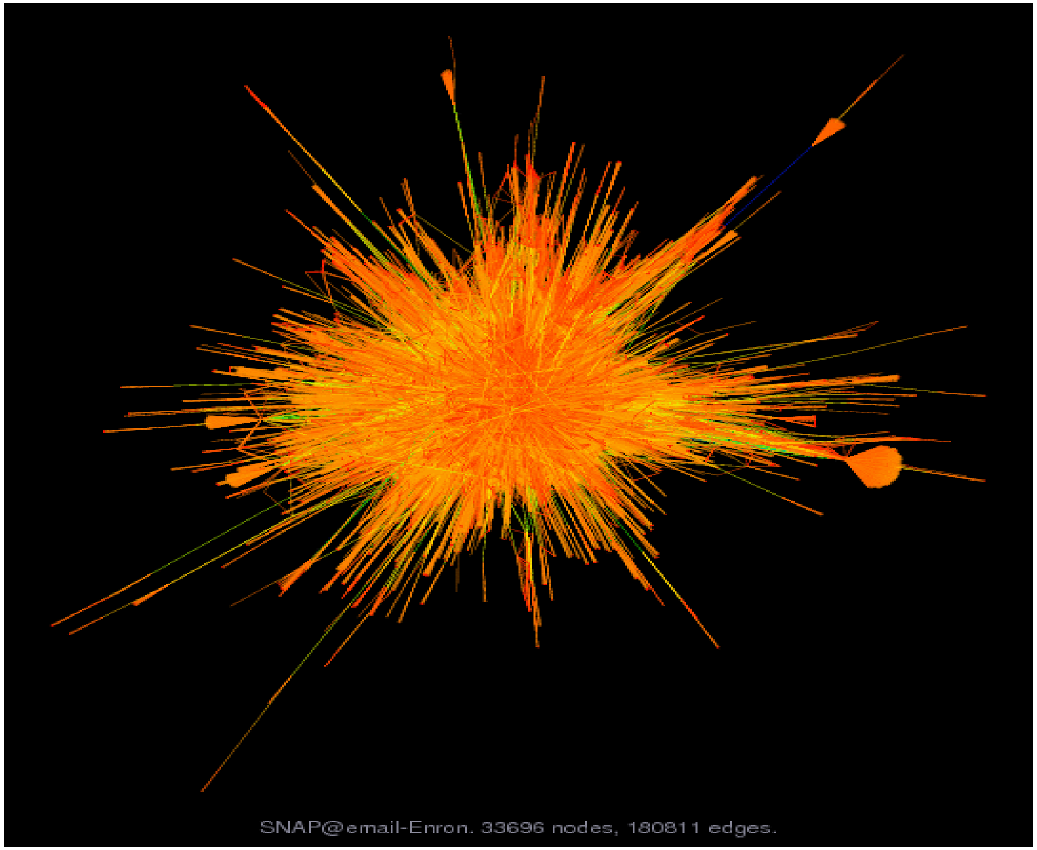
Introduction to Graph Mining

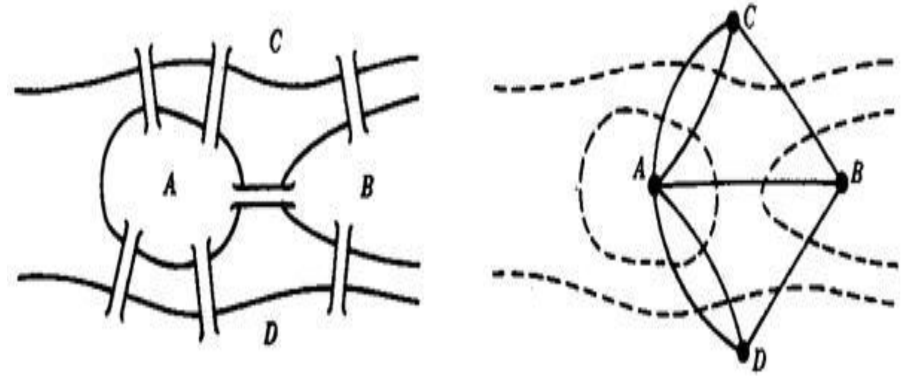


What is a graph?

- A graph $G = (V,E)$ is a set of vertices V and a set (possibly empty) E of pairs of vertices $e_1 = (v_1, v_2)$, where $e_1 \in E$ and $v_1, v_2 \in V$.

- Edges may
  - contain weights
  - contain labels
  - have direction

- Nodes may
  - contain additional labels



(a) Königsberg in 1736 (b) Euler's graphical representation

- Formal study of graphs is called graph theory.

Motivation

- Many domains produce data that can be intuitively represented by graphs.
- We would like to discover interesting facts and information about these graphs.
- Real world graph datasets too large for humans to make any sense of.
- We would like to discover patterns that have structural information present.

Application Domains

- Web structure graphs
- Social networks
- Protein interaction networks
- Chemical compound
- Program Flow
- Transportation networks

Graph Mining Problems

- Graph Theoretic Problems
  - Radius
  - Diameter

- Graph Statistics:
  - Degree Distribution
  - Clustering Co-efficient

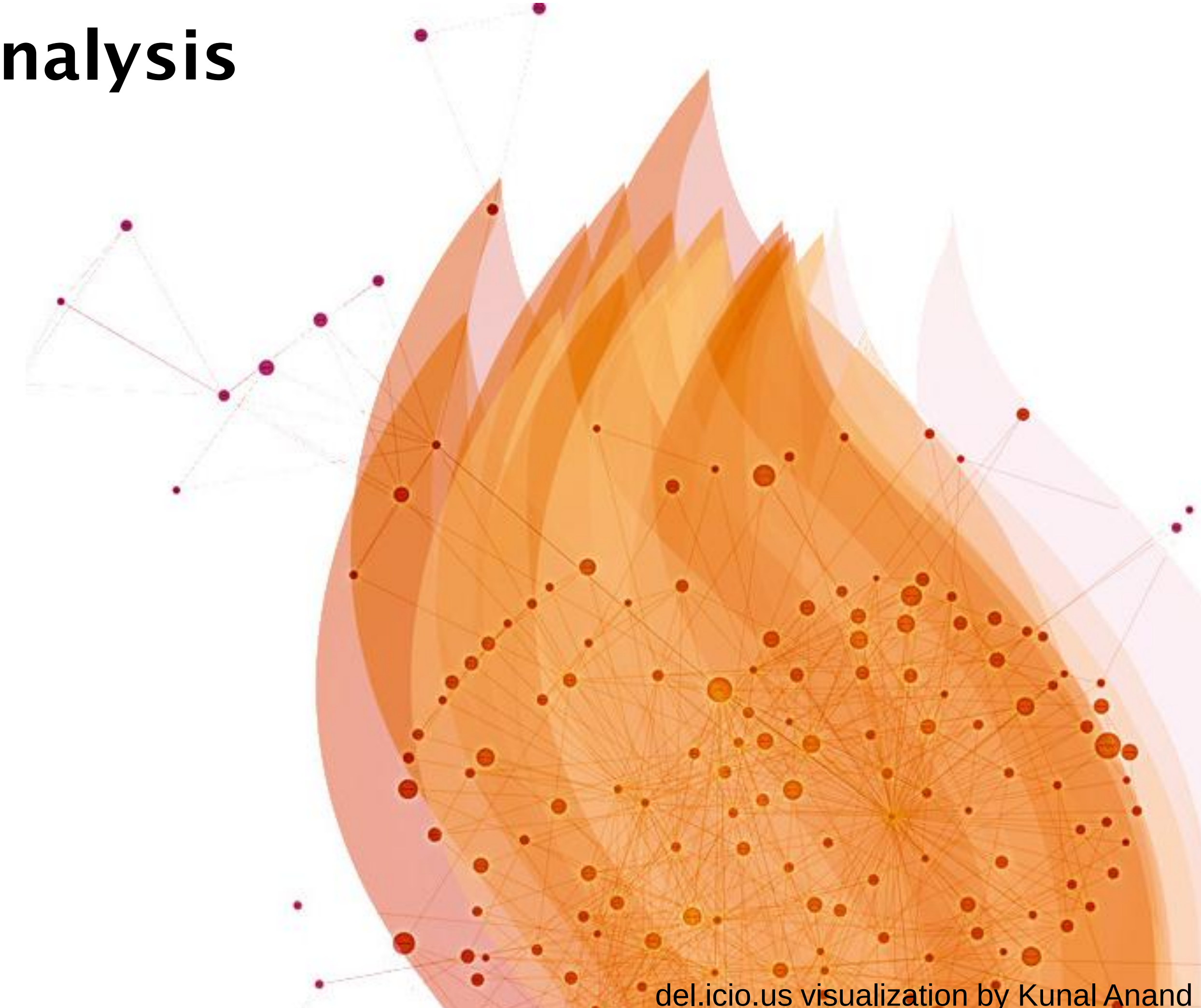
- Data Mining Problems
  - Pattern Mining
  - Clustering
  - Classification / Prediction
  - Compression
  - Modelling

