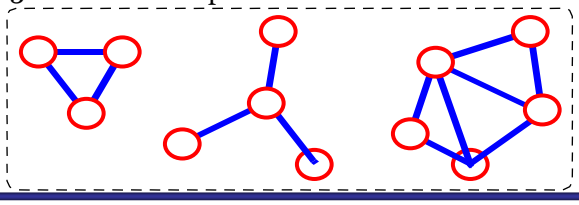
Terminology: Connected Graphs, Subgraphs

- connected graph: any two vertices are connected by some path



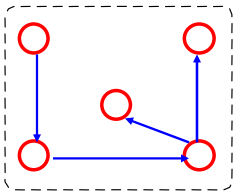
connected not connected

- subgraph: subset of vertices and edges forming a graph
- connected component: maximal connected subgraph. E.g., the graph below has 3 connected components.



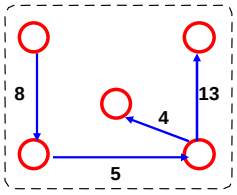
Directed and Undirected Graphs

- An undirected graph is one in which the pair of vertices in a edge is unordered, $(v_0, v_1) = (v_1, v_0)$
- A directed graph is one in which each edge is a directed pair of vertices, $<v_0, v_1> != <v_1, v_0>$

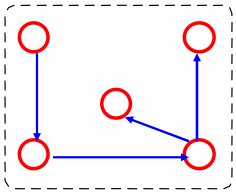


Weighted and Unweighted Graphs

- Edges have / do not have a weight associated with them



weighted



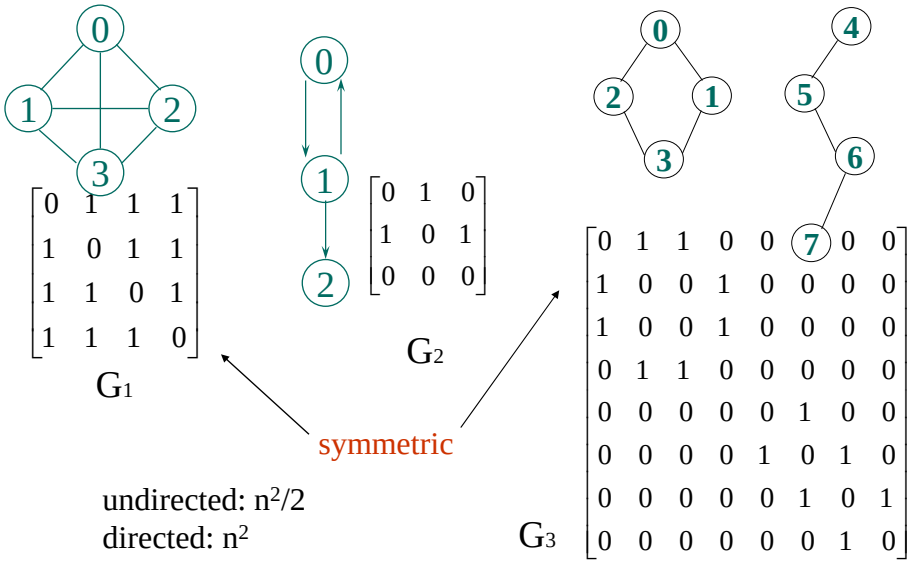
unweighted

Graph Representations(1): Adjacency Matrix

- Let G=(V,E) be a graph with n vertices.
- The adjacency matrix of G is a two-dimensional n by n array, say adj_mat
- If the edge (v_i, v_j) is in E(G), adj_mat[i][j]=1
- If there is no such edge in E(G), adj_mat[i][j]=0
- The adjacency matrix for an undirected graph is symmetric; the adjacency matrix for a digraph need not be symmetric

- Pros: Fast access to adjacent nodes, fast computation of in- and out-degree
- Cons: Memory inefficient

Examples of Adjacency Matrices

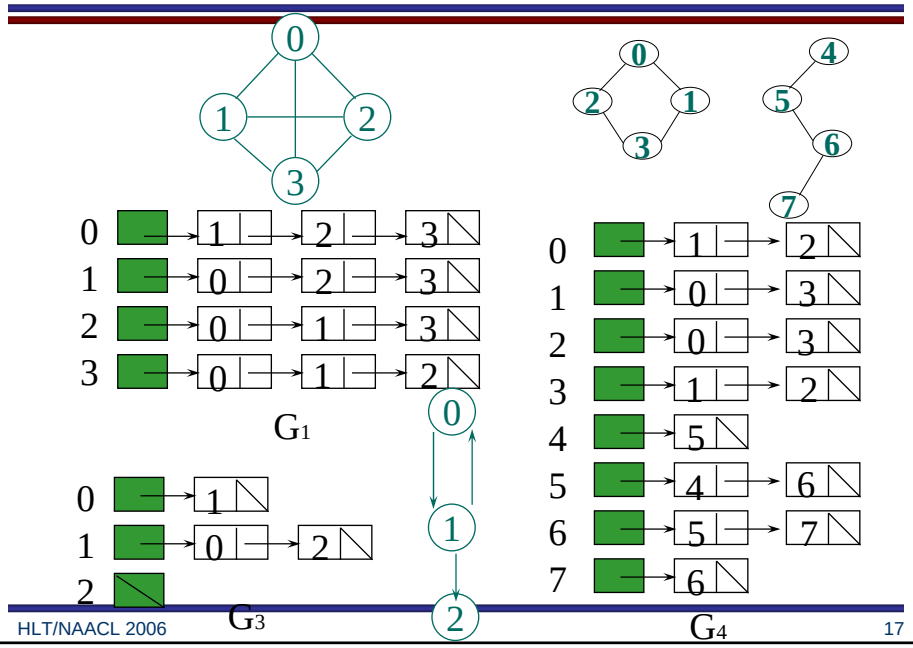


Graph Representations (2): Adjacency Lists

- Each row in an adjacency matrix is represented as a list
- An undirected graph with n vertices and e edges => n head nodes and 2e list nodes

- Pros: More compact representation
  - Memory efficient
- Cons: Sometimes less efficient computation of in- or out-degrees

Examples of Adjacency Lists



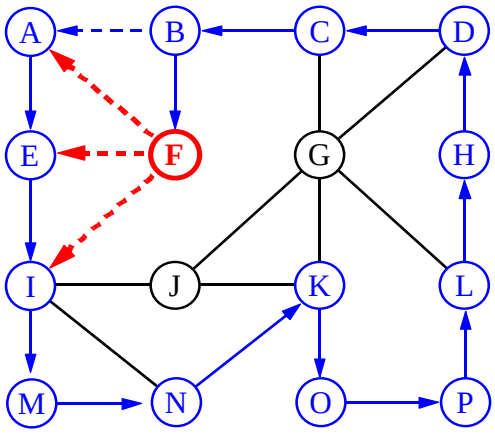
Outline

- graph representations and notations
- algorithms for graph traversal and path finding
- minimum spanning trees
- min-cut/max-flow algorithms
- graph-matching algorithms

Graph Traversal

- Traverse all the nodes in the graph or search for a certain node
- Depth First Search
  - Once a possible path is found, continue the search until the end of the path
- Breadth First Search
  - Start several paths at a time, and advance in each one step at a time

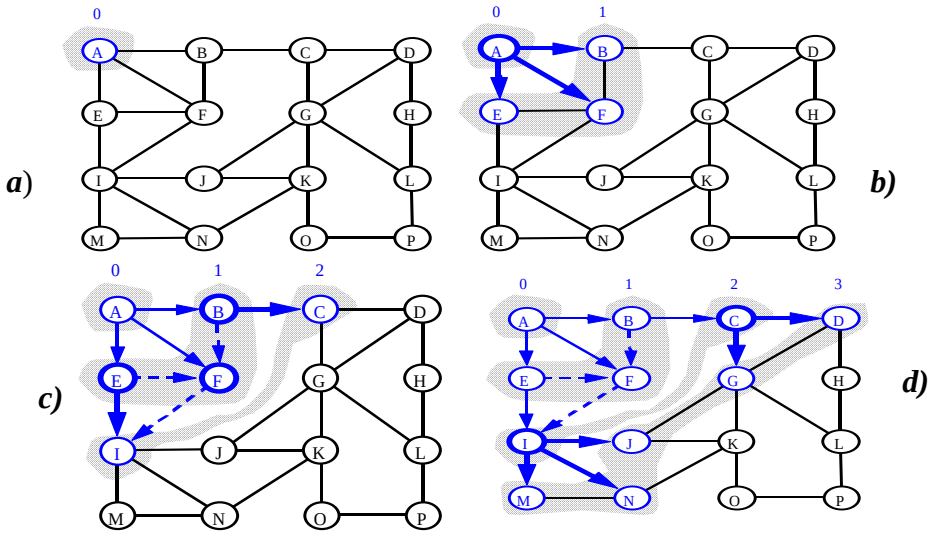
Depth-First Search



Depth-First Search

Algorithm DFS(v);
Input: A vertex v in a graph
Output: A labeling of the edges as "discovery" edges and "backedges"
for each edge *e* incident on *v* **do**
if edge *e* is unexplored **then** let *w* be the other endpoint of *e*
if vertex w is unexplored **then** label *e* as a discovery edge
recursively call **DFS(*w*)**
else label *e* as a backedge

Breadth-First Search



Breadth-First Search

Algorithm BFS(s):

Input: A vertex s in a graph

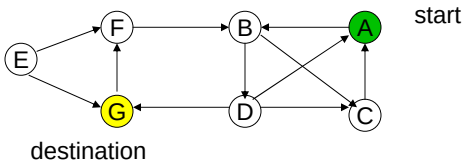
Output: A labeling of the edges as "discovery" edges and "cross edges"

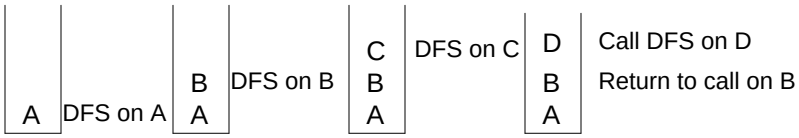
```
initialize container L_0 to contain vertex s
i ← 0
while L_i is not empty do
        create container L_{i+1} to initially be empty
        for each vertex v in L_i do
          if edge e incident on v do
             let w be the other endpoint of e
             if vertex w is unexplored then
                label e as a discovery edge
                insert w into L_{i+1}
             else label e as a cross edge
                i ← i + 1
```

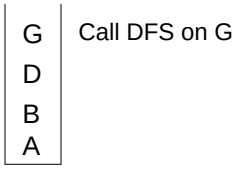

Path Finding

- Find path from source vertex s to destination vertex d
- Use graph search starting at s and terminating as soon as we reach d
  - Need to remember edges traversed
- Use depth – first search
- Use breath – first search

Path Finding with Depth First Search

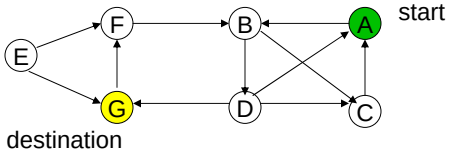


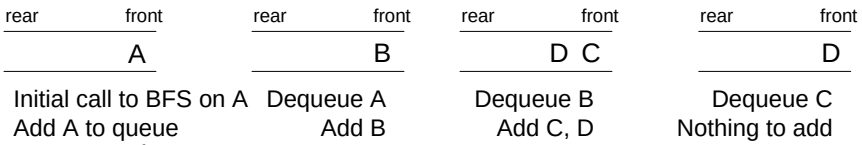


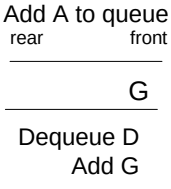


found destination - done
path is implicitly stored in DFS recursion
path is: A, B, D, G

Path Finding with Breadth First Search







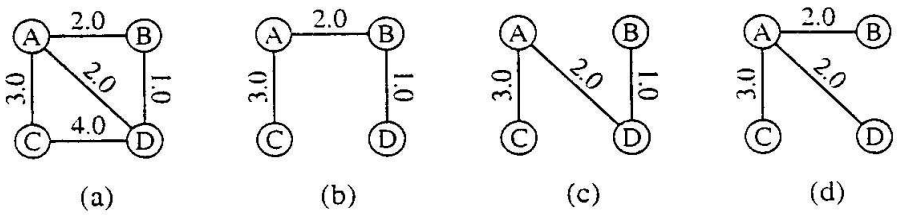
found destination - done
path must be stored separately

Outline

- graph representations and notations
- algorithms for graph traversal and path finding
- minimum spanning trees
- min-cut/max-flow algorithms
- graph-matching algorithms

Minimum Spanning Tree

- A *spanning tree* for a connected, undirected graph G=(V,E)
  - a subgraph of G that is
  - an undirected tree and contains
  - all the vertices of G.
- In a weighted graph G=(V,E,W), the weight of a subgraph
  - the sum of the weights of the edges in the subgraph
- A *minimum spanning tree* for a weighted graph
  - a spanning tree with the minimum weight.



Prim's Minimum Spanning Tree Algorithm

- Select an arbitrary starting vertex (the root)
- Branch out from the tree constructed so far by
  - choosing an edge at each iteration
  - attach the edge to the tree
    - the edge that has minimum weight among all edges that can be attached
  - add to the tree the vertex associated with the edge
- Vertices are divided into three disjoint categories:
  - Tree vertices: in the tree constructed so far,
  - Fringe vertices: not in the tree, but adjacent to some vertex in the tree,
  - Unseen vertices: all others

Prim's Algorithm

Algorithm PrimMinSpanningTree(G)

Initialize all nodes as *unseen*

Select an arbitrary vertex s to start the tree

Reclassify it as *tree*

Reclassify all vertices adjacent to **s** as *fringe*

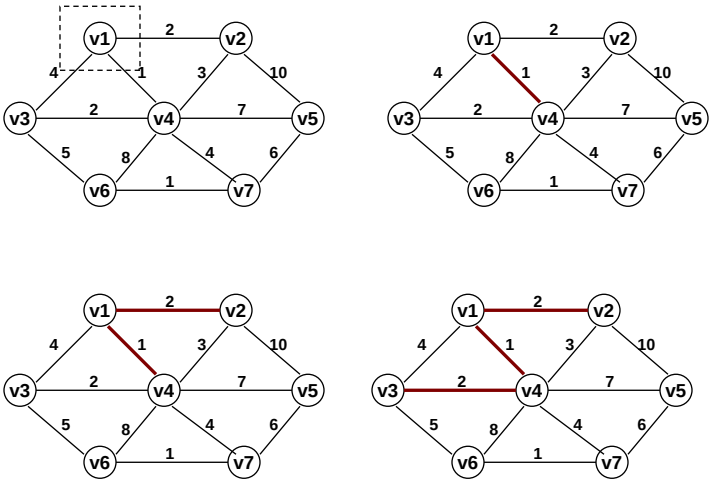
While there are fringe vertices

Select an edge of minimum weight between a tree vertex **t** and a fringe vertex **v**

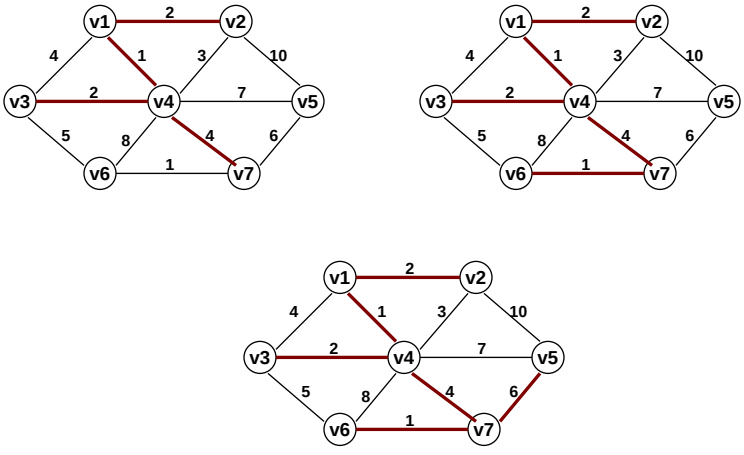
Reclassify v as *tree*; add edge **tv** to the *tree*

Reclassify all unseen vertices adjacent to **v** as *fringe*

Prim's Algorithm



Prim's Algorithm



Properties of Minimum Spanning Trees

- Definition: Minimum spanning tree property
  - Let G be a connected, weighted graph G=(V,E,W), and let T be any spanning tree of G. Suppose that for every edge **vw** of G that is not in T:
    - if uv is added to T, then it creates a cycle
    - uv is a maximum-weight edge on that cycle
  - The tree T is said to have the *minimum spanning tree property*
- A graph can admit several minimum spanning trees
- Lemma: In a connected, weighted graph G = (V, E, W), if T1 and T2 are two spanning trees that have the MST property, then they have the same total weight
- The number of edges in a minimum spanning tree is |V|-1

Outline

- graph representations and notations
- algorithms for graph traversal and path finding
- minimum spanning trees
- min-cut/max-flow algorithms
- graph-matching algorithms

Flow networks

What if weights in a graph are maximum capacities of some flow of material?