Researchers

- Xifeng Yan – UC Santa Barbara
- Jiawei Han - U Illinois Urbana Champaign
- George Karypis – U Minnesota
- Charu Aggarwal – IBM T.J. Watson Research Center
- Jennifer Neville – Purdue University

Social Network Analysis

with graphs!



del.icio.us visualization by Kunal Anand

WASHINGTON STATE UNIVERSITY

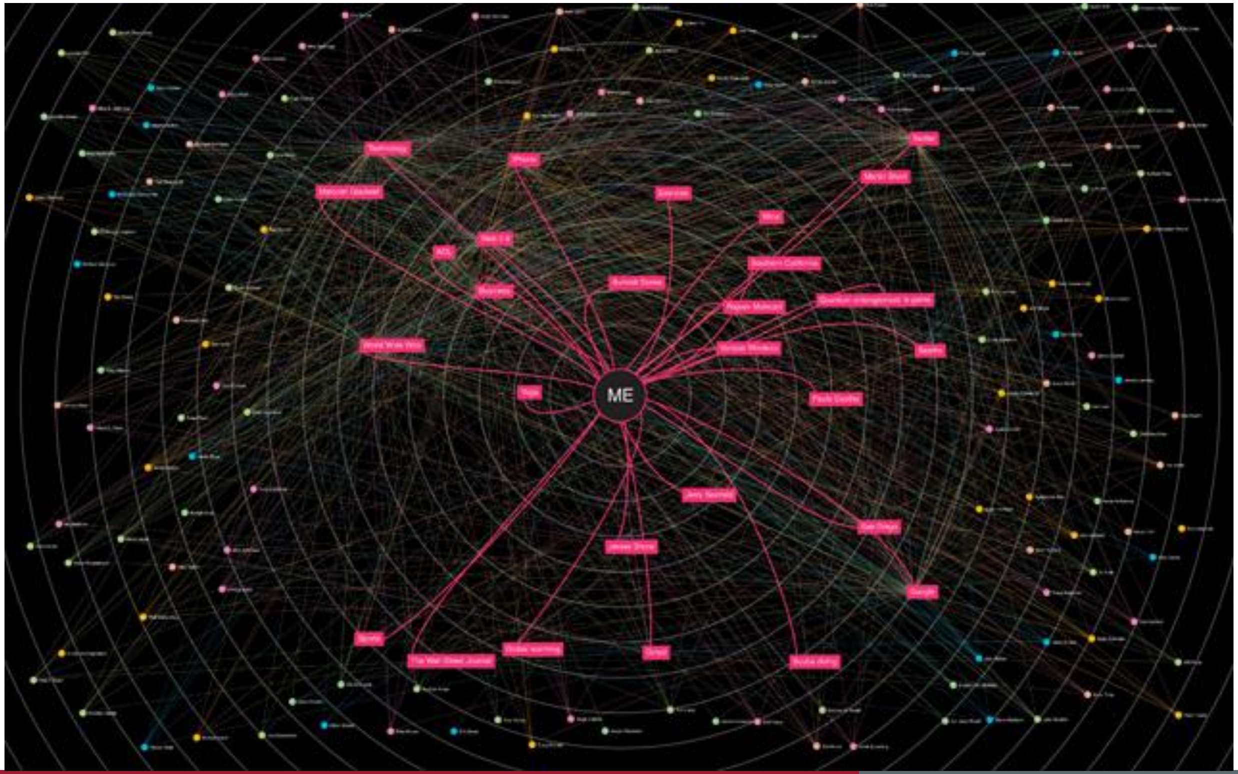
World Class. Face to Face.

Social Network Analysis

- Interested in populations not individuals
- Structure within society

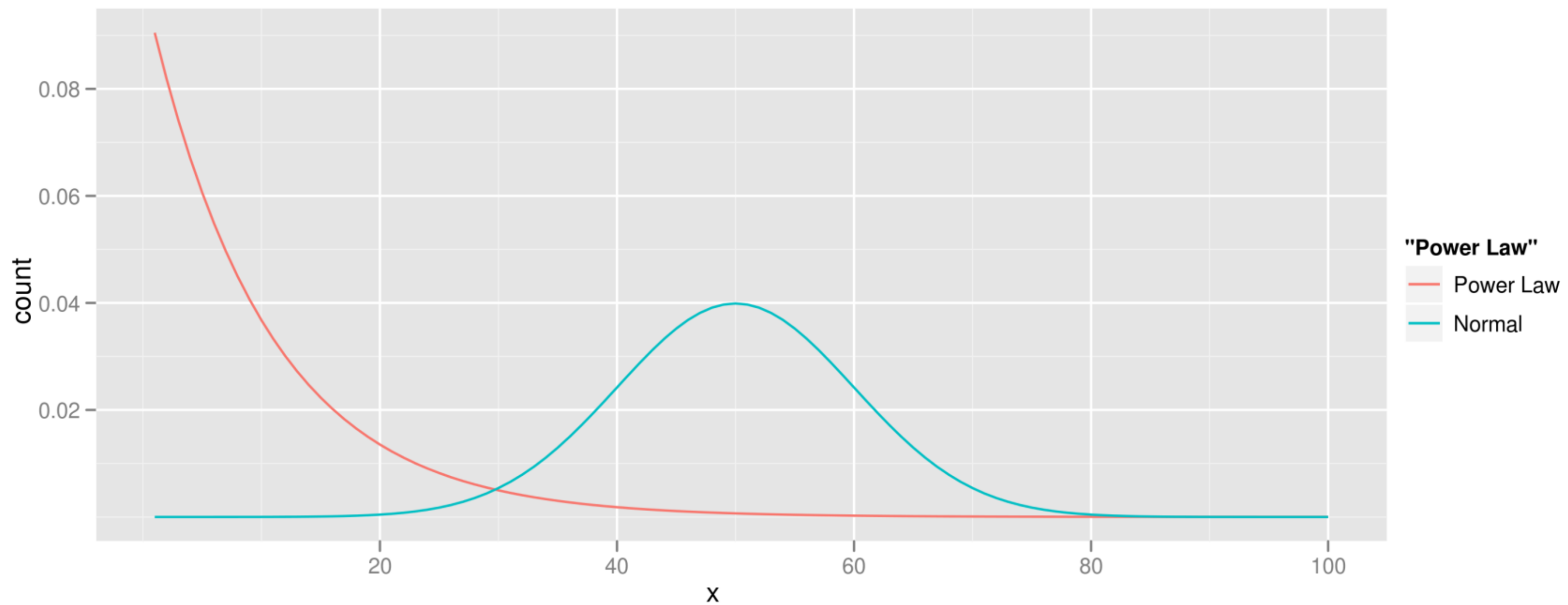
Social Network Analysis

- Network (Graph)
  - Nodes: Things (people, places, etc.)
  - Edges: Relationships (friends, visited, etc.)



Quantitative Analysis

- Analysis
  - Quantitative not qualitative
  - Usually Scale-Free Networks



Further Resources

- Social network analysis: methods and applications
  - Stanley Wasserman & Katherine Faust
- Christos Faloutsos, Watch the video
- Jure Leskovec @ Stanford
- The Mining Graph Data book

Towards Proximity Pattern Mining in Large Graphs

Arijit Khan, Xifeng Yan, Kun-Lung Wu

WASHINGTON STATE UNIVERSITY
World Class. Face to Face.

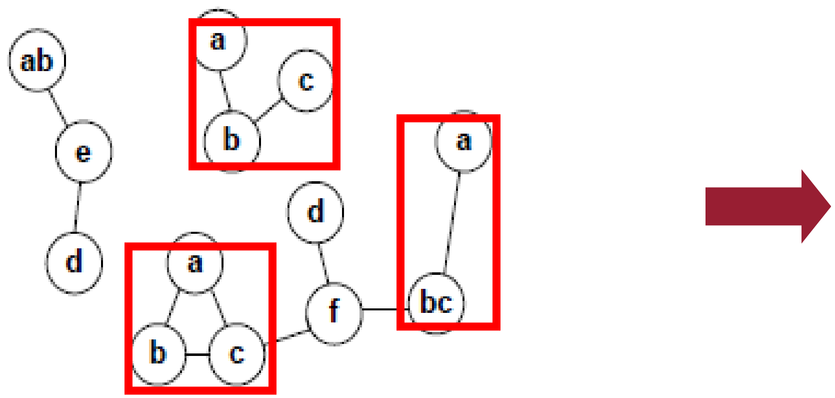
Overview

- Detects novel type of pattern in large graphs called proximity patterns
- Develop two models to model the growth of proximity patterns
- Use a probabilistic frequent pattern miner to extract proximity patterns
- Test their approach on Last.fm, Intrusion Alert Network, DBLP Collaboration Graph. Evaluate the interestingness of patterns generated using a statistical measure.

What is a proximity pattern?

- "A proximity pattern is a subset of labels that repeatedly appear in multiple tightly connected subgraphs" – (from [1]).

- Characteristics of proximity patterns:
  - Labels in pattern are tightly connected
  - They are frequent
  - They may not be connected in the same way always
- Useful in modelling information flow in a dynamic graph



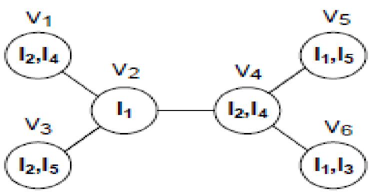
Proximity Pattern: {*a*,*b*,*c*}

Definitions

- The **support** sup(I) of an itemset P⊆ L (the set of labels) is the number of transactions in the data set that I is present in.
- A **frequent itemset** is an itemset which has a support greater than an user-defined threshold.
- The **downward closure property** of an itemset states that all subsets of a frequent itemset are frequent. Consequently all supersets of an infrequent itemset are infrequent.

Definitions

- Given a graph $G = (V,E)$ and a subset of vertices π, $\pi \in V(G)$, let $L(\pi)$ be the set of labels in π, i.e., $L(\pi) = \cup_{u \in \pi} L(u)$. Given a label subset I, π is called an embedding of I if $I \subseteq L(\pi)$. The vertices in π need not be connected. (from [1])



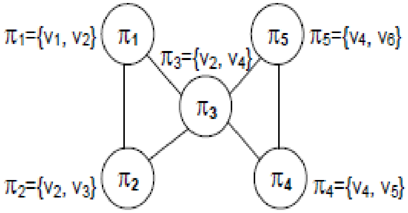
$v_1,v_2,v_3\}$ is an embedding of $\{l_1,l_2,l_5\}$

- $\Phi_1(l_1,l_2,l_5) = (v_2,v_1,v_3)$ (*minimum*)
- $\Phi_2(l_1,l_2,l_5) = (v_2,v_3,v_3)$ (non-minimum)

- A mapping φ between I and the vertices in π is a function $\varphi: I \rightarrow \pi$ s.t., $\exists\, l$, $\varphi(l) \in \pi$ and $l \in L(\varphi(l))$. A mapping is minimum if it is surjective, i.e., $\forall v \in \pi$, $\exists l$ s.t. $\varphi(l) = v$. (from [1])
- Given an itemset I and a mapping φ, we need a function $f(\varphi)$ to measure its association strength. (from [1])

Neighbour Association Model

- Procedure:
  - Find all embeddings $\pi_1, \pi_2, \dots, \pi_m$ of an itemset I.
  - For each embedding π measure the strength f(π
  - Support of itemset I = $\Sigma_{i=1}^{m}$ f(π).



- Problems are:
  - Support count based on overlapping embeddings violate the downward closure property
  - Any subset of vertices could be an embedding.
- To solve the first problem, a overlap graph is created. The support is the sum of weights of the nodes in the maximum weight independent set of this graph.
- Disadvantages:
  - NP-Hard w.r.t. to the number of embeddings for a given pattern.
  - Not feasible to generate all the embeddings for a pattern as there could be a large number of embeddings
  - Support count is NP-Complete.

Information Propagation Model

- $G_0 \rightarrow G_1 \rightarrow G_i \rightarrow . . . \rightarrow G_n$

- Let <u>L(u)</u> be the present state of *u*, denoted by the labels present in *u*, and *l* be a distinct label propagated by one of its neighbors and $l \notin L(u)$. Hence, the probability of observing *L(u)* and *l* is written as:

  $$P(L \cup \{l\}) = P(L|l)P(l),$$

  where *P(l)* is the probability of *l* in *u*'s neighbors and *P(L|l)* is the probability that *l* is successfully propagated to *u*.

- For multiple labels, $l_1, l_2, . . . , l_m$,

  $$P(L \cup \{l_1, l_2, . . . , l_m\}) = P(L|l_1) * . . . * P(L|l_m) * P(l_1) * . . . P(l_m).$$

- Propagation model indicates the characteristic of social networks where the influence of a given node on any other decreases as the distance from the given node to the other node increases.