Examples:

- Pipe network to transport fluid (e.g., water, oil)
  - Edges – pipes, vertices – junctions of pipes
- Data communication network
  - Edges – network connections of different capacity, vertices – routers (do not produce or consume data just move it)

Concepts (informally):

- *Source* vertex *s* (where material is produced)
- *Sink* vertex *t* (where material is consumed)
- For all other vertices – what goes in must go out
- *Goal: maximum rate of material flow from source to sink*

Formalization

Flow network – $G=(V,E)$

- Directed, each edge has capacity $c(u,v) \geq 0$
- Two special vertices: *source s,* and *sink t*
- For any other vertex v, there is a path $s \rightarrow \ldots \rightarrow v \rightarrow \ldots \rightarrow t$

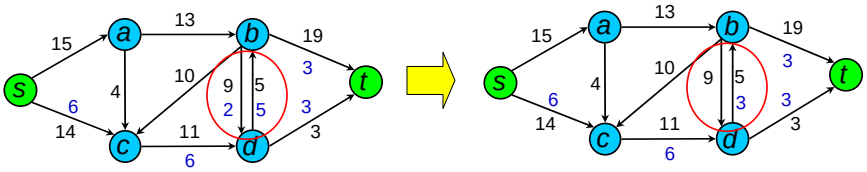
Flow – $f : V \times V \rightarrow R$

- *Capacity constraint*: $\forall u,v \in V$: $f(u,v) \leq c(u,v)$
- *Skew symmetry*: $\forall u,v \in V$: $f(u,v) = -f(v,u)$
- *Flow conservation:* $\forall u \in V - \{s, t\}$: $\sum_{v \in V} f(u,v) = f(u,V) = 0$, or $\sum_{v \in V} f(v,u) = f(V,u) = 0$

Cancellation of flows

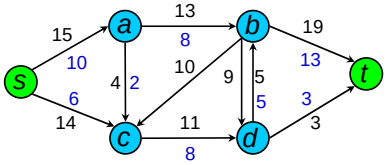
Do we want to have positive flows going in both directions between two vertices?

No! such flows *cancel* (maybe partially) each other.



Maximum flow

Want to maximize the total *value* of the flow f: $|f| = \sum_{v \in V} f(s,v) = f(s,V) = f(V,t)$



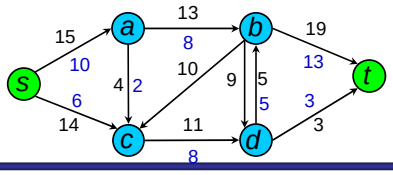
Find a flow of maximum value.

Augmenting path

Idea for the algorithm:

- If we have some flow, and can find a path p from s to t (*augmenting path*), such that there is $a > 0$, and for each edge (u,v) in p we can add a units of flow: $f(u,v) + a \leq c(u,v)$
- then mark down that path, to get a better flow

Augmenting path in this graph?



Ford-Fulkerson method

```
Ford-Fulkerson(G,s,t)
1  initialize flow f to 0 everywhere
2  while there is an augmenting path p do
3     augment flow f along p
4  return f
```

- How do we find augmenting path?
- How much additional flow can we send through that path?

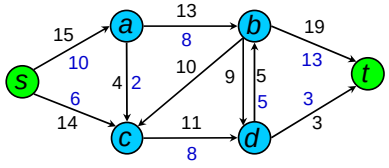
Residual network

How do we find augmenting path?

It is any path in *residual network*:

- Residual capacities: $c_f(u,v) = c(u,v) - f(u,v)$
- Residual network: $G_f=(V,E_f)$, where $E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$

Compute residual network:

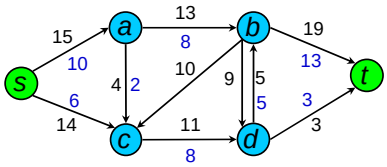


Residual capacity of a path

How much additional flow can we send through an augmenting path?

- *Residual capacity* of a path p in G_f:
  $c_f(p) = \min\{c_f(u,v): (u,v) \in p\}$
- Doing augmentation:
  - $\forall (u,v)$ in p, add $c_f(p)$ to $f(u,v)$ (and subtract from $f(v,u)$)
  - Resulting flow is a valid flow with a larger value.

What is the residual capacity of the path (s,a,b,t)?

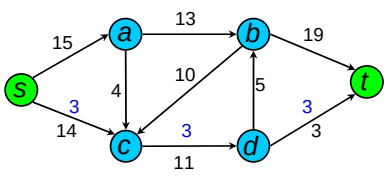


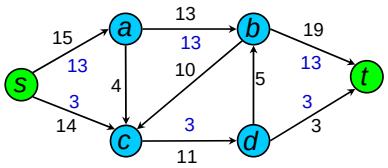
Ford-Fulkerson method, with details

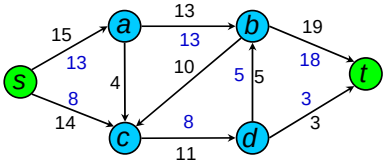
```
Ford-Fulkerson(G,s,t)
1  for each edge (u,v)∈G.E do
2     f(u,v) = f(v,u) = 0
3  while ∃ path p from s to t in residual network G_f do
4     c_f = min{c_f(u,v): (u,v)∈p}
5     for each edge (u,v) in p do
6         f(u,v) = f(u,v) + c_f
7         f(v,u) = -f(u,v)
8  return f
```

The algorithms based on this method differ in how they choose p in step 3.

Ford-Fulkerson method, example





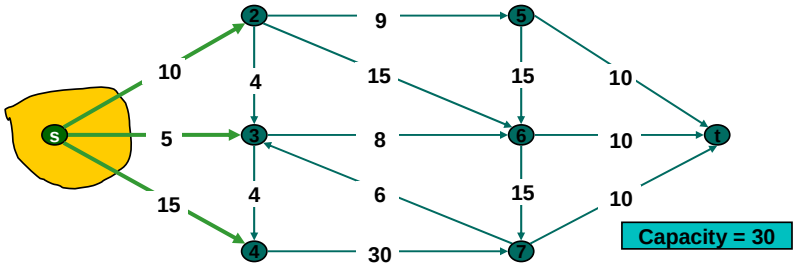


Cuts

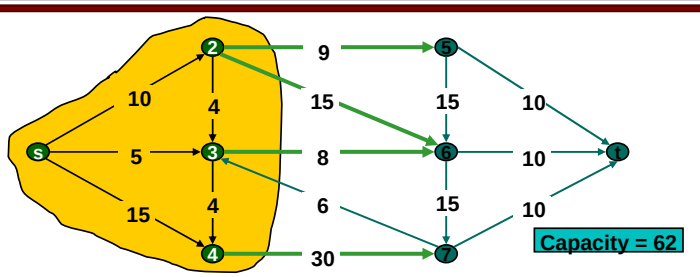
- An s-t cut is a partition (S, T) such that s ∈ S, t ∈ T.
  - The capacity of an s-t cut (S, T) is:

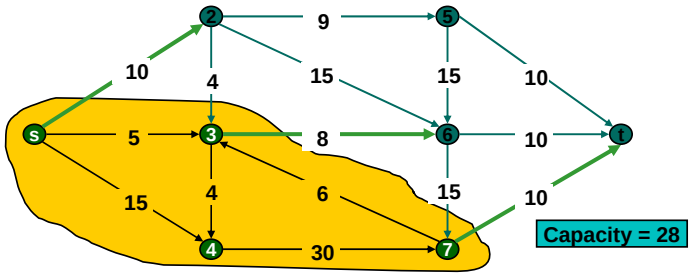
$$\sum_{e \text{ out of } S} u(e) \quad := \quad \sum_{\substack{(v,w) \in E \\ v \in S,\ w \in T}} u(v,w).$$

- Min s-t cut: find an s-t cut of minimum capacity.



Cuts

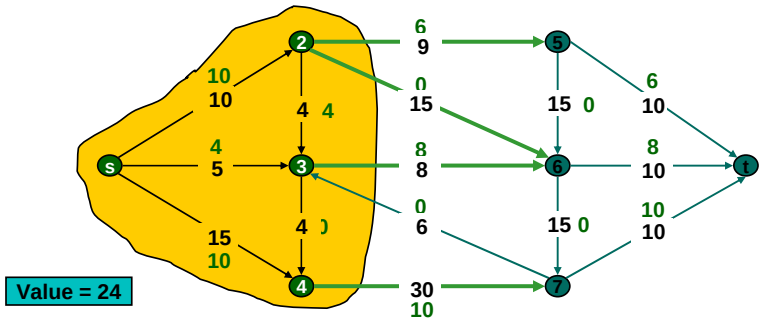




Flows and Cuts

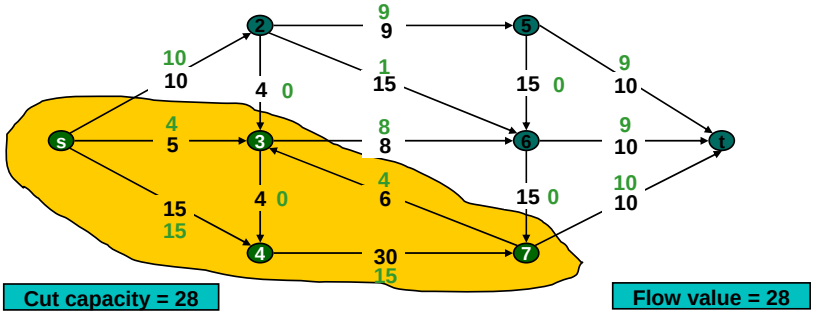
- L1. Let f be a flow, and let (S, T) be a cut. Then, the net flow sent across the cut is equal to the amount reaching t.

$$\sum_{e \text{ out of } S} f(e) - \sum_{e \text{ in to } S} f(e) = \sum_{e \text{ out of } s} f(e) = |f|$$



Max Flow and Min Cut

- Corollary. Let f be a flow, and let (S, T) be a cut. If |f| = cap(S, T), then f is a max flow and (S, T) is a min cut.



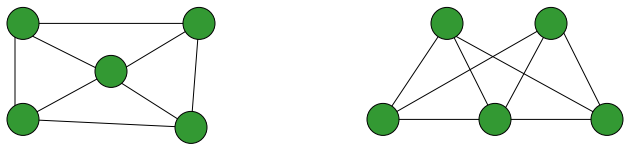
- Max-flow / min-cut theorem (Ford-Fulkerson): In any network, the value of the max flow is equal to the value of the min cut.

Outline

- graph representations and notations
- algorithms for graph traversal and path finding
- minimum spanning trees
- min-cut/max-flow algorithms
- graph-matching algorithms

Graph Isomorphism

- Two graphs G=(V,E) and H=(W,F) are isomorphic if there is a bijective function f: V → W such that for all v, w in V:
  - $\{v,w\} \in E \Leftrightarrow \{f(v),f(w)\} \in F$



- Let G = (V,E), H=(W,F) be graphs with vertex labelings l: V → L, l' → L.
- G and H are isomorphic labeled graphs, if there is a bijective function f: V → W such that
  - For all v, w in V: $\{v,w\} \in E \Leftrightarrow \{f(v),f(w)\} \in F$
  - For all v in V: $l(v) = l'(f(v))$.

Subgraph Isomorphism

- A graph G=(V,E) is isomorphic to a subgraph of H=(W,F) if there is a subgraph H1 of H such that G and H1 are isomorphic

Iterative Vertex Partition Heuristic

- If |V| ≠ |W|, or |E| ≠ |F|, output: no. Done.
- Otherwise, we partition the vertices of G and H into classes, such that
  - Each class for G has a corresponding class for H
  - If f is an isomorphism from G to H, then f(v) belongs to the class, corresponding to the class of v.
- First try: vertices belong to the same class, when they have the same degree.
  - If f is an isomorphism, then the degree of f(v) equals the degree of v for each vertex v.

Scheme

- Start with sequence SG = (A_1) of subsets of G with A_1=V, and sequence SH = (B_1) of subsets of H with B_1=W.
- Repeat
  - Replace A_i in SG by $A_{i1},\ldots,A_{ir}$ and replace B_i in SH by $B_{i1},\ldots,B_{ir}$.
    - $A_{i1},\ldots,A_{ir}$ is partition of A_i
    - $B_{i1},\ldots,B_{ir}$ is partition of B_i
    - For each isormorphism f: v in A_{ij} if and only if f(v) in B_{ij}
- Refinement
  - Count for each vertex in A_i and B_i how many neighbors they have in A_j and B_j for some i, j.
  - Set A_{is} = {v in A_i | v has s neighbors in A_j}
  - Set B_{is} = {v in B_i | v has s neighbors in B_j}
  - If some $|A_i| \neq |B_i|$, then stop: no isomorphism

Part II

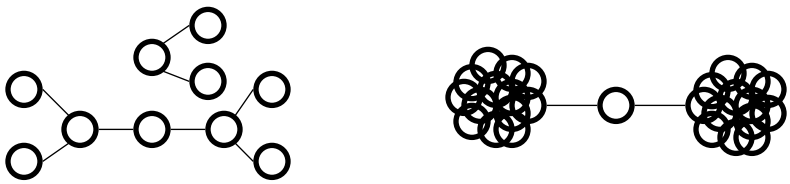
Graph-based Algorithms
Centrality/Ranking, Clustering, Learning

Outline

- Centrality measures
- Harmonic functions
- Eigenvalues and eigenvectors
- Random walks
- Stationary solutions

Graph-based centrality measures

- Centrality defines how important a vertex is in a graph. It can be defined in both directed and undirected graphs
- There are multiple definitions of C(V):
  - Degree centrality: number of V's neighbors
  - Distance centrality: average distance of V to all other nodes
  - Betweenness centrality: how many paths include V
  - Eigenvector centrality: how likely is a random walk on the graph to end in V

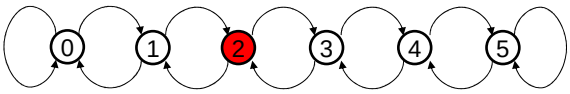


Node ranking algorithms

- The most popular node ranking algorithm is PageRank (named after Larry Page of Google).
- It is based on eigenvector centrality.
- A node's centrality is defined as a degree-weighted average of the centralities of its neighbors.
- Another interpretation is the "random surfer" model: at each step, one can do one of two things:
  - Either click on a link on a page
  - Or jump at random to a different page

Random walks and harmonic functions

- Drunkard's walk:
  - Start at position 0 on a line



- What is the prob. of reaching 5 before reaching 0?
- Harmonic functions:
  - $P(0) = 0$
  - $P(N) = 1$
  - $P(x) = \frac{1}{2}*p(x-1) + \frac{1}{2}*p(x+1)$, for $0<x<N$
  - (in general, replace ½ with the bias in the walk)

The original Dirichlet problem (**)

- Distribution of temperature in a sheet of metal.
- One end of the sheet has temperature t=0, the other end: t=1.
- Laplace's differential equation:

$$\nabla^2 u = u_{xx} + u_{yy} = 0$$

- This is a special (steady-state) case of the (transient) heat equation :

$$k\nabla^2 u = u_t$$

- In general, the solutions to this equation are called harmonic functions.

Learning harmonic functions

- The method of relaxations
  - Discrete approximation.
  - Assign fixed values to the boundary points.
  - Assign arbitrary values to all other points.
  - Adjust their values to be the average of their neighbors.
  - Repeat until convergence.
- Monte Carlo method
  - Perform a random walk on the discrete representation.
  - Compute f as the probability of a random walk ending in a particular fixed point.
- Eigenvector methods
  - Look at the stationary distribution of a random walk

Eigenvectors and eigenvalues

- An eigenvector is an implicit "direction" for a matrix

$$A\vec{v} = \lambda\vec{v}$$

where v (eigenvector) is non-zero, though λ (eigenvalue) can be any complex number in principle

- Computing eigenvalues:

$$\det(A - \lambda I) = 0$$

Eigenvectors and eigenvalues

- Example:

$$A = \begin{pmatrix} -1 & 3 \\ 2 & 0 \end{pmatrix} \quad A - \lambda I = \begin{pmatrix} -1-\lambda & 3 \\ 2 & -\lambda \end{pmatrix}$$

- Det (A-λI) = (-1-λ)*(-λ)-3*2=0
- Then: $\lambda+\lambda^2-6=0$; $\lambda_1=2$; $\lambda_2=-3$
- For $\lambda_1=2$:

$$\begin{pmatrix} -3 & 3 \\ 2 & -2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0$$

- Solutions: $x_1=x_2$

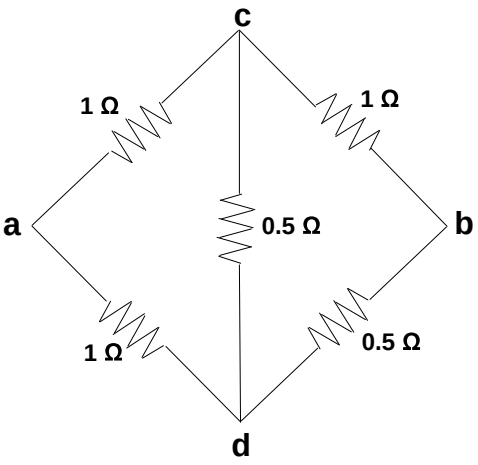
Stochastic matrices

- Stochastic matrices: each row (or column) adds up to 1 and no value is less than 0. Example:

$$A = \begin{pmatrix} 3/8 & 5/8 \\ 1/4 & 3/4 \end{pmatrix}$$

- The largest eigenvalue of a stochastic matrix E is real: $\lambda_1 = 1$.
- For λ_1, the left (principal) eigenvector is p, the right eigenvector = $\mathbf{1}$
- In other words, $G^T p = p$.