Nearest Probabilistic Association

- $A_u(l) = P(L(u)|l) = e^{-\alpha.d}$,

  where 'l' is a label present in a vertex 'v' closest to 'u',

  'd' is the distance from 'v' to 'u' (=1 for unweighted graph)

  'α' is the decay constant ($\alpha > 0$)

- Probabilistic Support of I:

$$\frac{1}{|V|} \sum_{u \in V} A_u(l_1) \cdots A_u(l_m).$$

- If I = {l1, l2, . . . ,m} and J = {l1, l2, . . . , lm, lm+1 . . . , n}, then since,

$$\prod_{i=1}^{m} A_u(l_i) \geq \prod_{i=1}^{m} A_u(l_i) \prod_{i=m+1}^{n} A_u(l_i).$$

sup(I) $\geq$ sup(J)

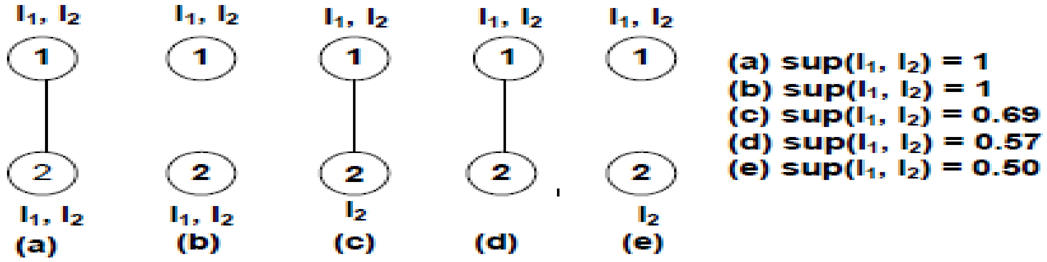
Nearest Probabilistic Association

Algorithm 1 Generate Intermediate Dataset $\widetilde{G}$

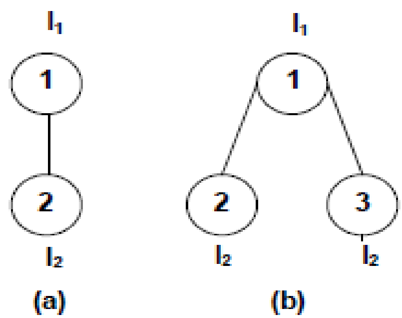
Input: Graph G, cut-off parameter ϵ.
Output: Intermediate Dataset $\widetilde{G}$.

1: $i = 0$ // iteration
2: **for all** vertex u of G **do**
3:    Let $L_0(u)$ be the label set of u
4:    $\forall l \in L_0(u)$, $A_u(l)$=1; otherwise $A_u(l)$=0
5: **end for**
6: **for all** vertex u of G **do**
7:    **for all** label l in $L_i(v) \setminus L_i(u)$, v is u's neighbor **do**
8:       update $A_u(l)$ using Definition 4 (choose the maximum one)
9:       If less than ϵ, do not propagate l to u
10:    **end for**
11:    $L_{i+1}(u) = \{L_i(u) \cup \{l\} | A_u(l) > 0\}$
12: **end for**
13: **if** $L_{i+1} = L_i$ for all vertices in G **then**
14:    Output A_u for all $u \in V(G)$
15: **else**
16:    $i = i + 1$, goto step 2
17: **end if**

Nearest Probabilistic Association



Consistency between NPA and Frequent Itemset



Problem with NPA

NPA – Complexity, Advantages and Disadvantages

- **O(|V| . d^t . s),**
  - V is the total no. of vertices
  - ‘d’ is the average degree
  - ‘t’ is the number of iterations
  - ‘s’ is the number of labels in every vertex

- **Advantage:** It is fast to calculate

- **Disadvantage:** Considers only the nearest neighbour for propagation.

Normalized Probabilistic Association

- Normalized Probabilistic Association of label ‘l’ at vertex ‘u’

$$NA_u(l) = P(L(u)|l) = \frac{m}{n+1}e^{-\alpha}$$

- NMPA can break ties when
  - two vertices have the same number of neighbors
  - two vertices have different number of neighbours but the same number of neighbours having label ‘l’.

- The update rule of Algorithm 1 is changed:

$$NA_u(l) = \frac{1}{n+1}\sum_{v \in N(u)} e^{-\alpha} * NA_v(l)$$

- Complexity same as NPA, however propagation decays faster due to normalization.

Probabilistic Itemset Mining

- Uses a probablistic version of FP-growth algorithm[3][4] to mine frequent itemsets from uncertain data.

- Two variations of probabilistic FP-growth algorithm used
  - Exact Mining: Using an FP-tree where nodes are augmented with the probability values present in the proximity patterns
  - Approximate Mining: FP-tree augmented not with all probability values, but only sum of probability values and number of occurrences.

- C. Aggarwal et al. in [2] proposes various techniques to mine frequent patterns from uncertain data. Probabilistic version of H-Mine found to be the best.

Experiments

- Datasets used
  - **LAST.FM**: Nodes labelled with artists, edges represent communication between users
  - **Intrusion Alert Network**: Node is a computer labelled with attacks, edge represents possible attack
  - **DBLP Collaboration Graph**: Nodes represent authors labelled with keywords, edges represent collaboration.

- Labels are randomized and G-test score used to compare between the support of patterns in actual graph and randomized graph.

Experiments

- Effectiveness test:

#	Proximity Patterns	Score
1	Tiësto, Armin van Buuren , ATB	0.62
2	Katy Perry, Lady Gaga, Britney Spears	0.58
3	Ferry Corsten, Tiësto, Paul van Dyk	0.55
4	Neaera, Caliban, Cannibal Corpse	0.52
5	Lacuna Coil, Nightwish, Within Temptation	0.47

Table 3: Top-5 Proximity Patterns (Last.fm)

#	Proximity Patterns	Score
1	Tiësto, Armin van Buuren , ATB	0.62
2	Katy Perry, Lady Gaga, Britney Spears	0.58
3	Ferry Corsten, Tiësto, Paul van Dyk	0.55
4	Neaera, Caliban, Cannibal Corpse	0.52
5	Lacuna Coil, Nightwish, Within Temptation	0.47

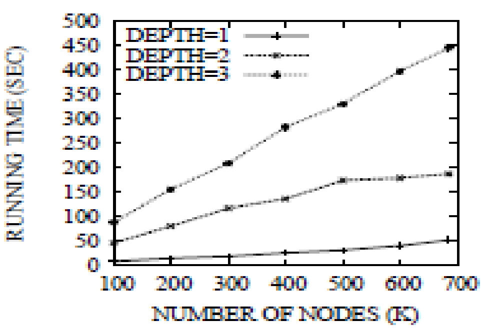
Table 4: Proximity Patterns minus Frequent Item-sets (Last.fm)