Electrical networks and random walks



- Ergodic (connected) Markov chain with transition matrix P

$$P_{xy} = \frac{C_{xy}}{C_x} \qquad C_x = \sum_y C_{xy} \qquad C_{xy} = \frac{1}{R_{xy}}$$

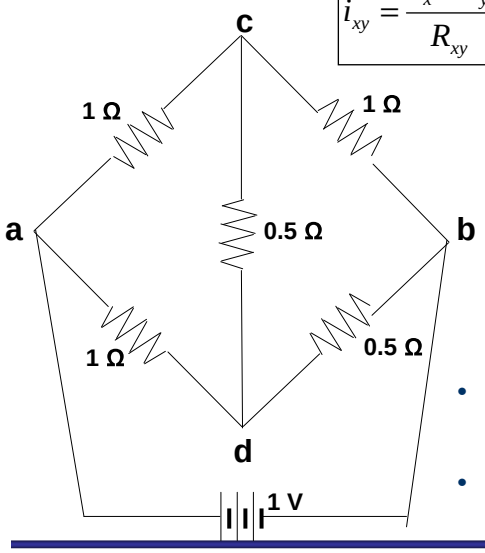
$$\begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{pmatrix} 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{3} & \frac{2}{3} \\ \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{2} \\ \frac{1}{5} & \frac{2}{5} & \frac{2}{5} & 0 \end{pmatrix} \end{array}$$

w=Pw

$$\left(\frac{2}{14}, \frac{3}{14}, \frac{4}{14}, \frac{5}{14}\right)^T$$

From Doyle and Snell 2000

Electrical networks and random walks



$$i_{xy} = \frac{v_x - v_y}{R_{xy}} = (v_x - v_y)C_{xy}$$

$$\sum_y i_{xy} = 0$$

$$v_x = \sum_y \frac{c_{xy}}{c_x} v_y = \sum_y P_{xy} v_y$$

$$v_a = 1$$
$$v_b = 0$$

$$v_c = \frac{1}{4} + \frac{1}{2} v_d = \frac{7}{16}$$
$$v_d = \frac{1}{5} + \frac{2}{5} v_c = \frac{3}{8}$$

- v_x is the probability that a random walk starting at x will reach a before reaching b.
- The random walk interpretation allows us to use Monte Carlo methods to solve electrical circuits.

Energy-based interpretation (**)

- The energy dissipation through a resistor is

$$i_{xy}^2 R_{xy}$$

- Over the entire circuit,

$$E = \frac{1}{2} \sum_{x,y} i_{xy}^2 R_{xy}$$

- The flow from x to y is defined as follows:

$$f_{xy} = -f_{yx} \qquad \sum_y f_{xy} = 0, \text{for y} \neq \text{a, b}$$

- Conservation of energy

$$(w_a - w_b) j_a = \frac{1}{2} \sum_{x,y} (w_x - w_y) j_{xy}$$

Thomson's principle (**)

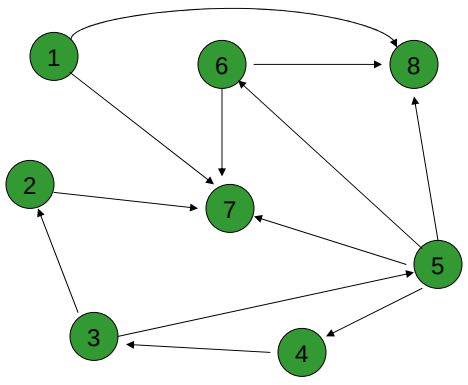
- One can show that:

$$i_{xy}^2 R_{eff} = \frac{1}{2} \sum_{x,y} i_{xy}^2 R_{xy}$$

- The energy dissipated by the unit current flow (for v_b=0 and for i_a=1) is R_{eff}. This value is the smallest among all possible unit flows from a to b (Thomson's Principle)

Markov chains

Graph $G(V,E)$



Transition matrix

	1	2	3	4	5	6	7	8
1							1	1
2							1	
3		1			1			
4			1					
5				1		1	1	1
6							1	1
7								
8								

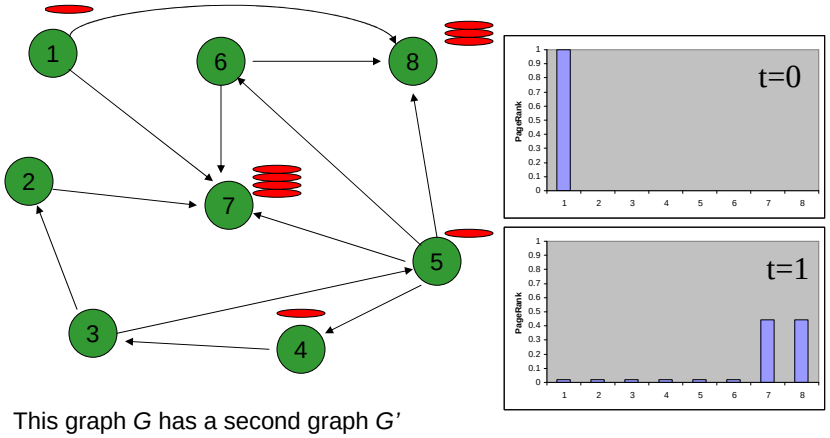
Markov chains

- A homogeneous Markov chain is defined by an initial distribution x and a Markov transition matrix G.
- Path = sequence $(x_0, x_1, \ldots, x_n)$.
  $X_i = x_{i-1} * G$
- The probability of a path can be computed as a product of probabilities for each step i.
- Random walk = find X_j given x_0, G, and j.

Stationary solutions

- The fundamental Ergodic Theorem for Markov chains [Grimmett and Stirzaker 1989] says that the Markov chain G has a stationary distribution p under three conditions:
  - G is stochastic
  - G is irreducible
  - G is aperiodic
- To make these conditions true:
  - All rows of G add up to 1 (and no value is negative)
  - Make sure that G is strongly connected
  - Make sure that G is not bipartite
- Example: PageRank [Brin and Page 1998]: use "teleportation"

An example



This graph *G* has a second graph *G'* *(not drawn)* superimposed on it:
G' is a uniform transition graph.

Computing the stationary distribution

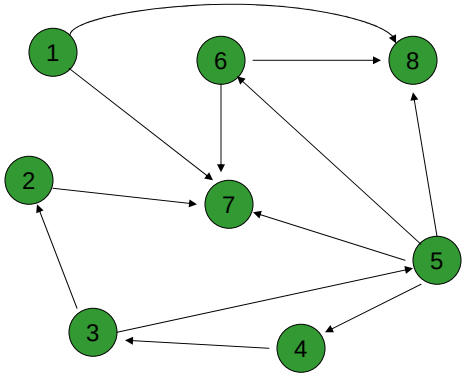
Solution for the stationary distribution

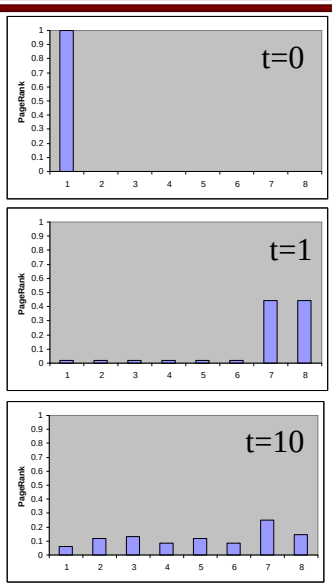
$$p = G^T p$$

$$(I - G^T)p = 0$$

function *PowerStatDist* (*G*):
begin
  $p^{(0)} = u$; (or $p^{(0)} = [1,0,\ldots 0]$)
  i=1;
  repeat
    $p^{(i)} = E^T p^{(i-1)}$
    $L = ||p^{(i)} - p^{(i-1)}||_1$;
    $i = i + 1$;
  until $L < \varepsilon$
  return $p^{(i)}$
end

Example





Part III

Information Retrieval Applications
Web-Search and Text Classification

Outline

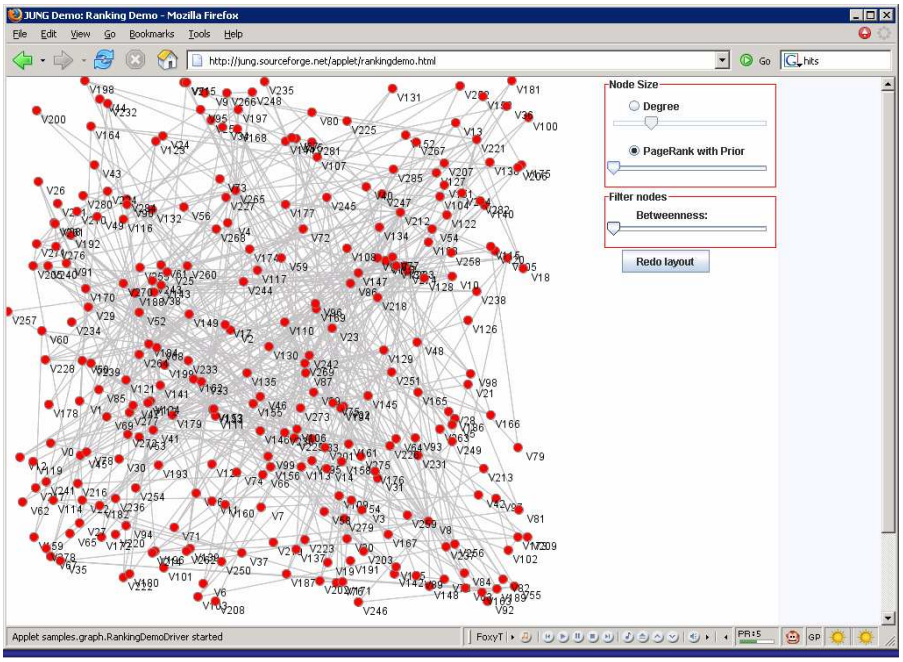
- PageRank
- HITS
- Spectral partitioning

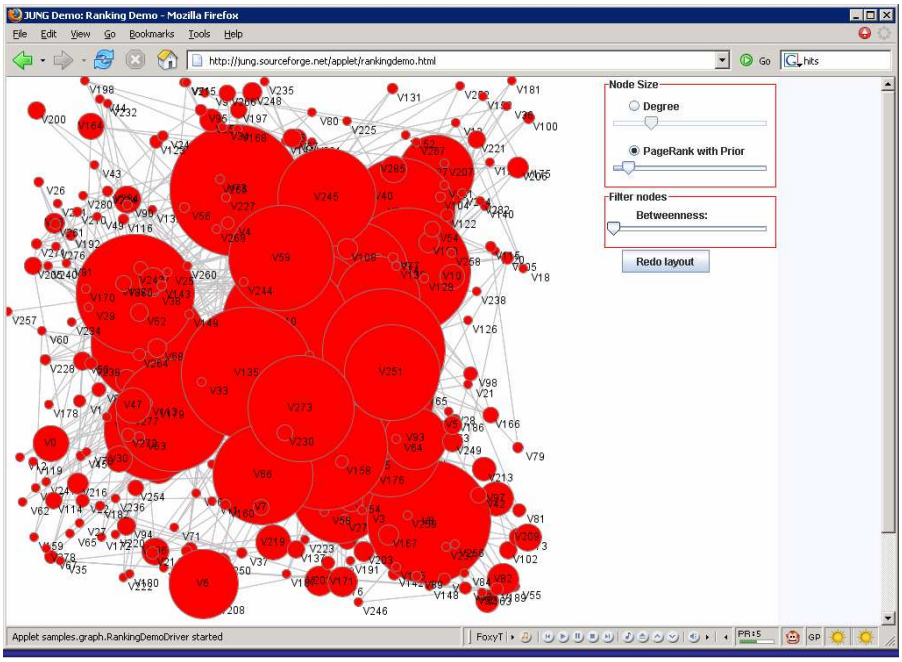
Information retrieval

- Given a collection of documents and a query, rank the documents by similarity to the query.
- On the Web, queries are very short (mode = 2 words).
- Question – how to utilize the network structure of the Web?

PageRank

- Developed at Stanford and allegedly still being used at Google.
- Not query-specific, although query-specific varieties exist.
- In general, each page is indexed along with the anchor texts pointing to it.
- Among the pages that match the user's query, Google shows the ones with the largest PageRank.



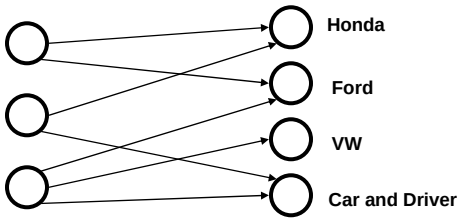


More on PageRank

- Problems
  - PageRank is easy to game.
  - A link farm is a set of pages that (mostly) point to each other.
  - A copy of a "hub" page is created that points to the root page of each site. In exchange, the root page of each participating site should point to the "hub".
  - Thus each root page gets n links from each of the n copies of the hub.
  - Link farms are not hard to detect in principle although a number of variants exist which make the problem actually more difficult.
- Modifications
  - Personalized PageRank (biased random walk)
  - Topic-based (Haveliwala 2002): use a topical source such as DMOZ and compute PageRank separately for each topic.

HITS

- Hypertext-induced text selection.
- Developed by Jon Kleinberg and colleagues at IBM Almaden as part of the CLEVER engine.
- HITS is query-specific.
- Hubs and authorities, e.g. collections of bookmarks about cars vs. actual sites about cars.



HITS

- Each node in the graph is ranked for *hubness* (h) and *authoritativeness* (a).
- Some nodes may have high scores on both.
- Example authorities for the query "java":
  - www.gamelan.com
  - java.sun.com
  - digitalfocus.com/digitalfocus/... (The Java developer)
  - lightyear.ncsa.uiuc.edu/~srp/java/javabooks.html
  - sunsite.unc.edu/javafaq/javafaq.html

HITS

- HITS algorithm:
  - obtain root set (using a search engine) related to the input query
  - expand the root set by radius one on either side (typically to size 1000-5000)
  - run iterations on the hub and authority scores together

$$\underline{a}' = G^T \underline{h} \qquad \underline{h}' = Ga$$

  - report top-ranking authorities and hubs
- Eigenvector interpretation:

$$\underline{a} = \omega(G^T G) \qquad \underline{h} = \omega(GG^T) \qquad \underline{p} = \omega(G)$$

HITS

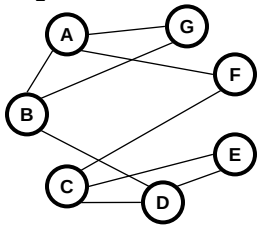
- HITS is now used by Ask.com and Teoma.com .
- It can also be used to identify communities (e.g., based on synonyms as well as controversial topics.
- Example for “jaguar”
  - Principal eigenvector gives pages about the animal
  - The positive end of the second nonprincipal eigenvector gives pages about the football team
  - The positive end of the third nonprincipal eigenvector gives pages about the car.
- Example for “abortion”
  - The positive end of the second nonprincipal eigenvector gives pages on “planned parenthood” and “reproductive rights”
  - The negative end of the same eigenvector includes “pro-life” sites.

Spectral algorithms

- The spectrum of a matrix is the list of all eigenvectors of a matrix.
- The eigenvectors in the spectrum are sorted by the absolute value of their corresponding eigenvalues.
- In spectral methods, eigenvectors are based on the Laplacian of the original matrix.

Laplacian matrix

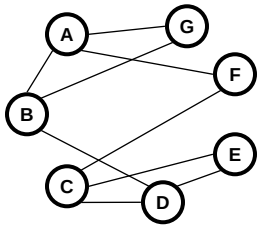
- The Laplacian L of a matrix is a symmetric matrix.
- L = D – G, where D is the degree matrix corresponding to G.
- Example:



	A	B	C	D	E	F	G
A	**3**	-1	0	0	0	-1	-1
B	-1	**3**	0	-1	0	0	-1
C	0	0	**3**	-1	-1	-1	0
D	0	-1	-1	**3**	-1	0	0
E	0	0	-1	-1	**2**	0	0
F	-1	0	-1	0	0	**2**	0
G	-1	-1	0	0	0	0	**2**

Fiedler vector

- The Fiedler vector is the eigenvector of L(G) with the second smallest eigenvalue.