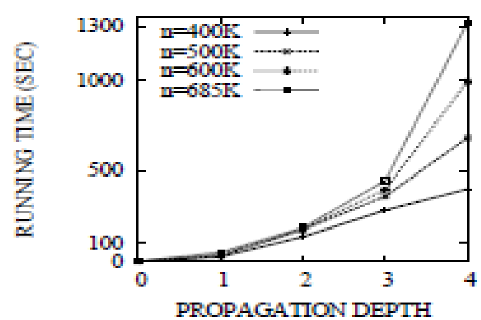
Experiments

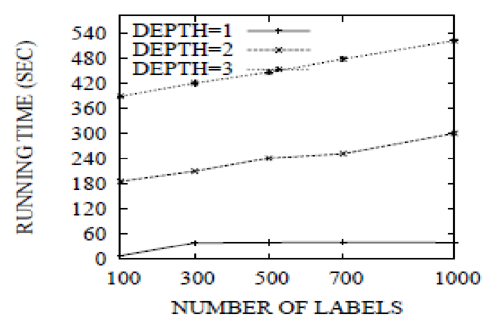
- Efficiency and Scalability

Steps	Last.fm	Intrusion	DBLP
NmPA	2.0	5.0	187.0
FP-tree Formation	1.0	10.0	89.0
Top-k Pattern Mining	4.0	2.0	254.0

Table 8: Runtime (sec)







Experiments

- Exact vs. Approximate Mining

#	Proximity Patterns	Score
1	Katy Perry, Lady Gaga, Britney Spears	0.58
2	Ferry Corsten, Tiësto, Paul van Dyk	0.55
3	Tiësto, Armin van Buuren, ATB	0.55
4	Neaera, Caliban, Cannibal Corpse	0.51
5	Lacuna Coil, Nightwish, Within Temptation	0.46

Steps	aFP(approximate)	pFP(exact)
FP-tree Formation	1.0	3.0
Top-k Pattern Mining	4.0	21.0

- Gives more proximity patterns than Frequent Subgraph Mining, and is faster as it avoids isomorphism checking

Strengths

- Discovers new kind of patterns.
- Considers structure without incurring isomorphism checking.
- Applicable to weighted and un-weighted graphs.
- Can be extended to directed graphs

Weaknesses

- Run-time exponential with depth of propagation.
- Proximity pattern itself does not retain structural information.
- Large number of patterns might be generated.
- Different paths in the graph might have different decay rates.

Discussions

- What real life graphs/problems might this approach be really good for?
- Is there a better way to determine proximity?
- What modifications would be needed for the technique to handle directed subgraphs?
- How does the rate of decay affect the patterns found?
- How well will the approach fare when graphs have billions of nodes and edges or more?

References

1. A. Khan, X. Yan, and K.-L.Wu. Towards proximity pattern mining in large graphs. In *SIGMOD*, 2010.
2. C.C. Aggarwal, Y. Li, J. Wang, J. Wang. Frequent Pattern Mining with Uncertain Data. KDD 2009.
3. Presentation on Frequent Pattern Growth (FP-Growth) Algorithm: An Introduction by Florian Verhein
4. J. Han, J. Pei, Y. Yin. Mining frequent patterns without candidate generation. SIGMOD, 2000.

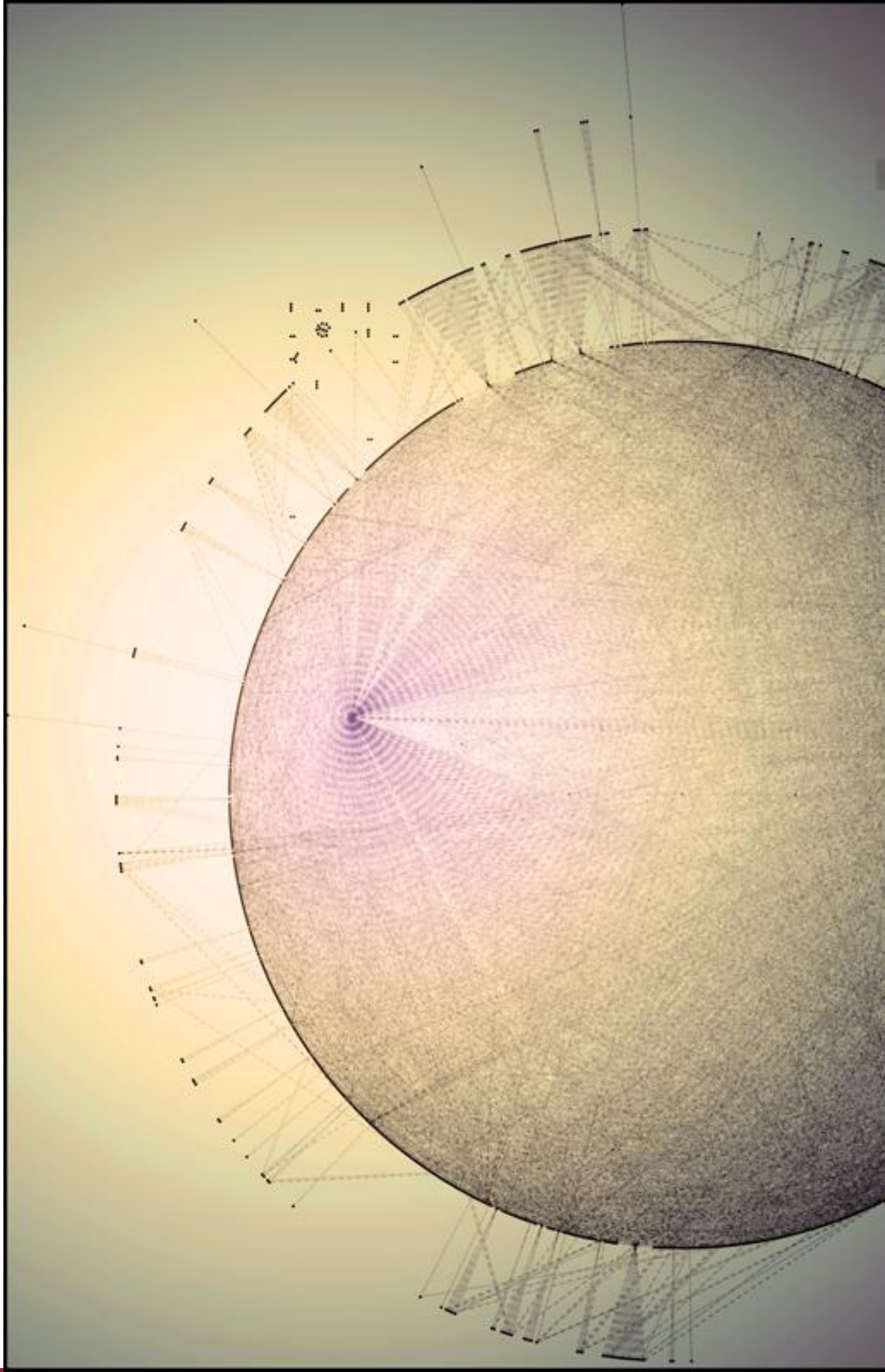
What are your research problems?

Avista Data (HW 4)

Radius Plots for Mining Tera-byte Scale Graphs

Algorithms, Patterns, and Observations

WASHINGTON STATE UNIVERSITY
World Class. Face to Face.