A	-0.3682	C1
B	-0.2808	C1
C	0.3682	C2
D	0.2808	C2
E	0.5344	C2
F	0.0000	?
G	-0.5344	C1

Spectral bisection algorithm

- Compute $\lambda 2$
- Compute the corresponding v2
- For each node n of G
  - If v2(n) < 0
    - Assign n to cluster C1
  - Else if v2(n) > 0
    - Assign n to cluster C2
  - Else if v2(n) = 0
    - Assign n to cluster C1 or C2 at random

Part IV

Natural Language Processing Applications
Semantics

Outline

- word sense disambiguation
- entity disambiguation
- thesaurus construction / semantic classes
- textual entailment
- sentiment classification

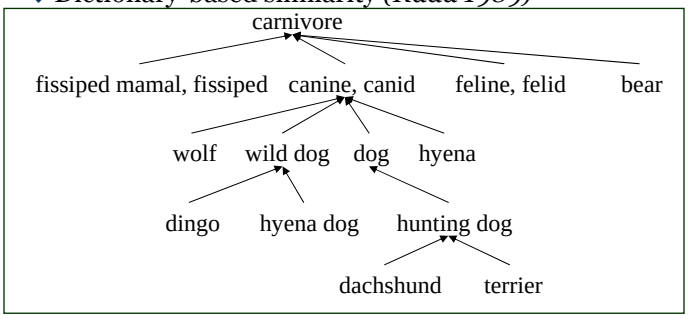
Word Sense Disambiguation

- The problem of selecting a sense for a word from a set of predefined possibilities.
  - Sense Inventory usually comes from a dictionary or thesaurus.
  - Knowledge intensive methods, supervised learning, and (sometimes) bootstrapping approaches
- Word polysemy (with respect to a dictionary)
  - Ex: "chair" – furniture or person
  - Ex: "child" – young person or human offspring
- Determine which sense of a word is used in a specific sentence

  "Sit on a chair" "Take a seat on this chair"
  "The chair of the Math Department" "The chair of the meeting"

Graph-based Solutions for WSD

- Use information derived from dictionaries / semantic networks to construct graphs
- Build graphs using measures of "similarity"
  - Similarity determined between pairs of concepts, or between a word and its surrounding context
    - Distributional similarity *(Lee 1999) (Lin 1999)*
    - Dictionary-based similarity *(Rada 1989)*



Semantic Similarity Metrics

- Input: two concepts (same part of speech)
- Output: similarity measure
- E.g. *(Leacock and Chodorow 1998)*

$$Similarity\ (C_1, C_2) = -\log\left(\frac{Path(C_1, C_2)}{2D}\right)$$

where D is the taxonomy depth

  - E.g. Similarity(wolf,dog) = 0.60 Similarity(wolf,bear) = 0.42
- Other metrics:
  - Similarity using information content *(Resnik 1995) (Lin 1998)*
  - Similarity using gloss-based paths across different hierarchies *(Mihalcea and Moldovan 1999)*
  - Conceptual density measure between noun semantic hierarchies and current context *(Agirre and Rigau 1995)*

Lexical Chains for WSD

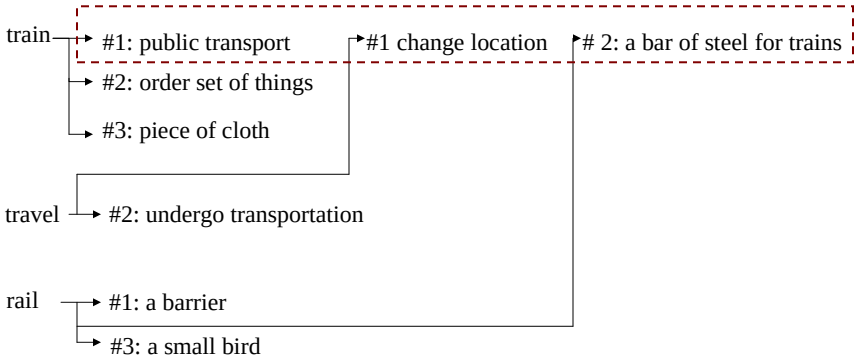
- Apply measures of semantic similarity in a global context
- Lexical chains *(Hirst and St-Onge 1988), (Haliday and Hassan 1976)*
- "*A lexical chain is a sequence of semantically related words, which creates a context and contributes to the continuity of meaning and the coherence of a discourse*"

Algorithm for finding lexical chains:

1. Select the candidate words from the text. These are words for which we can compute similarity measures, and therefore most of the time they have the same part of speech.
2. For each such candidate word, and for each meaning for this word, find a chain to receive the candidate word sense, based on a semantic relatedness measure between the concepts that are already in the chain, and the candidate word meaning.
3. If such a chain is found, insert the word in this chain; otherwise, create a new chain.

Lexical Chains

A very long **train traveling** along the **rails** with a constant **velocity** v in a certain **direction** …

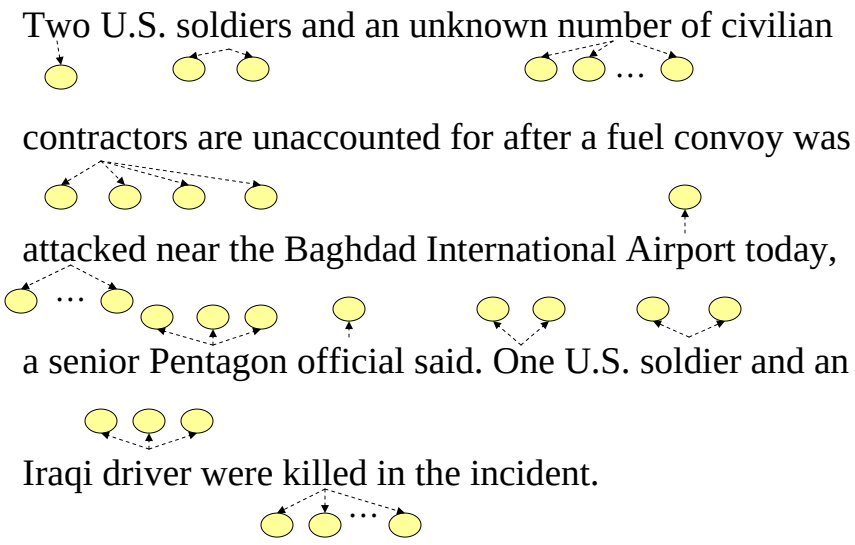


Lexical Chains for WSD

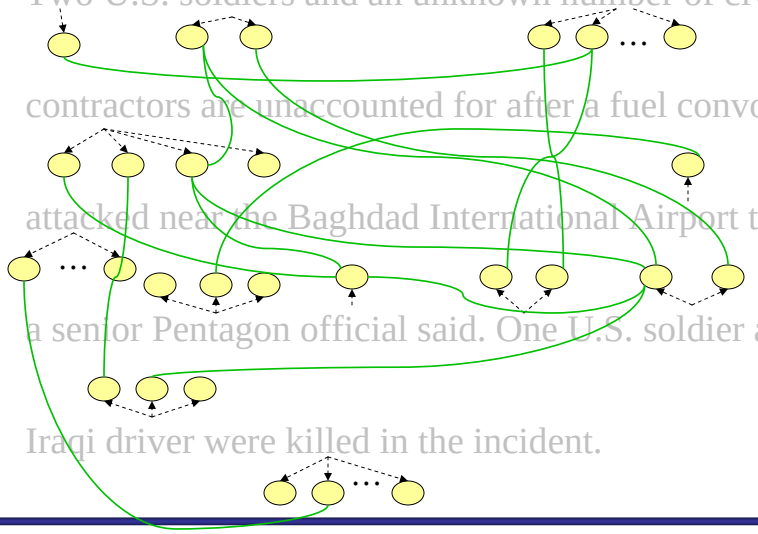
- Identify lexical chains in a text
  - Usually target one part of speech at a time
- Identify the meaning of words based on their membership to a lexical chain
- Evaluation:
  - *(Galley and McKeown 2003)* lexical chains on 74 SemCor texts give 62.09%
  - *(Mihalcea and Moldovan 2000)* on five SemCor texts give 90% with 60% recall
    - lexical chains "anchored" on monosemous words

Graph-based Ranking on Semantic Networks

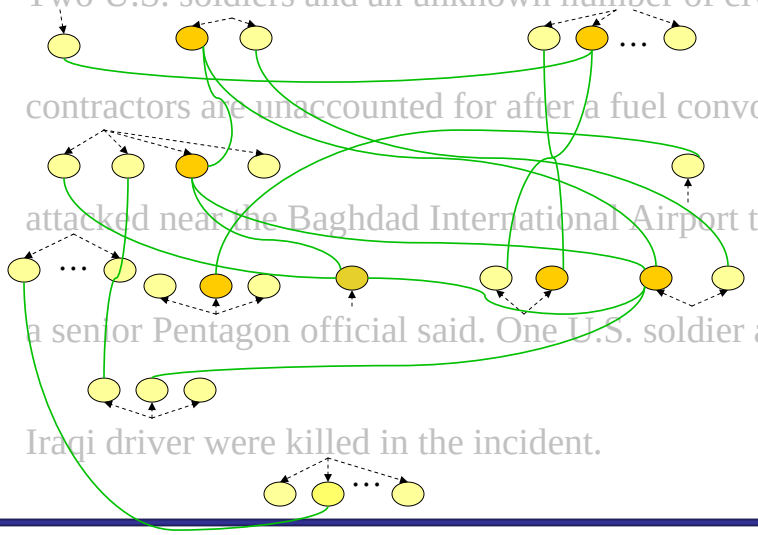
- Goal: build a semantic graph that represents the meaning of the text
- Input: Any open text
- Output: Graph of meanings (synsets)
  - "importance" scores attached to each synset
  - relations that connect them
- Models text cohesion
  - *(Halliday and Hasan 1979)*
  - From a given concept, follow "links" to semantically related concepts
- Graph-based ranking identifies the most recommended concepts



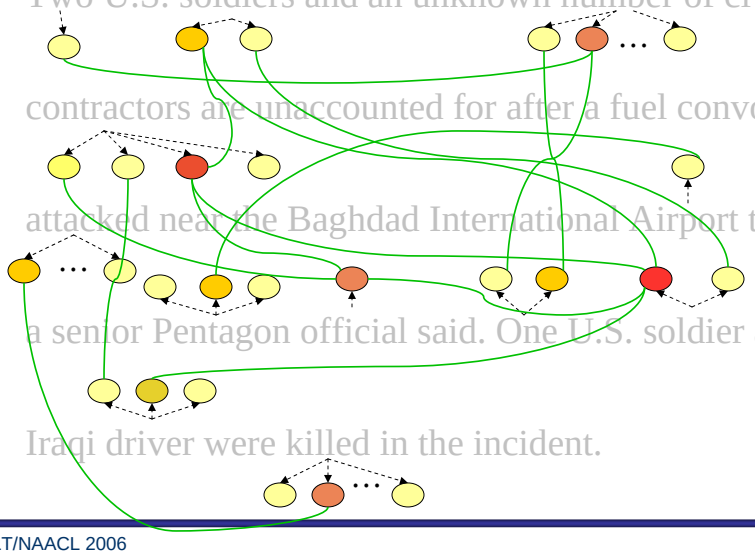
Two U.S. soldiers and an unknown number of civilian contractors are unaccounted for after a fuel convoy was attacked near the Baghdad International Airport today, a senior Pentagon official said. One U.S. soldier and an Iraqi driver were killed in the incident.



Two U.S. soldiers and an unknown number of civilian contractors are unaccounted for after a fuel convoy was attacked near the Baghdad International Airport today, a senior Pentagon official said. One U.S. soldier and an Iraqi driver were killed in the incident.



Two U.S. soldiers and an unknown number of civilian contractors are unaccounted for after a fuel convoy was attacked near the Baghdad International Airport today, a senior Pentagon official said. One U.S. soldier and an Iraqi driver were killed in the incident.



Main Steps

- *Step 1*: Preprocessing
  - SGML parsing, text tokenization, part of speech tagging, lemmatization
- Step 2: Assume any possible meaning of a word in a text is potentially correct
  - Insert all corresponding synsets into the graph
- *Step 3*: Draw connections (edges) between vertices
- *Step 4*: Apply the graph-based ranking algorithm
  - PageRank, HITS, Positional power

Semantic Relations

- Main relations provided by WordNet
  - ISA (hypernym/hyponym)
  - PART-OF (meronym/holonym)
  - causality
  - attribute
  - nominalizations
  - domain links
- Edges (connections)
  - directed / undirected
  - best results with undirected graphs
- Output: Graph of concepts (synsets) identified in the text
  - "importance" scores attached to each synset
  - relations that connect them

Word Sense Disambiguation

- Rank the synsets/meanings attached to each word
- Unsupervised method for semantic ambiguity resolution of all words in unrestricted text *(Mihalcea et al. 2004) (Mihalcea 2005)*
- Related algorithms:
  - Lesk
  - Baseline (most frequent sense / random)
- Hybrid:
  - Graph-based ranking + Lesk
  - Graph-based ranking + Most frequent sense
- Evaluation
  - "Informed" (with sense ordering)
  - "Uninformed" (no sense ordering)
- Data
  - Senseval-2 all words data (three texts, average size 600)
  - SemCor subset (five texts: law, sports, debates, education, entertainment)

Evaluation

	Size(words)	Random	Lesk	TextRank	TextRank+Lesk
SemCor					
law	825	37.12%	39.62%	46.42%	49.36%
sports	808	29.95%	33.00%	40.59%	46.18%
education	898	37.63%	41.33%	46.88%	52.00%
debates	799	40.17%	42.38%	47.80%	50.52%
entertainment	802	39.27%	43.05%	43.89%	49.31%
Avg.	826	36.82%	39.87%	45.11%	49.47%
Senseval-2					
d00	471	28.97%	43.94%	43.94%	47.77%
d01	784	45.47%	52.65%	54.46%	57.39%
d02	514	39.24%	49.61%	54.28%	56.42%
Avg.	590	37.89%	48.73%	50.89%	53.86%
Average (all)	740	37.22%	43.19%	47.27%	51.16%

"uninformed" (no sense order)

Evaluation

	Size(words)	MFS	Lesk	TextRank	TextRank+Lesk
SemCor					
law	825	69.09%	72.65%	73.21%	73.97%
sports	808	57.30%	64.21%	68.31%	68.31%
education	898	64.03%	69.33%	71.65%	71.53%
debates	799	66.33%	70.07%	71.14%	71.67%
entertainment	802	59.72%	64.98%	66.02%	66.16%
Avg.	826	63.24%	68.24%	70.06%	70.32%
Senseval-2					
d00	471	51.70%	53.07%	58.17%	57.74%
d01	784	60.80%	64.28%	67.85%	68.11%
d02	514	55.97%	62.84%	63.81%	64.39%
Avg.	590	56.15%	60.06%	63.27%	63.41%
Average (all)	740	60.58%	65.17%	67.51%	67.72%

"informed" (sense order integrated)

Outline

- word sense disambiguation
- entity disambiguation
- thesaurus construction / semantic classes
- textual entailment
- sentiment classification

Ambiguous Entitites

- Name ambiguity in research papers:
  - David S. Johnson, David Johnson, D. Johnson
  - David Johnson (Rice), David Johnson (AT & T)
- Similar problem across entities:
  - Washington (person), Washington (state), Washington (city)
- Number ambiguity in texts:
  - quantity: e.g. 100 miles
  - time: e.g. 100 years
  - money: e.g. 100 euro
  - misc.: anything else
- Can be modeled as a clustering problem

Name Disambiguation

- Extract attributes for each person – e.g. for research papers:
  - Co-authors
  - Paper titles
  - Venue titles
- Each word in these attribute sets constitutes a binary feature
- Apply a weighting scheme:
  - E.g. normalized TF, TF/IDF
- Construct a vector for each occurrence of a person name
- *(Han et al., 2005)*

Spectral Clustering

- Apply k-way spectral clustering to name data sets
- Two data sets:
  - DBLP: authors of 400,000 citation records – use top 14 ambiguous names
  - Web-based data set: 11 authors named "J. Smith" and 15 authors named "J. Anderson", in a total of 567 citations
- Clustering evaluated using confusion matrices
  - Disambiguation accuracy = sum of diagonal elements A[i,i] divided by the sum of all elements in the matrix

Name Disambiguation – Results

- DBLP data set:

Name	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
A. Gupta	51.1	51.3	51.0	51.0	52.5	50.7	51.2	50.1	50.8	53.9
A. Kumar	55.6	54.6	58.2	61.6	55.7	63.0	63.4	62.6	64.5	64.3
C. Chen	50.4	45.2	43.3	46.8	45.6	46.3	45.8	47.0	47.3	50.6
D. Johnson	66.0	70.0	66.9	69.1	70.4	72.4	72.0	73.3	77.6	79.1
J. Lee	56.1	53.4	52.2	53.1	51.7	53.8	56.9	56.0	56.4	56.2
J. Martin	82.0	74.5	73.6	88.3	89.7	90.3	92.6	91.1	91.8	96.8
J. Robinson	68.2	56.0	62.4	60.4	55.4	59.3	56.2	47.6	47.5	39.2
J. Smith	67.7	70.8	70.6	72.5	73.3	73.7	77.2	77.9	77.0	77.4
K. Tanaka	63.9	62.7	62.4	62.1	67.7	63.9	56.7	69.8	62.5	50.8
M. Brown	63.0	68.5	67.1	80.2	78.7	73.3	81.0	87.1	86.0	87.0
M. Jones	71.0	66.8	76.3	76.4	70.0	68.0	79.4	77.3	76.6	70.6
M. Miller	66.5	67.2	75.2	74.0	69.5	69.3	68.9	69.1	67.4	67.4
S. Lee	53.4	49.2	46.9	48.9	48.5	48.5	49.3	49.8	51.1	50.4
Y. Chen	46.1	43.6	46.1	44.2	46.2	45.9	47.8	46.4	47.3	45.5
Mean	61.5	59.6	60.9	63.5	62.5	62.7	64.2	64.7	64.6	63.5
Std	9.4	9.9	11.0	13.2	13.0	12.4	14.0	14.9	14.7	16.8

- Web-based data set
  - 11 "J.Smith": 84.7% (k-means 75.4%)
  - 15 "J.Anderson": 71.2% (k-means 67.2%)

Outline

- word sense disambiguation
- entity disambiguation
- thesaurus construction / semantic classes
- textual entailment
- sentiment classification

Automatic Thesaurus Generation

- Idea: Use an (online) traditional dictionary to generate a graph structure, and use it to create thesaurus-like entries for all words (stopwords included)
  - *(Jannink and Wiederhold, 1999)*
- For instance:
  - *Input*: the 1912 Webster's Dictionary
  - *Output*: a repository with rank relationships between terms
- The repository is comparable with handcrafted efforts such as WordNet or other automatically built thesauruses such as MindNet *(Dolan et al. 1993)*

Algorithm

1. Extract a directed graph from the dictionary
   - e.g. relations between head-words and words included in definitions
2. Obtain the relative measure of arc importance
3. Rank the arcs with ArcRank

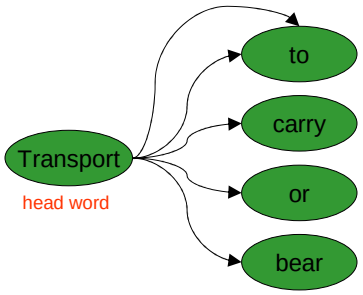
   (graph-based algorithm for edge ranking)

Algorithm (cont.)