Authors

- U. Kang (CMU)
- C. Tsourakakis (CMU)
- A. Appel (USP São Carlos)
- C. Faloutsos (CMU)
- J. Leskovec (Stanford)

Basic Definitions

- Distance (u,v)
  - d(u,v) => shortest path between u and v
- Eccentricity (v)
  - e(i) => max(d(i, v)), for all v in G
- Radius (G)
  - Minimum e(v) for all v in G
- Diameter (G)
  - Maximum e(v) for all v in G

Big Data

- YahooWeb
  - 6B edges, 125GB on disk
    - TBs of memory required to run algos
    - Large enough nobody else has published on it
- Billion Triple Challenge
  - ~2B edges
- Social networking data
  - Spinn3r
- Google's 1T word language model

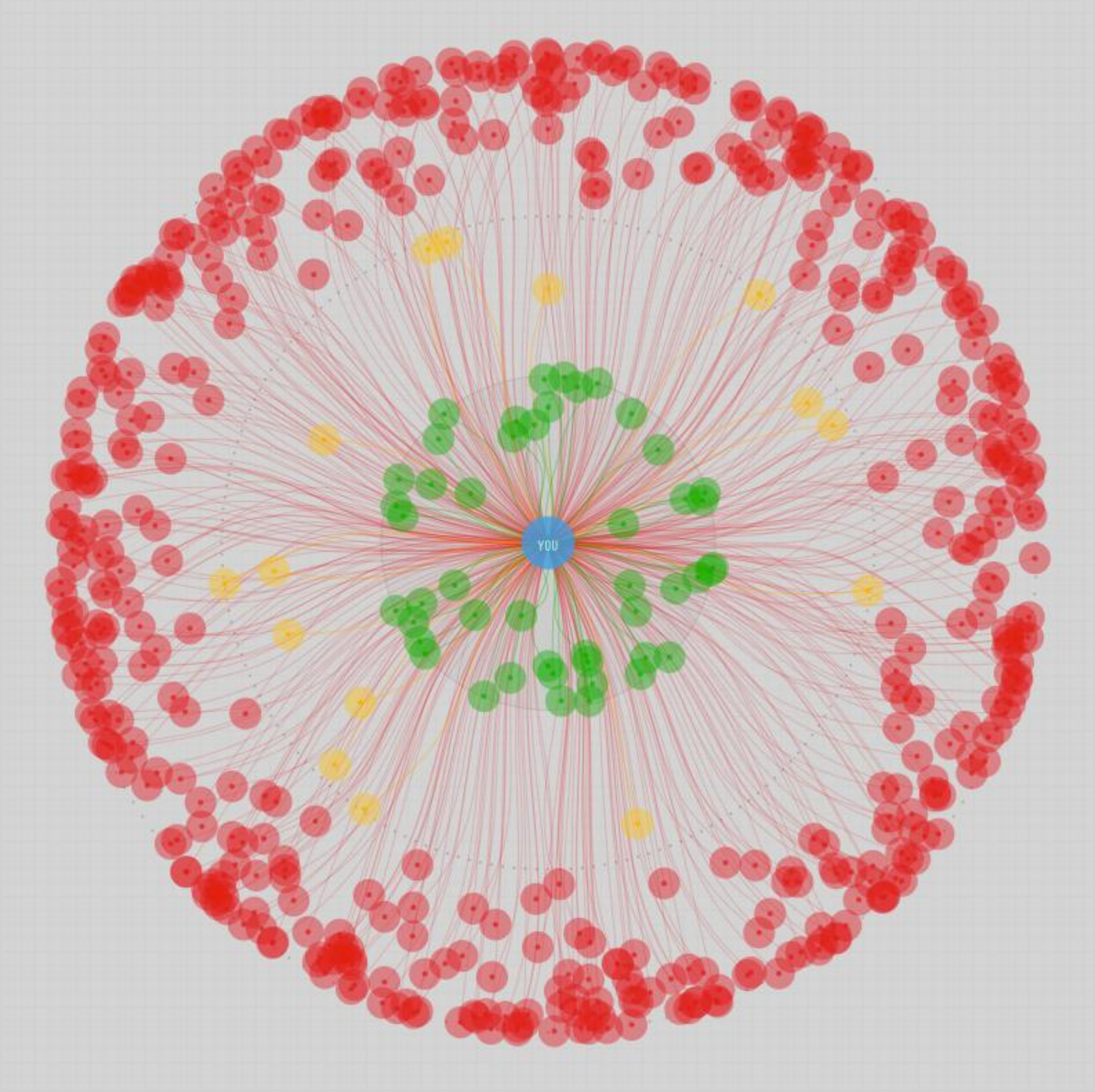
Flajolet-Martin

- Probabilistic counting paper from 1985
- Given
  - M => K x 64 bit matrix
  - h(a) => mostly uniform 64-bit hashing function
  - ρ(a) => index of lowest cleared bit in a
- Pseudo-Code
  - For x in data:
    - M[h(x) % K, ρ(h(x)/K)] = 1
  - S=0
  - For row in M:
    - from i=0 until row[i++] == 0:
      - S += i
  - Return K/0.77351 * $2^{S/K}$

Flajolet-Martin for Radius Estimation

- Authors set K to 32
  - Original paper shows this gives stdev 13% error
- 64-bit bitstrings used
- Bit-Shuffle Encoding
  - Only pass around the number of starting 1s and ending 0s
  - Optimizes network load

http://d3.do/en/?p=306



Map/Reduce in a nutshell

- From the paper:

MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key.

- Implementation is out of our scope

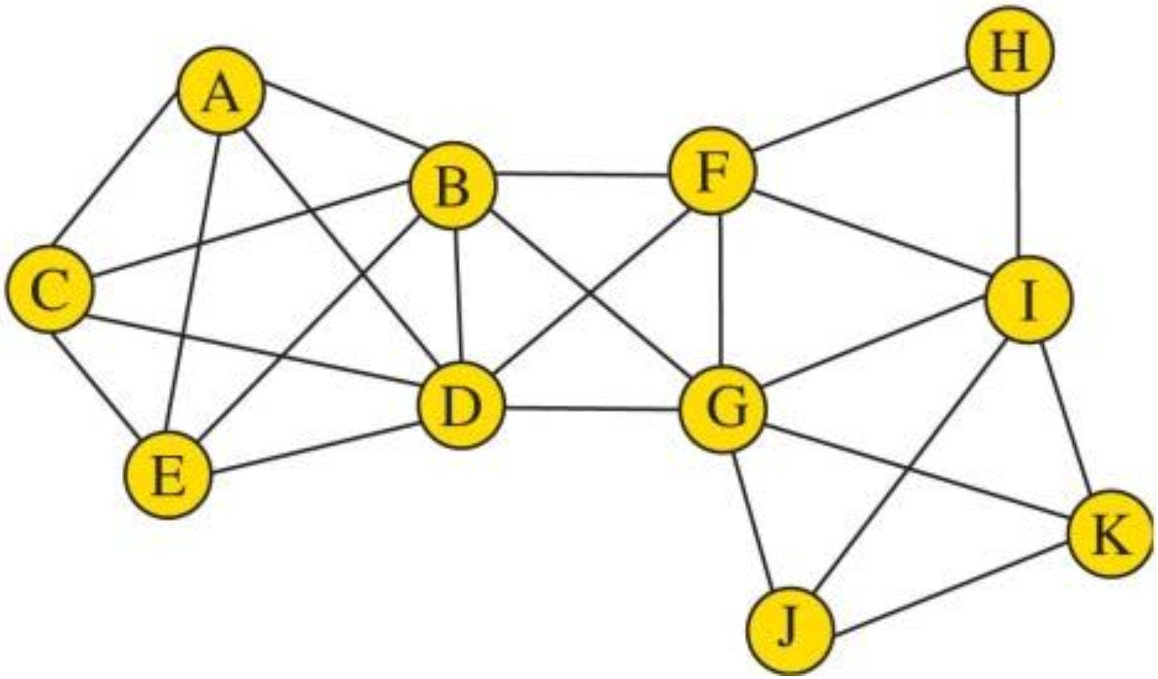
MapReduce/Hadoop Radius

- Map: Output (node_id, list of neighbor bitstrings)
- Reduce: Add bitstrings to my own
- Optimizations
  - Bundle edges into blocks
  - Change bitstring to count blocks
  - Perform sequence length encoding on bitstrings

Recall Basic Definitions

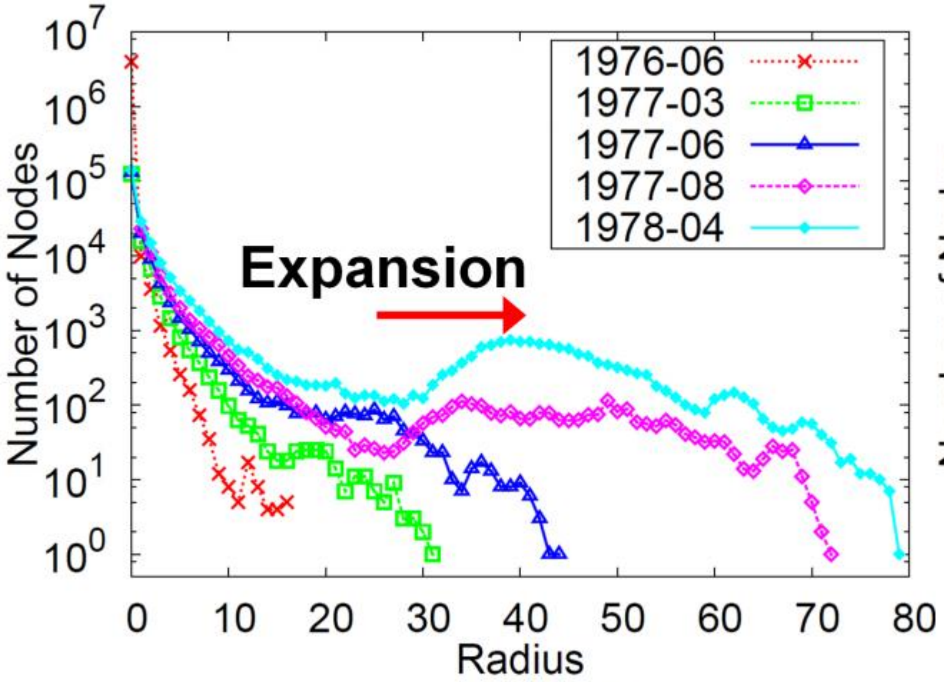
- Distance (u,v)
  - d(u,v) => shortest path between u and v
- Eccentricity (v)
  - e(i) => max(d(i, v)), for all v in G
- Radius (G)
  - Minimum e(v) for all v in G
- Diameter (G)
  - Maximum e(v) for all v in G