1. Extract a directed graph from the Dictionary

Directed Graph

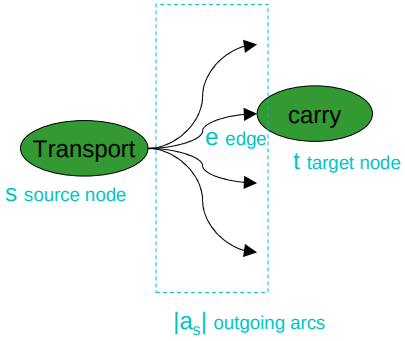
Transport. To carry or bear from one place to another;…



- One arc from each headword to all words in the definition.
- Potential problems as: syllable and accent markers in head words, misspelled head words, accents, special characters, mistagged fields, common abbreviations in definitions, steaming, multi-word head words, undefined words with common prefixes, undefined hyphenated words.
- Source words: words never used in definitions
- Sink words: undefined words

Algorithm (cont.)

2. Obtain the relative measure of arc importance



$$r_e = \frac{p_s / |a_s|}{p_t}$$

- Where:
  - r_e is the rank of the edge
  - p_s is the rank of the source node
  - p_t is the rank of the target node
  - $|a_s|$ is the number of outgoing edges

$$r_{s,t} = \sum_{e=1}^{m} \frac{p_s / |a_s|}{p_t}$$

For more than 1 edge (m) between s and t

Algorithm (cont.)

3. Rank the arcs using ArcRank
- Rank the importance of arcs with respect to source and target nodes
- It promotes arcs that are important in both endpoints

ArcRank
Input: triples (source s, target t, importance $v_{s,t}$)

1. given source s and target t nodes
2. at s, sort $v_{s,tj}$ and rank arcs $rs(v_{s,tj})$
3. at t, sort $v_{sj,t}$ and rank arcs $rt(c_{sj},t)$
4. compute ArcRank: mean(rs (vs,t), rt(cs,t))
5. Rank Arcs input: sorted arc importance
   - {0.9, 0.75, 0.75, 0.75, 0.6, 0.5, …. , 0.1} sample values
   - {1, 2, 2, 2, …} equal values take same rank
   - {1, 2, 2, 2, 3, …} number ranks consecutively

Results

An automatically built thesaurus starting with Webster

Webster	**WordNet**	**MindNet**
96,800 terms 112,897 distinct words	99,642 terms 173,941 word senses	159,000 head words 713,000 relationships between headwords
error rates: <1% of original input <0.05% incorrect arcs (hyphenation) <0.05% incorrect terms (spelling) 0% artificial terms	**error rates:** ~0.1% inappropriate classifications ~1-10% artificial & repeated terms	*(not publicly available)*

Results Analysis

The Webster's repository

PROS

- It has a very general structure
- It can also address stopwords
- It has more relationships than WordNet
- It allows for any relationships between words (not only within a lexical category)
- It is more "natural": it does not include artificial concepts such as non-existent words and artificial categorizations

CONS

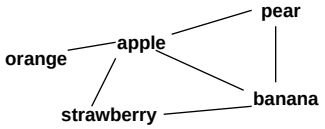
- The type of relationships is not always evident
- The accuracy increases with the amount of data, nevertheless, the dictionary contains sparse definitions
- It only distinguishes senses based on usage, not grammar
- It is less precise than other thesauruses (e.g. WordNet)

Automatic Thesaurus Generation: Semantic Classes

- Automatic unsupervised lexical acquisition
  - E.g. identify clusters of semantically related words
- Use syntactic relations between words to generate large graphs, starting with raw corpora
  - Syntactic relations that can be gathered from POS tagged data
    - Verb Noun
    - Noun *and* Noun
    - Adjective Noun
    - …
  - Implicit ambiguity resolution through graph relations
  - Incremental cluster-building algorithm
  - *(Widdows & Dorow, 2002)*

Graphs for Lexical Acquisition

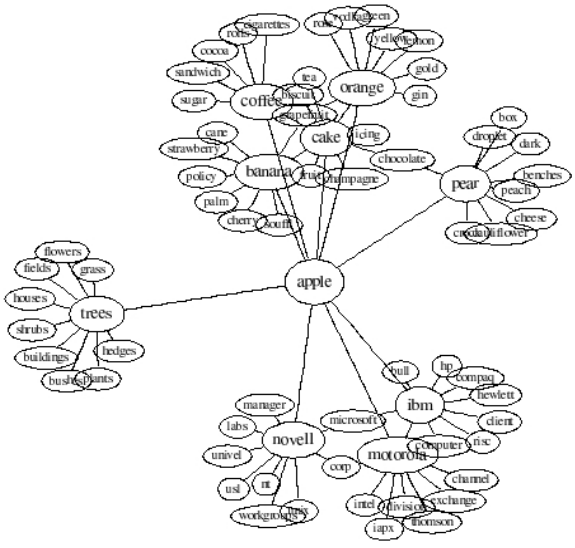
- Automatic acquisition of Semantic Classes
  - E.g. [FRUIT] apple, banana, orange, …
- Algorithm
  - Process corpus and extract all noun-noun pairs linked by an and/or relationship
    - E.g. “apple and banana”
  - Start with a seed, build graph centered on seed
    - E.g. “apple and banana”, “apple and orange”, “apple and pear”, “pear and banana”, “apple and strawberry”, “banana and strawberry”



  - Add the most connected node
  - Repeat

Examples and Evaluation

- Evaluation against 20 WordNet semantic classes
  - E.g. instruments, tools, diseases, …
- Precision measured at 82%
  - An order of magnitude better than previous approaches relying on traditional information extraction with bootstrapping



Outline

- word sense disambiguation
- entity disambiguation
- thesaurus construction / semantic classes
- textual entailment
- sentiment classification

Textual Entailment

- "Textual entailment recognition is the task of deciding, given two text fragments, whether the meaning of one text is entailed (can be inferred) from another text." *(Dagan et al., 2005)*

Eyeing the huge market potential, currently led by Google, Yahoo took over search company Overture Services Inc last year → **Yahoo acquired Overture**

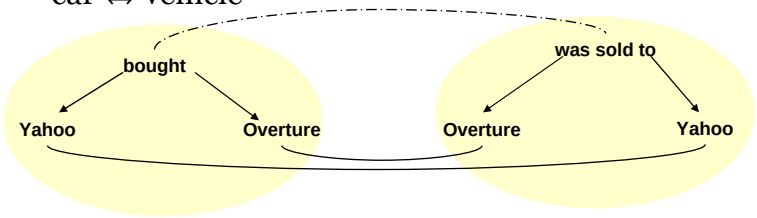
Applications to:	
Information Retrieval (IR)	Question Answering (QA)
Comparable Documents (CD)	Information Extraction (IE)
Reading Comprehension (RC)	Machine Translation (MT)
	Paraphrase Acquisition (PP)

Textual Entailment

- Knowledge required
  - Syntactic:
    - nominalization, verb syntactic frames, argument insertion/deletions
    - *[Yahoo bought Overture] => H: [Overture was bought by Yahoo]*
  - Semantic:
    - word meaning relations (synonymy, hypernymy, antinomy)
    - *[Yahoo buys] => H: [Yahoo owns]*
  - World knowledge
    - common sense facts
    - *[The train to Paris leaves at noon] => H: [The train to France leaves after 11:00]*
- RTE challenge:
  - 567 training, 800 test
  - baseline 50%

Graph Representations

- Text entailment as a graph matching problem
- Model the text as a graph, accounting for
  - syntactic relations
  - semantic relations (semantic roles)
- Seek a minimum cost match, allowing also for semantic similarities
  - car ⇔ vehicle



Textual Entailment

- Accuracy
  - overall: 52.4%
  - various tasks: 76.5% (comparable documents), 39.5% (question answering)
  - *(Pazienza et al. 2005)*
- Improved model
  - add negation, antonymy check, numeric mismatch
  - use logic-like formula representations
  - use matching scores for each pair of terms and weighted graph-matching
  - accuracy 56.8%
  - *(Haghighi et al. 2005)*

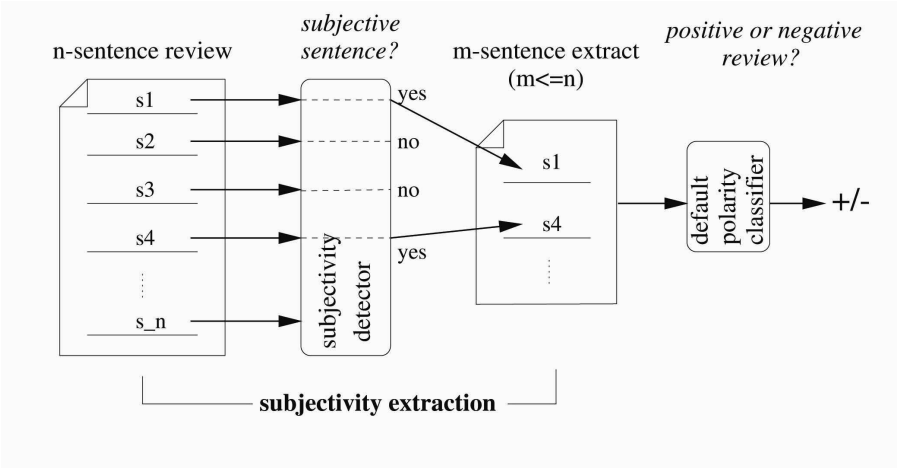
Outline

- word sense disambiguation
- entity disambiguation
- thesaurus construction / semantic classes
- textual entailment
- sentiment classification

Subjectivity Analysis for Sentiment Classification

- The objective is to detect subjective expressions in text (opinions against facts)
- Use this information to improve the polarity classification (positive vs. negative)
  - E.g. Movie reviews (see: www.rottentomatoes.com)
- Sentiment analysis can be considered as a document classification problem, with target classes focusing on the authors sentiments, rather than topic-based categories
  - Standard machine learning classification techniques can be applied

Subjectivity Extraction



Subjectivity Detection/Extraction

- Detecting the subjective sentences in a text may be useful in filtering out the objective sentences creating a subjective extract
- Subjective extracts facilitate the polarity analysis of the text (increased accuracy at reduced input size)
- Subjectivity detection can use local and contextual features:
  - Local: relies on individual sentence classifications using standard machine learning techniques (SVM, Naïve Bayes, etc) trained on an annotated data set
  - Contextual: uses context information, such as e.g. sentences occurring near each other tend to share the same subjectivity status (coherence)
- *(Pang and Lee, 2004)*

Cut-based Subjectivity Classification

- Standard classification techniques usually consider only individual features (classify one sentence at a time).
- Cut-based classification takes into account both individual and contextual (structural) features
- Suppose we have n items $x_1,\ldots,x_n$ to divide in two classes: C_1 and C_2 .
- Individual scores: $ind_j(x_i)$ - non-negative estimates of each x_i being in C_j based on the features of x_i alone
- Association scores: $assoc(x_i,x_k)$ - non-negative estimates of how important it is that x_i and x_k be in the same class

Cut-based Classification

- Maximize each item's assignment score (individual score for the class it is assigned to, minus its individual score for the other class), while penalize the assignment of different classes to highly associated items
- Formulated as an optimization problem: assign the x_i items to classes C_1 and C_2 so as to minimize the partition cost:

$$\sum_{x \in C_1} ind_2(x) + \sum_{x \in C_2} ind_1(x) + \sum_{\substack{x_i \in C_1 \\ x_k \in C_2}} assoc(x_i, x_k)$$

Cut-based Algorithm

- There are 2^n possible binary partitions of the n elements, we need an efficient algorithm to solve the optimization problem
- Build an undirected graph G with vertices $\{v_1,\ldots v_n,s,t\}$ and edges:
  - (s,v_i) with weights $ind_1(x_i)$
  - (v_i,t) with weights $ind_2(x_i)$
  - (v_i,v_k) with weights $assoc(x_i,x_k)$

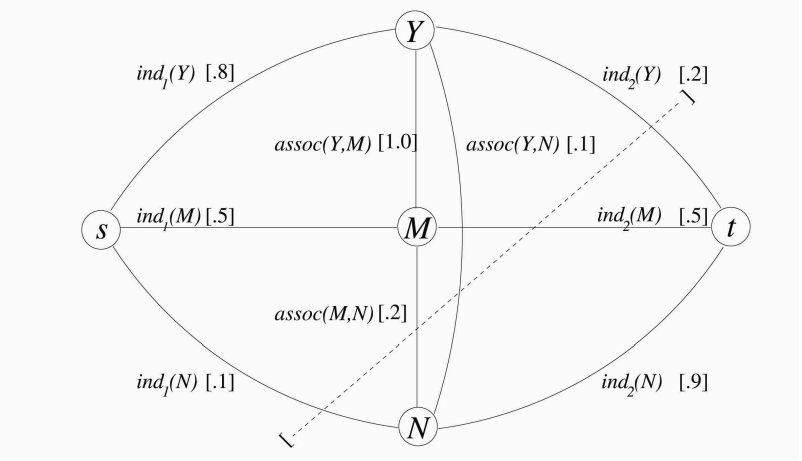
Cut-based Algorithm (cont.)

- Cut: a partition of the vertices in two sets:

$$S = \{s\} \cup S' \text{ and } T = \{t\} \cup T'$$
$$\text{where } s \notin S', t \notin T'$$

- The cost is the sum of the weights of all edges crossing from S to T
- A minimum cut is a cut with the minimal cost
- A minimum cut can be found using maximum-flow algorithms, with polynomial asymptotic running times
- Use the min-cut / max-flow algorithm

Cut-based Algorithm (cont.)

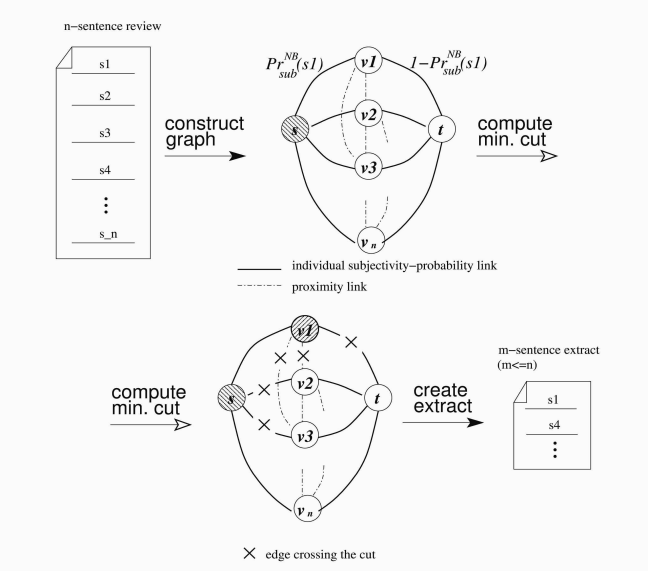


Notice that without the structural information we would be undecided about the assignment of node M

Subjectivity Extraction

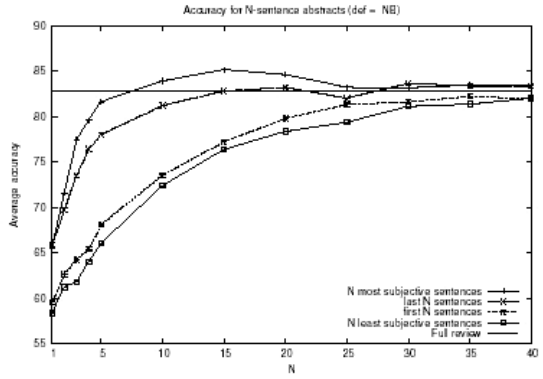
- Assign every individual sentence a subjectivity score
  - e.g. the probability of a sentence being subjective, as assigned by a Naïve Bayes classifier, etc
- Assign every sentence pair a proximity or similarity score
  - e.g. physical proximity = the inverse of the number of sentences between the two entities
- Use the min-cut algorithm to classify the sentences into objective/subjective

Subjectivity Extraction with Min-Cut



Results

- 2000 movie reviews (1000 positive / 1000 negative)



- The use of subjective extracts improves or maintains the accuracy of the polarity analysis while reducing the input data size

References

- Dolan, William B., L. Vanderwende, and S. Richardson. 1993. Automatically Deriving Structured Knowledge Base from On-line Dictionaries. In Proceedings of the Pacific Association for Computational Linguistics, April 21-24, 1993, Vancouver, British Columbia.
- Erkan, G. and Radev, D.. "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization", Journal of Artificial Intelligence Research , 2004
- Galley, M. and McKeown, K.: Improving Word Sense Disambiguation in Lexical Chaining. IJCAI 2003.
- Hirst, G. and St-Onge, D. Lexical chains as representations of context in the detection and correction of malaproprisms. WordNet: An electronic lexical database, MIT Press, 1998
- Haghighi, A. and A. Ng and C. Manning, Robust Textual Inference via Graph Matching, EMNLP 2005.
- Halliday, M. and Hasan, R. Cohesion in English. Longman, 1976.
- Hobbs, J. A model for natural language semantics, Tech. Report, Yale University, 1974.
- Jannink, J. and G. Wiederhold. Thesaurus entry extraction from an on-line dictionary. Fusion 1999.
- Lee, L. Measures of distributional similarity, ACL 1999.

References

- Lesk, M. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. SIGDOC 1986.
- Lin, D. An Information-Theoretic Definition of Similarity, ICML 1998.
- Mihalcea, R. and Moldovan, D. An iterative approach to word sense disambiguation. FLAIRS 2000.
- Mihalcea, R. and Tarau, P and Figa, E. PageRank on Semantic Networks with Application to Word Sense Disambiguation, COLING 2004
- Mihalcea, R. and Tarau, P. TextRank: Bringing Order Into Texts, EMNLP 2004
- Mihalcea, R. Graph-based ranking algorithms for large vocabulary word sense disambiguation, HTL-EMNLP 2005.
- Han, H and Zha, H. and Giles, C.L. Name disambiguation in author citations using a K-way spectral clustering method, ACM/IEEE – IJCDL 2005
- Jannink, J. and Wiederhold, G., Thesaurus Entry Extraction from an On-line Dictionary, Fusion 1999, Sunnyvale CA, 1999.
- Pang, B. and Lee, L., A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts, *ACL* 2004.
- Widdows, D. and Dorow, B., A Graph Model for Unsupervised Lexical Acquisition, COLING 2002

Part V

Natural Language Processing Applications
Summarization and Syntax

Outline

- Text summarization
- Passage retrieval
- Prepositional phrase attachment
- Dependency parsing
- Keyword extraction

Centrality in summarization

- Extractive summarization (pick k sentences that are most representative of a collection of n sentences).
- Motivation: capture the most central words in a document or cluster.
- Typically done by picking sentences that contain certain words (e.g., overlap with the title of the document) or by position.
- The centroid method [Radev et al. 2000].
- Alternative methods for computing centrality?

Sample multidocument cluster (DUC cluster d1003t)

1 (d1s1) Iraqi Vice President Taha Yassin Ramadan announced today, Sunday, that Iraq refuses to back down from its decision to stop cooperating with disarmament inspectors before its demands are met.

2 (d2s1) Iraqi Vice president Taha Yassin Ramadan announced today, Thursday, that Iraq rejects cooperating with the United Nations except on the issue of lifting the blockade imposed upon it since the year 1990.

3 (d2s2) Ramadan told reporters in Baghdad that "Iraq cannot deal positively with whoever represents the Security Council unless there was a clear stance on the issue of lifting the blockade off of it.

4 (d2s3) Baghdad had decided late last October to completely cease cooperating with the inspectors of the United Nations Special Commission (UNSCOM), in charge of disarming Iraq's weapons, and whose work became very limited since the fifth of August, and announced it will not resume its cooperation with the Commission even if it were subjected to a military operation.

5 (d3s1) The Russian Foreign Minister, Igor Ivanov, warned today, Wednesday against using force against Iraq, which will destroy, according to him, seven years of difficult diplomatic work and will complicate the regional situation in the area.

6 (d3s2) Ivanov contended that carrying out air strikes against Iraq, who refuses to cooperate with the United Nations inspectors, ``will end the tremendous work achieved by the international group during the past seven years and will complicate the situation in the region."

7 (d3s3) Nevertheless, Ivanov stressed that Baghdad must resume working with the Special Commission in charge of disarming the Iraqi weapons of mass destruction (UNSCOM).

8 (d4s1) The Special Representative of the United Nations Secretary-General in Baghdad, Prakash Shah, announced today, Wednesday, after meeting with the Iraqi Deputy Prime Minister Tariq Aziz, that Iraq refuses to back down from its decision to cut off cooperation with the disarmament inspectors.

9 (d5s1) British Prime Minister Tony Blair said today, Sunday, that the crisis between the international community and Iraq ``did not end" and that Britain is still ``ready, prepared, and able to strike Iraq."

10 (d5s2) In a gathering with the press held at the Prime Minister's office, Blair contended that the crisis with Iraq ``will not end until Iraq has absolutely and unconditionally respected its commitments" towards the United Nations.

11 (d5s3) A spokesman for Tony Blair had indicated that the British Prime Minister gave permission to British Air Force Tornado planes stationed in Kuwait to join the aerial bombardment against Iraq.

Cosine between sentences

- Let s_1 and s_2 be two sentences.
- Let x and y be their representations in an n-dimensional vector space
- The cosine between is then computed based on the inner product of the two.

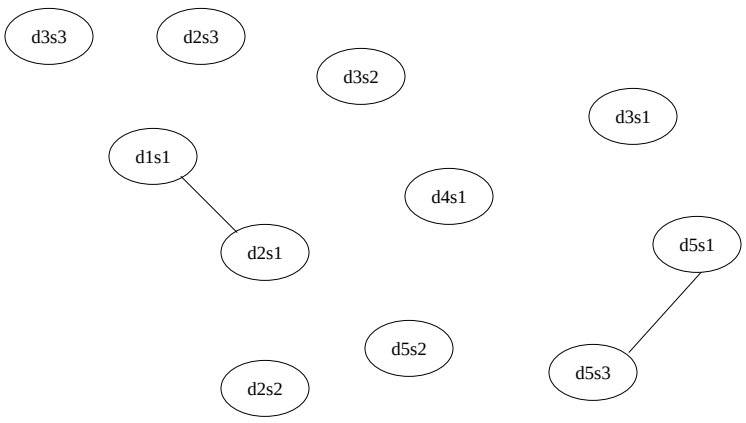
$$\cos(x, y) = \frac{\sum_{i=1,n} x_i y_i}{\|x\| \|y\|}$$

- The cosine ranges from 0 to 1.

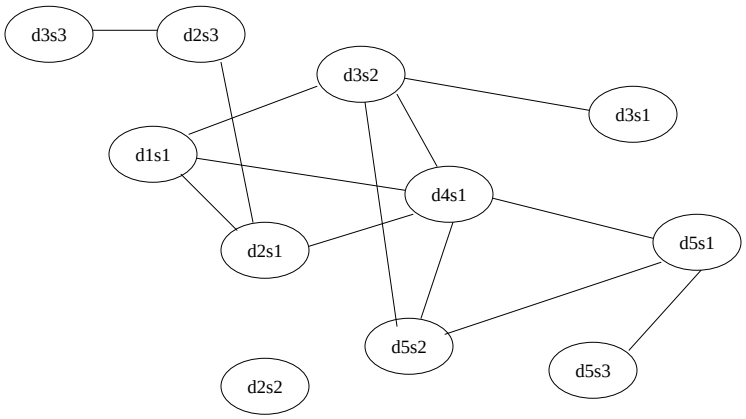
Lexical centrality [Erkan and Radev 2004]

	1	2	3	4	5	6	7	8	9	10	11
1	1.00	0.45	0.02	0.17	0.03	0.22	0.03	0.28	0.06	0.06	0.00
2	0.45	1.00	0.16	0.27	0.03	0.19	0.03	0.21	0.03	0.15	0.00
3	0.02	0.16	1.00	0.03	0.00	0.01	0.03	0.04	0.00	0.01	0.00
4	0.17	0.27	0.03	1.00	0.01	0.16	0.28	0.17	0.00	0.09	0.01
5	0.03	0.03	0.00	0.01	1.00	0.29	0.05	0.15	0.20	0.04	0.18
6	0.22	0.19	0.01	0.16	0.29	1.00	0.05	0.29	0.04	0.20	0.03
7	0.03	0.03	0.03	0.28	0.05	0.05	1.00	0.06	0.00	0.00	0.01
8	0.28	0.21	0.04	0.17	0.15	0.29	0.06	1.00	0.25	0.20	0.17
9	0.06	0.03	0.00	0.00	0.20	0.04	0.00	0.25	1.00	0.26	0.38
10	0.06	0.15	0.01	0.09	0.04	0.20	0.00	0.20	0.26	1.00	0.12
11	0.00	0.00	0.00	0.01	0.18	0.03	0.01	0.17	0.38	0.12	1.00

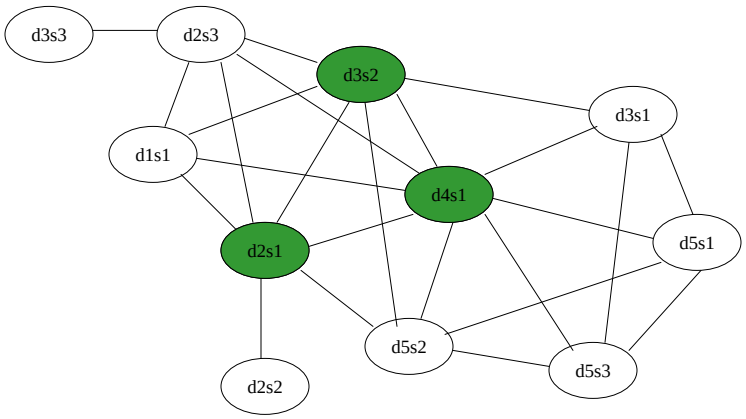
Lexical centrality (t=0.3)



Lexical centrality (t=0.2)



Lexical centrality (t=0.1)

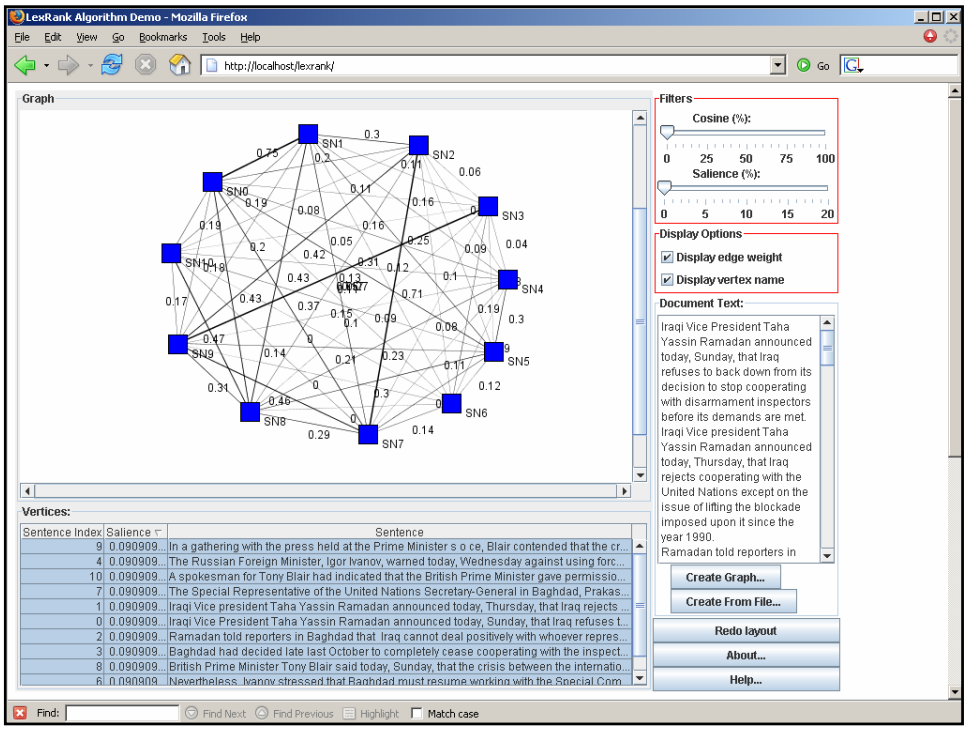


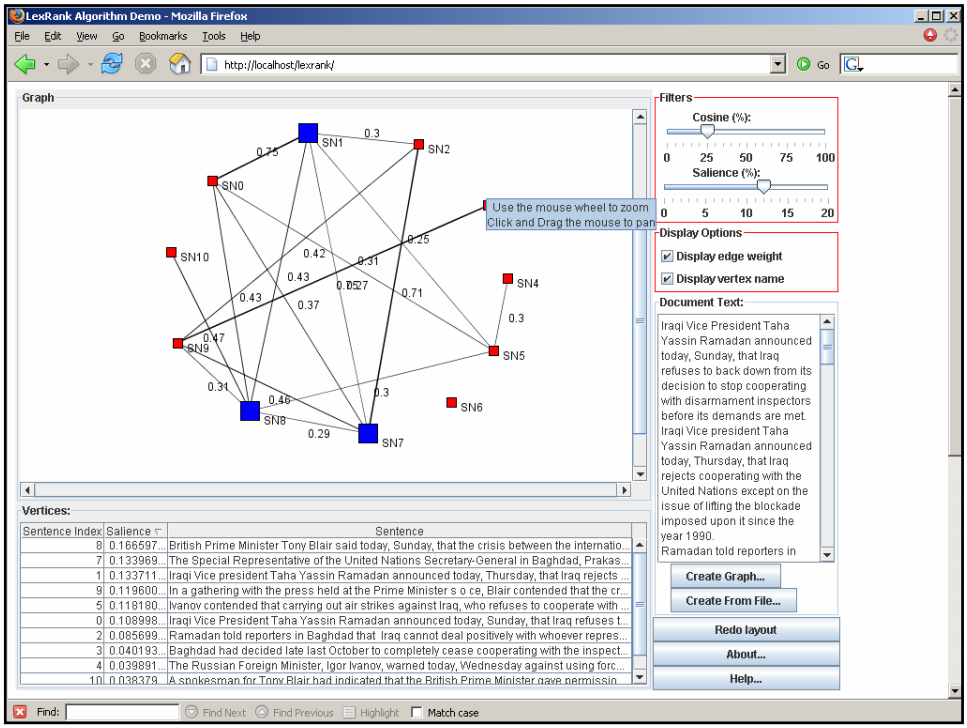
Sentences vote for the most central sentence!
Need to worry about diversity reranking.

LexRank

$$p(Ti) = \frac{d}{c(T_1)} p(T_1)E(T_1,Ti) + \ldots + \frac{d}{c(T_n)} p(T_n)E(T_n,Ti) + \frac{1-d}{N}$$

- $T_1 \ldots T_n$ are pages that link to A, $c(T_i)$ is the outdegree of page T_i, *and* N is the total number of pages.
- d is the "damping factor", or the probability that we "jump" to a far-away node during the random walk. It accounts for disconnected components or periodic graphs.
- When d = 0, we have a strict uniform distribution. When d = 1, the method is not guaranteed to converge to a unique solution.
- Typical value for d is between [0.1,0.2] (Brin and Page, 1998).



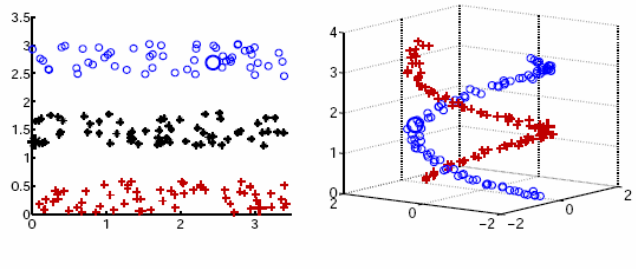


Extensions to LexRank

- For document ranking (Kurland and Lee 2005): replace cosine with the asymmetric generation probability $p(D_j|D_i)$. Also (Kurland and Lee 2006): a variant of HITS without hyperlinks.
- Document clustering using random walks (Erkan 2006): look at distance 1-3 neighbors of each document.

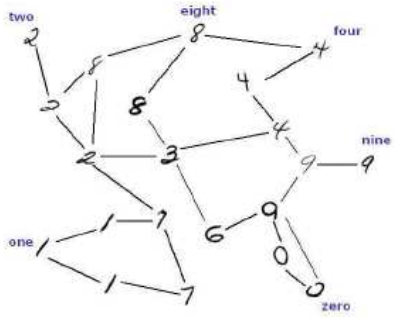
Learning on graphs

- Example:



Example from
Zhu et al. 2003

Learning on graphs



Example from
Zhu et al. 2003

- Search for a lower dimensional manifold
- Relaxation method
- Monte Carlo method
- Supervised vs. semi-supervised

PP attachment

Pierre Vinken , 61 years old , will join the board as a nonexecutive director Nov. 29. Mr. Vinken is chairman of Elsevier N.V. , the Dutch publishing group. Rudolph Agnew , 55 years old and former chairman of Consolidated Gold Fields PLC , was named a nonexecutive director of this British industrial conglomerate.

A form of asbestos once used to make Kent cigarette filters has caused a high percentage of cancer deaths among a group of workers exposed to it more than 30 years ago , researchers reported . The asbestos fiber , crocidolite , is unusually resilient once it enters the lungs , with even brief exposures to it causing symptoms that show up decades later , researchers said . Lorillard Inc. , the unit of New York-based Loews Corp. that makes Kent cigarettes , stopped using crocidolite in its Micronite cigarette filters in 1956 . Although preliminary findings were reported more than a year ago , the latest results appear in today 's New England Journal of Medicine , a forum likely to bring new attention to the problem .

- High vs. low attachment

V	**x02_join**	x01_board	x0_as	x11_director
N	x02_is	**x01_chairman**	x0_of	x11_entitynam
N	x02_name	**x01_director**	x0_of	x11_conglomer
N	x02_caus	**x01_percentag**	x0_of	x11_death
V	**x02_us**	x01_crocidolit	x0_in	x11_filter
V	**x02_bring**	x01_attent	x0_to	x11_problem

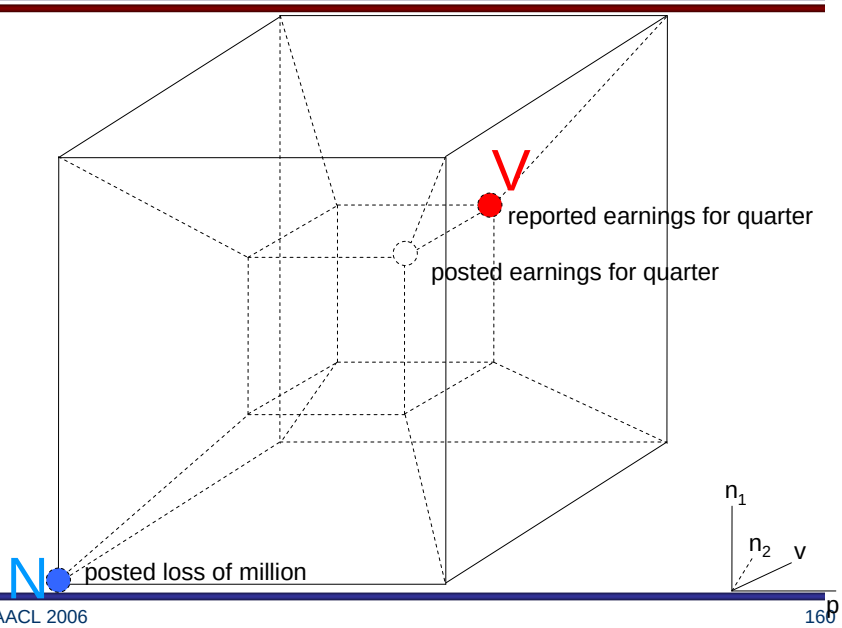
PP attachment

- The first work using graph methods for PP attachment was done by Toutanova et al. 2004.
- Example: training data: “hang with nails” – expand to “fasten with nail”.
- Separate transition matrices for each preposition.
- Link types: V→N, V→V (verbs with similar dependents), Morphology, WordnetSynsets, N→V (words with similar heads), External corpus (BLLIP).
- Excellent performance: 87.54% accuracy (compared to 86.5% by Zhao and Lin 2004).

Example

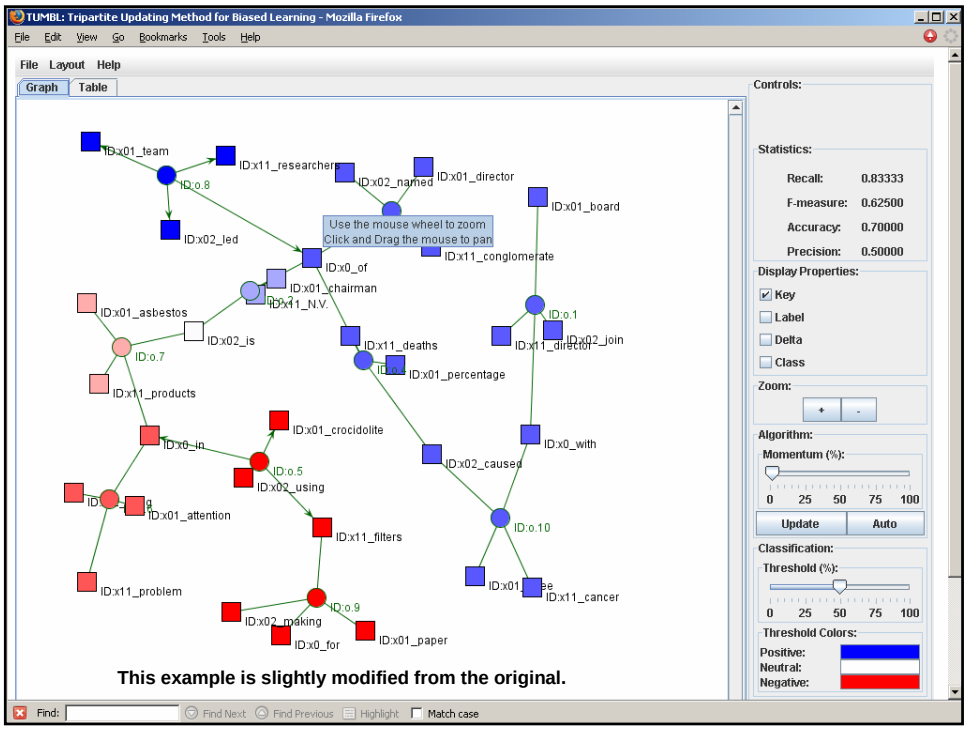
	reported	earnings	for	quarter	V
	reported	loss	for	quarter	?
	posted	loss	for	quarter	?
*	posted	loss	of	quarter	?
	posted	loss	of	million	N

Hypercube (**)

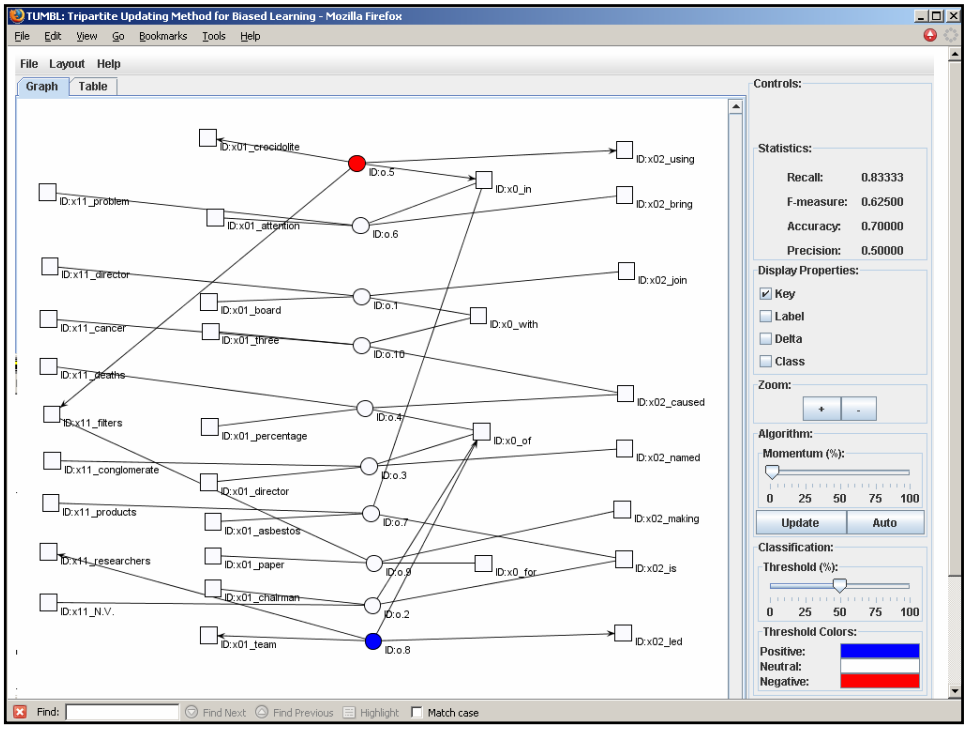


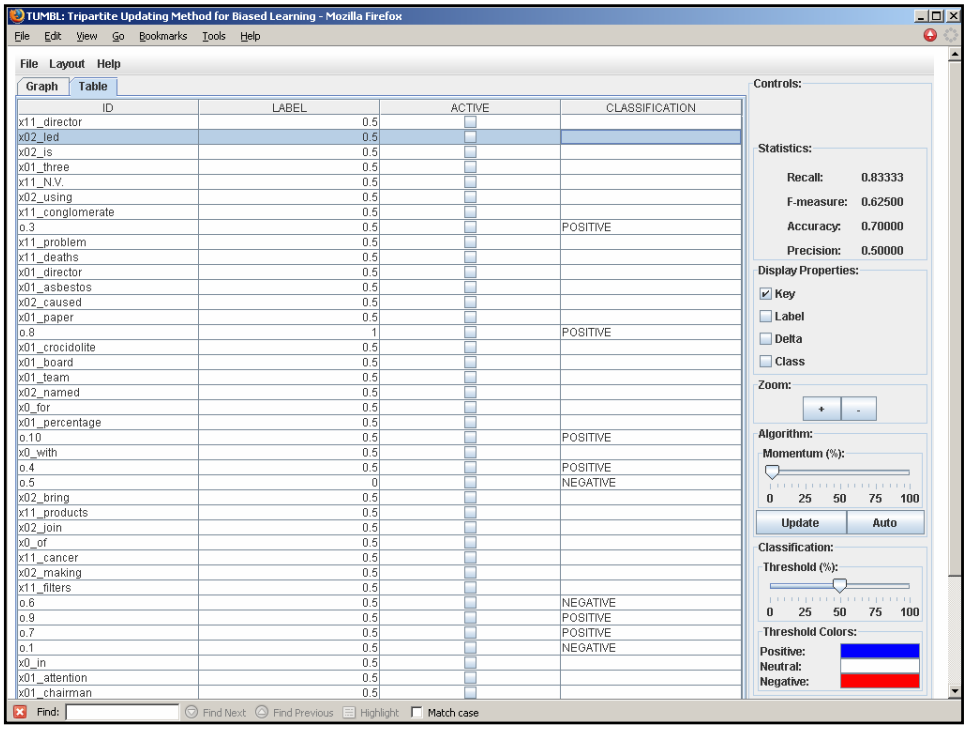
TUMBL (**)

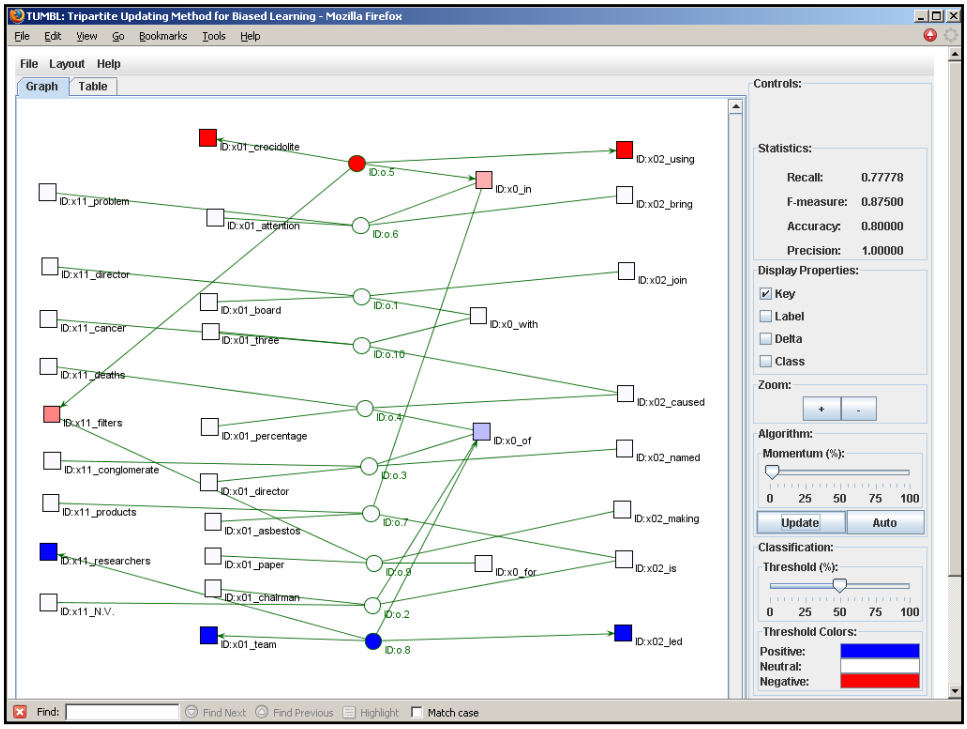
TUMBL(n, d_0)
// n: number of iterations; d_0: damping factor
Initialize class distribution matrices X, Y_0, and Z_0.
$d = 1$
for $t = 1 : n$
$Y_t = d \times T_1^T X + Y_{t-1}$
Row-normalize Y_t
$Z_t = d \times T_2 Y_t + Z_{t-1}$
Row-normalize Z_t
$Y_t = d \times T_2^T Z_t + Y_t$
Row-normalize Y_t
$d = d * d_0$
endfor
for $i = 1 : ||U||$
if $Z_n(i, 1) > Z_n(i, 2)$ **then**
Assign positive to instance
elseif $Z_n(i, 1) < Z_n(i, 2)$ **then**
Assign negative to instance
else
Assign more probable class to instance
endif
endfor

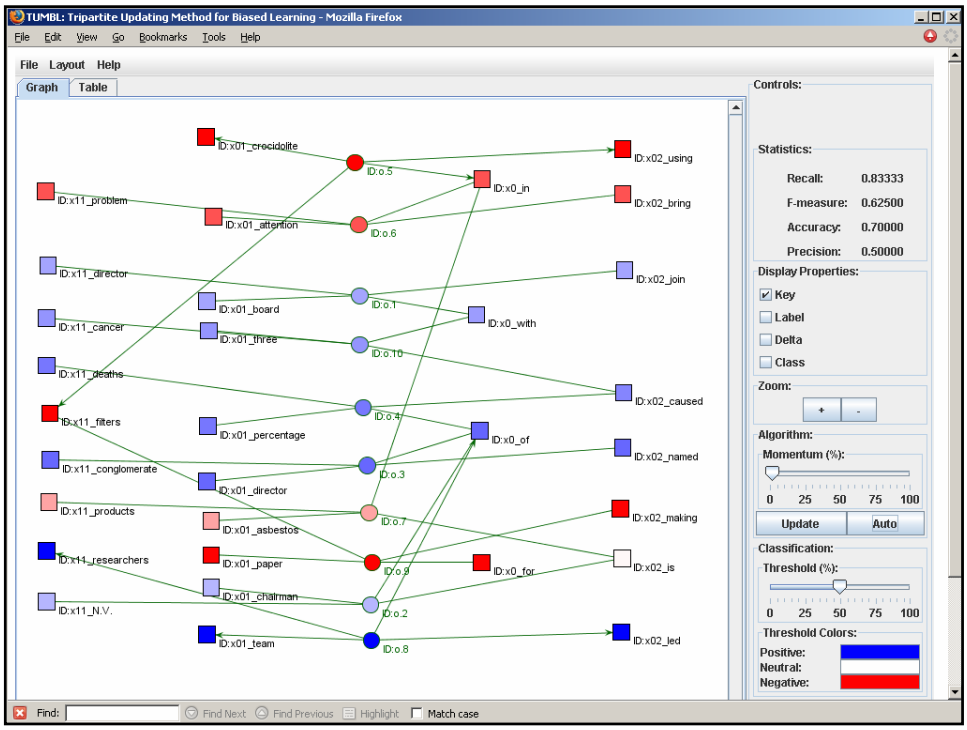


This example is slightly modified from the original.


TUMBL: Tripartite Updating Method for Biased Learning - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
File Layout Help
Graph Table
ID:x01_crocidolite
ID:o.5
ID:x02_using
ID:x0_in
ID:x11_problem
ID:x01_attention
ID:o.6
ID:x02_bring
ID:x11_director
ID:x01_board
ID:o.1
ID:x02_join
ID:x11_cancer
ID:x01_three
ID:o.10
ID:x0_with
ID:x11_deaths
ID:x02_caused
ID:x11_filters
ID:o.4
ID:x01_percentage
ID:x0_of
ID:x11_conglomerate
ID:x01_director
ID:o.3
ID:x02_named
ID:x11_products
ID:x01_asbestos
ID:o.7
ID:x02_making
ID:x11_researchers
ID:x01_paper
ID:o.9
ID:x0_for
ID:x02_is
ID:x01_chairman
ID:x11_N.V.
ID:o.2
ID:x01_team
ID:o.8
ID:x02_led
Controls:
Statistics:
Recall: 0.83333
F-measure: 0.62500
Accuracy: 0.70000
Precision: 0.50000
Display Properties:
Key
Label
Delta
Class
Zoom:
+ -
Algorithm:
Momentum (%):
0 25 50 75 100
Update Auto
Classification:
Threshold (%):
0 25 50 75 100
Threshold Colors:
Positive:
Neutral:
Negative:
Find: Find Next Find Previous Highlight Match case

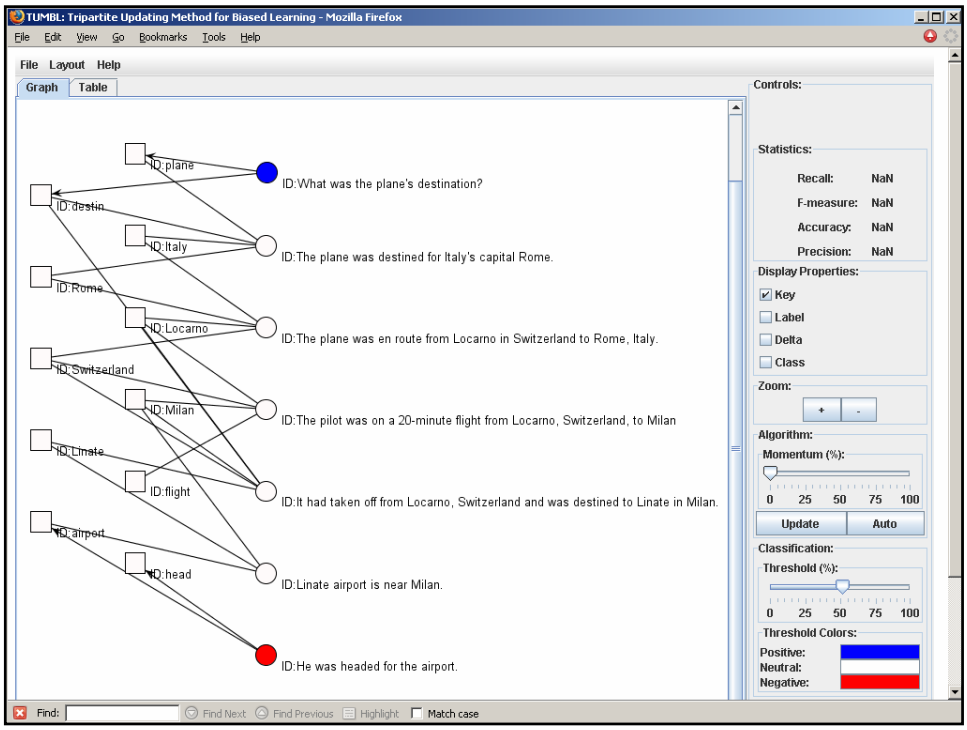

TUMBL: Tripartite Updating Method for Biased Learning - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
File Layout Help
Graph Table
ID LABEL ACTIVE CLASSIFICATION
x11_director 0.5
x02_led 0.5
x02_is 0.5
x01_three 0.5
x11_N.V. 0.5
x02_using 0.5
x11_conglomerate 0.5
o.3 0.5 POSITIVE
x11_problem 0.5
x11_deaths 0.5
x01_director 0.5
x01_asbestos 0.5
x02_caused 0.5
x01_paper 0.5
o.8 1 POSITIVE
x01_crocidolite 0.5
x01_board 0.5
x01_team 0.5
x02_named 0.5
x0_for 0.5
x01_percentage 0.5
o.10 0.5 POSITIVE
x0_with 0.5
o.4 0.5 POSITIVE
o.5 0 NEGATIVE
x02_bring 0.5
x11_products 0.5
x02_join 0.5
x0_of 0.5
x11_cancer 0.5
x02_making 0.5
x11_filters 0.5
o.6 0.5 NEGATIVE
o.9 0.5 POSITIVE
o.7 0.5 POSITIVE
o.1 0.5 NEGATIVE
x0_in 0.5
x01_attention 0.5
x01_chairman 0.5
Controls:
Statistics:
Recall: 0.83333
F-measure: 0.62500
Accuracy: 0.70000
Precision: 0.50000
Display Properties:
Key
Label
Delta
Class
Zoom:
+ -
Algorithm:
Momentum (%):
0 25 50 75 100
Update Auto
Classification:
Threshold (%):
0 25 50 75 100
Threshold Colors:
Positive:
Neutral:
Negative:
Find: Find Next Find Previous Highlight Match case

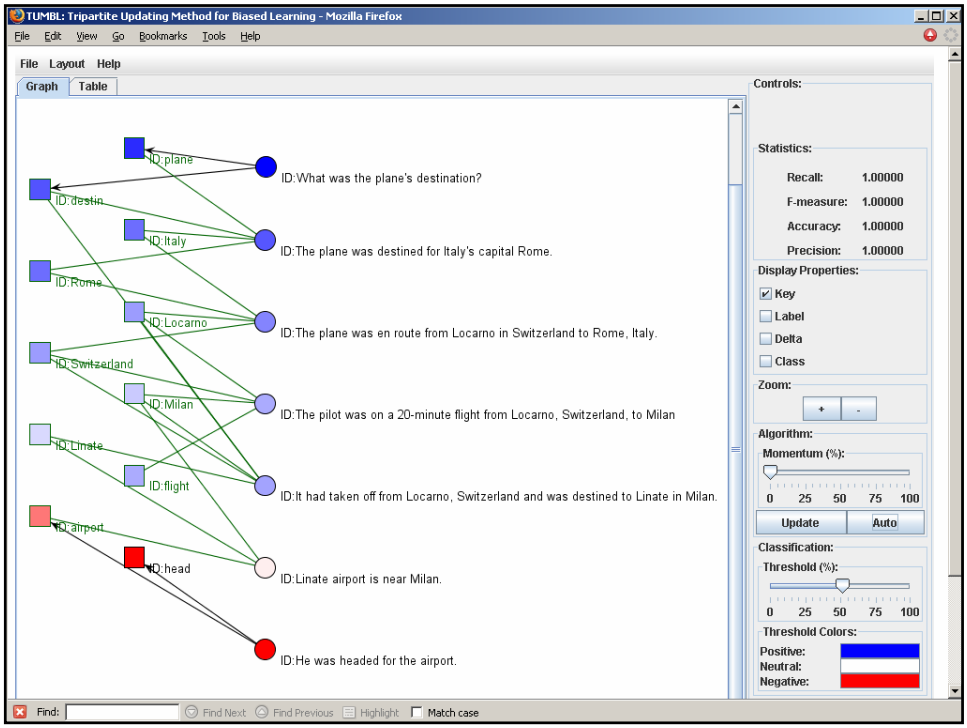

TUMBL: Tripartite Updating Method for Biased Learning - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
File Layout Help
Graph Table
ID:x01_crocidolite
ID:x11_problem
ID:x01_attention
ID:x11_director
ID:x01_board
ID:x11_cancer
ID:x01_three
ID:x11_deaths
ID:x11_filters
ID:x01_percentage
ID:x11_conglomerate
ID:x01_director
ID:x11_products
ID:x01_asbestos
ID:x11_researchers
ID:x01_paper
ID:x01_chairman
ID:x11_N.V.
ID:x01_team
ID:o.5
ID:o.6
ID:o.1
ID:o.10
ID:o.4
ID:o.3
ID:o.7
ID:o.9
ID:o.2
ID:o.8
ID:x0_in
ID:x0_with
ID:x0_of
ID:x0_for
ID:x02_using
ID:x02_bring
ID:x02_join
ID:x02_caused
ID:x02_named
ID:x02_making
ID:x02_is
ID:x02_led
Controls:
Statistics:
Recall: 0.77778
F-measure: 0.87500
Accuracy: 0.80000
Precision: 1.00000
Display Properties:
Key
Label
Delta
Class
Zoom:
+ -
Algorithm:
Momentum (%):
0 25 50 75 100
Update Auto
Classification:
Threshold (%):
0 25 50 75 100
Threshold Colors:
Positive:
Neutral:
Negative:
Find: Find Next Find Previous Highlight Match case


TUMBL: Tripartite Updating Method for Biased Learning - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
File Layout Help
Graph Table
ID:x01_crocidolite
ID:x11_problem
ID:x01_attention
ID:x11_director
ID:x01_board
ID:x11_cancer
ID:x01_three
ID:x11_deaths
ID:x11_filters
ID:x01_percentage
ID:x11_conglomerate
ID:x01_director
ID:x11_products
ID:x01_asbestos
ID:x11_researchers
ID:x01_paper
ID:x01_chairman
ID:x11_N.V.
ID:x01_team
ID:o.5
ID:o.6
ID:o.1
ID:o.10
ID:o.4
ID:o.3
ID:o.7
ID:o.9
ID:o.2
ID:o.8
ID:x0_in
ID:x0_with
ID:x0_of
ID:x0_for
ID:x02_using
ID:x02_bring
ID:x02_join
ID:x02_caused
ID:x02_named
ID:x02_making
ID:x02_is
ID:x02_led
Controls:
Statistics:
Recall: 0.83333
F-measure: 0.62500
Accuracy: 0.70000
Precision: 0.50000
Display Properties:
Key
Label
Delta
Class
Zoom:
+ -
Algorithm:
Momentum (%):
0 25 50 75 100
Update Auto
Classification:
Threshold (%):
0 25 50 75 100
Threshold Colors:
Positive:
Neutral:
Negative:
Find: Find Next Find Previous Highlight Match case

Semi-supervised passage retrieval

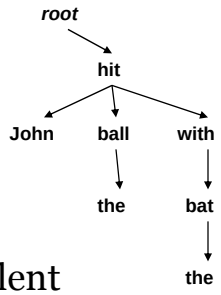
- Otterbacher et al. 2005.
- Graph-based semi-supervised learning.
- The idea is to propagate information from labeled nodes to unlabeled nodes using the graph connectivity.
- A passage can be either positive (labeled as relevant) or negative (labeled as not relevant), or unlabeled.





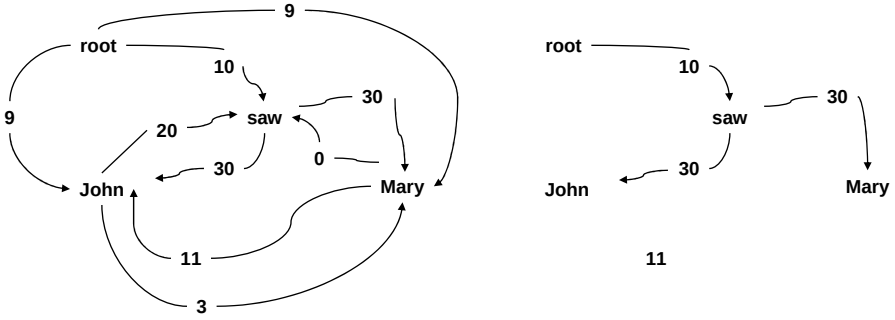
Dependency parsing

- McDonald et al. 2005.
- Example of a dependency tree:
- English dependency trees are mostly projective (can be drawn without crossing dependencies). Other languages are not.
- Idea: dependency parsing is equivalent to search for a maximum spanning tree in a directed graph.
- Chu and Liu (1965) and Edmonds (1967) give an efficient algorithm for finding MST for directed graphs.



Dependency parsing

- Consider the sentence “John saw Mary” (left).
- The Chu-Liu-Edmonds algorithm gives the MST on the right hand side (right). This is in general a non-projective tree.



Keyword Extraction

- Identify important words in a text
  [Mihalcea & Tarau 2004]
- Keywords useful for
  - Automatic indexing
  - Terminology extraction
  - Within other applications: Information Retrieval, Text Summarization, Word Sense Disambiguation
- Previous work
  - mostly supervised learning
  - genetic algorithms [Turney 1999], Naïve Bayes [Frank 1999], rule induction [Hulth 2003]

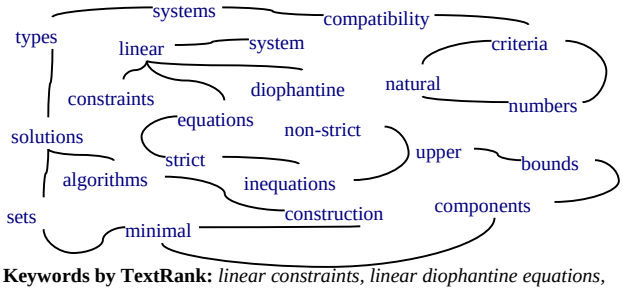
TextRank for Keyword Extraction

- Store words in vertices
- Use co-occurrence to draw edges
- Rank graph vertices across the entire text
- Pick top N as keywords

- Variations:
  - rank all open class words
  - rank only nouns
  - rank only nouns + adjectives

An Example

Compatibility of systems of linear constraints over the set of natural numbers Criteria of compatibility of a system of linear Diophantine equations, strict inequations, and nonstrict inequations are considered. Upper bounds for components of a minimal set of solutions and algorithms of construction of minimal generating sets of solutions for all types of systems are given. These criteria and the corresponding algorithms for constructing a minimal supporting set of solutions can be used in solving all the considered types of systems and systems of mixed types.



Keywords by TextRank: *linear constraints, linear diophantine equations, natural numbers, non-strict inequations, strict inequations, upper bounds*
Keywords by human annotators: *linear constraints, linear diophantine equations, non-strict inequations, set of natural numbers, strict inequations, upper bounds*

TextRank
numbers (1.46)
inequations (1.45)
linear (1.29)
diophantine (1.28)
upper (0.99)
bounds (0.99)
strict (0.77)

Frequency
systems (4)
types (4)
solutions (3)
minimal (3)
linear (2)
inequations (2)
algorithms (2)

Evaluation

- Evaluation:
  - 500 INSPEC abstracts
  - collection previously used in keyphrase extraction [Hulth 2003]
- Various settings. Here:
  - nouns and adjectives
  - select top N/3
- Evaluation in previous work
  - mostly supervised learning
  - training/development/test : 1000/500/500 abstracts

Results

Method	Assigned		Correct				
	Total	Mean	Total	Mean	Precision	Recall	F-measure
Ngram with tag	7,815	15.6	1,973	3.9	25.2	**51.7**	33.9
NP-chunks with tag	4,788	9.6	1,421	2.8	29.7	37.2	33
Pattern with tag	7,012	14.0	1,523	3.1	21.7	39.9	28.1
TextRank	6,784	13.7	2,116	4.2	**31.2**	43.1	**36.2**

(Hulth, 2003)

Literature

Blum and Chawla 2001 – Learning from Labeled and Unlabeled Data using Graph Mincuts, ICML

Dhillon 2001 – Co-clustering documents and words using Bipartite Spectral Graph Partitioning, SIGKDD

Doyle and Snell – random walks and electric networks

Erkan and Radev 2004 – LexPageRank: Prestige in Multi-Document Text Summarization", EMNLP

Erkan 2006 – Language Model-Based Document Clustering Using Random Walks, HLT-NAACL

Joachims 2003 – Transductive Learning via Spectral Graph Partitioning, ICML

Kamvar, Klein, and Manning 2003 – Spectral Learning, IJCAI

Kurland and Lee 2005 – PageRank without Hyperlinks: Structural Re-ranking using Links Induced by Language Models, SIGIR

Kurland and Lee 2006 – Respect my authority! HITS without hyperlinks, utilizing cluster-based language models, SIGIR

Mihalcea and Tarau 2004 – TextRank: Bringing Order into Texts, EMNLP

Literature

Senellart and Blondel 2003 – Automatic Discovery of Similar Words

Szummer and Jaakkola 2001 – Partially Labeled Classification with Markov Random Walks, NIPS

Zha et al. 2001 – Bipartite Graph Partitioning and Data Clustering, CIKM

Zha 2002 – Generic Summarization and Keyphrase Extraction Using Mutual Reinforcement Principle and Sentence Clustering, SIGIR

Zhu, Ghahramani, and Lafferty 2003 – Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions, ICML

Wu and Huberman 2004. Finding communities in linear time: a physics approach. The European Physics Journal B, 38:331—338

Large bibliography: http://tangra.si.umich.edu/~radev/webgraph/bibliography.pdf

Conclusions

Graph-based Algorithms for NLP & IR

- Graphs: encode information about nodes **and** relations between nodes
- Graph-based algorithms can lead to state-of-the-art results in several IR and NLP problems
- Graph-based NLP/IR:
  - On the verge between graph-theory and NLP / IR
  - Many opportunities for novel approaches to traditional problems