Visualizing HADI

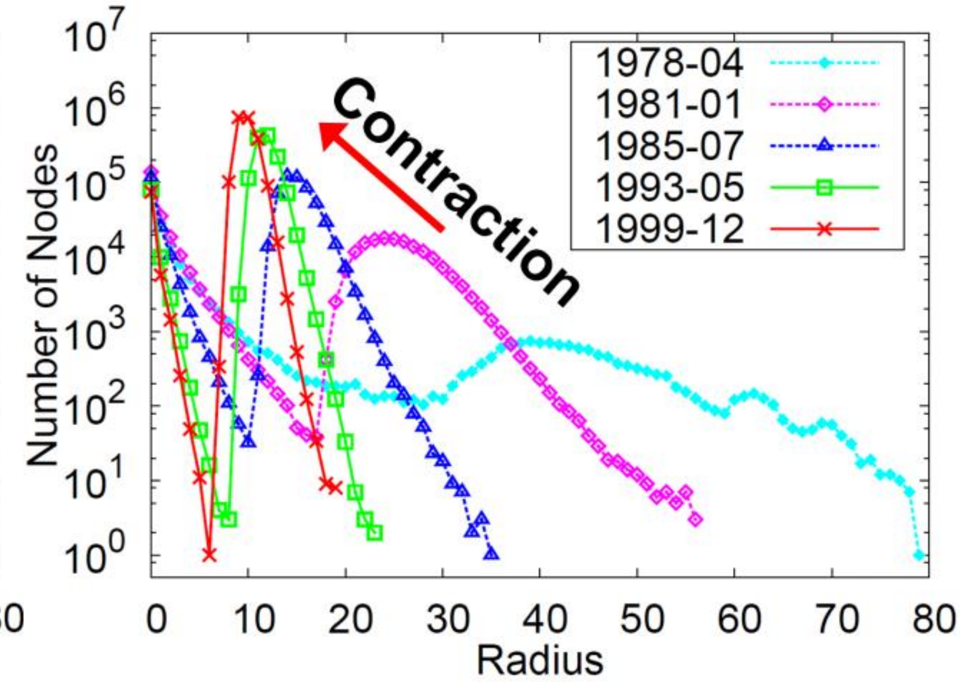


Network Dynamics

- Why diameter shrinks over time



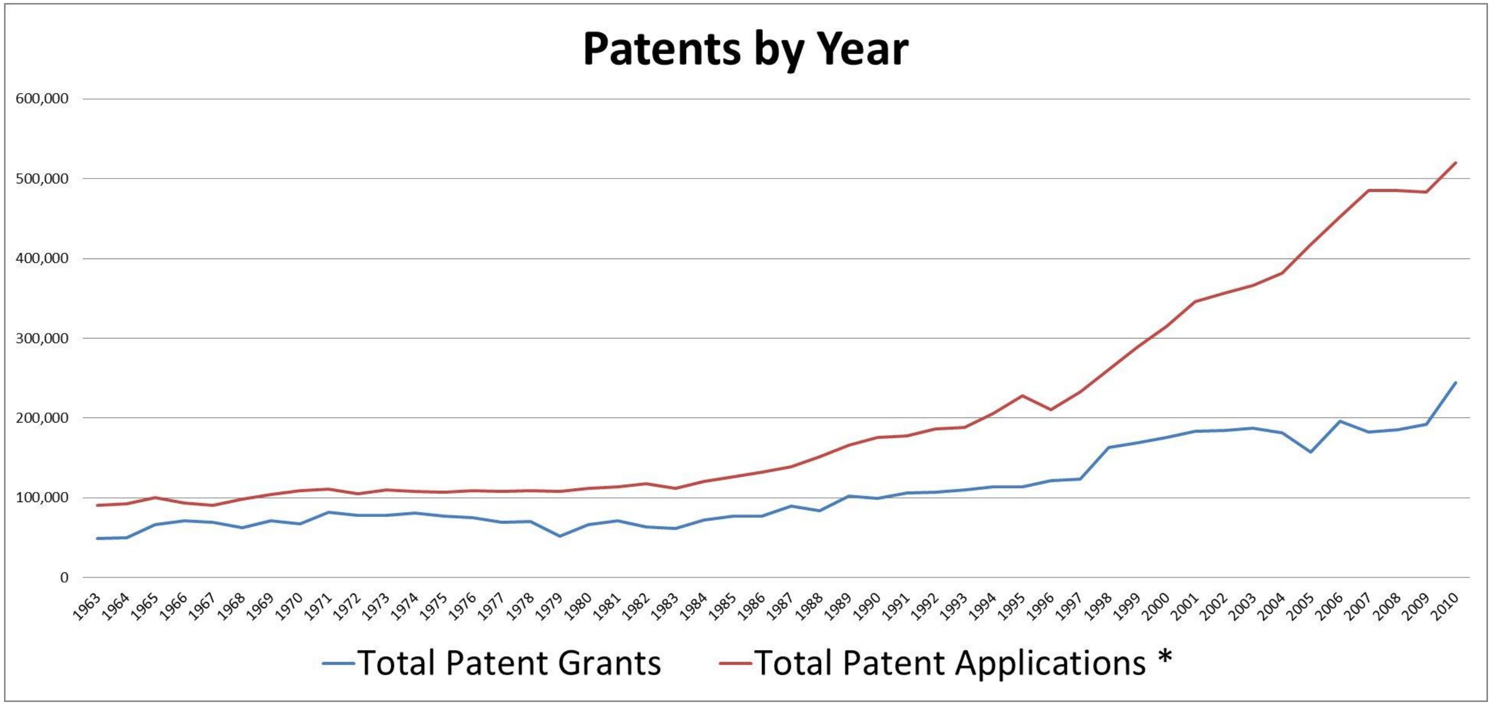
(a) Patent-Expansion



(b) Patent-Contraction

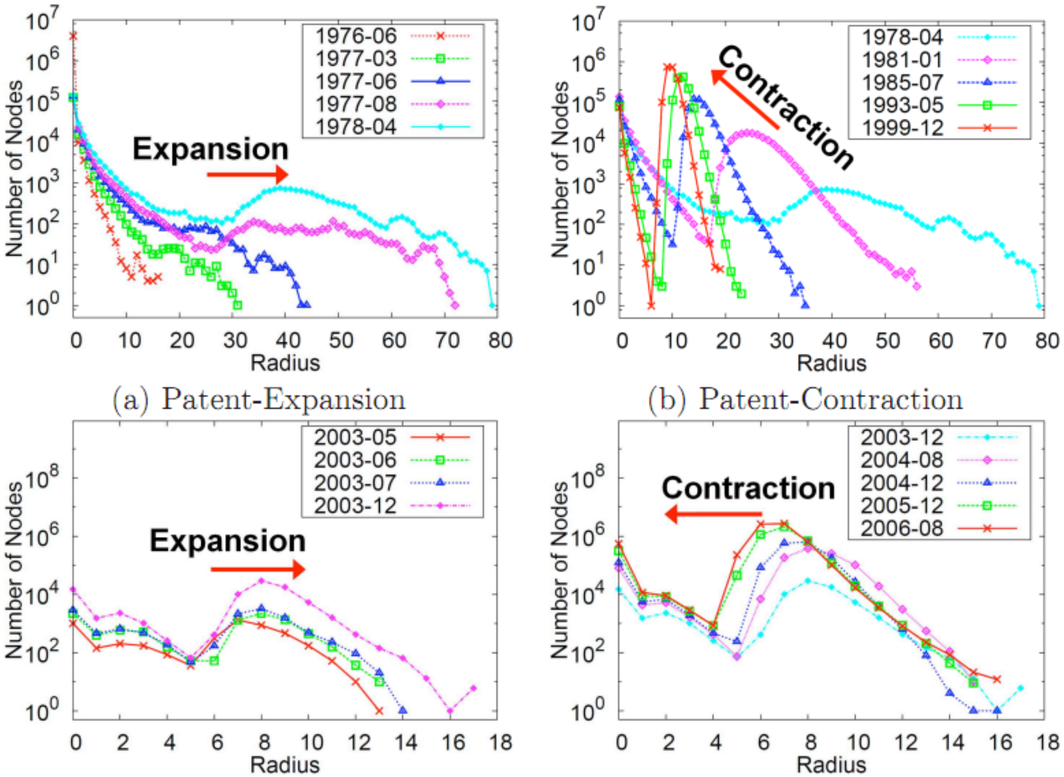
What is happening with Patents

- Edges are undirected
- Patents reference previous work
- Always more previous work to reference



Network Dynamics

- Patent vs. LinkedIn
- See HADI: Mining Radii of Large Graphs

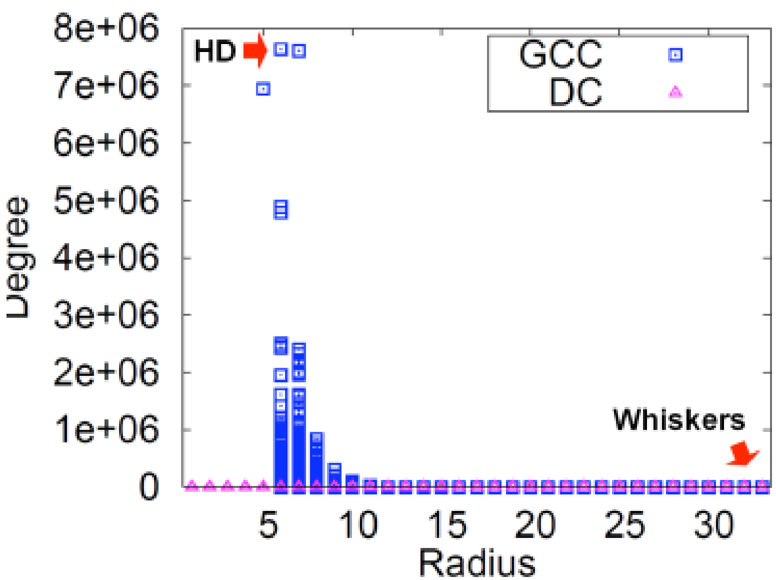


(a) Patent-Expansion (b) Patent-Contraction

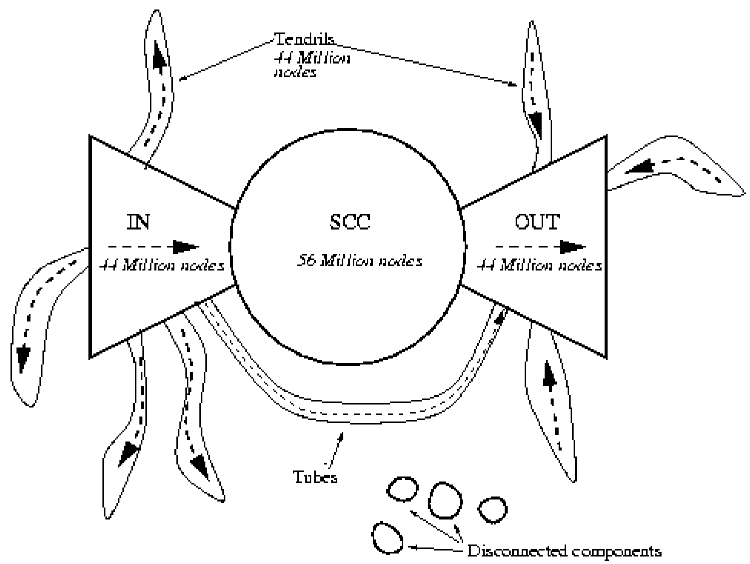
(c) LinkedIn-Expansion (d) LinkedIn-Contraction

Other observations

- "Surprising" diameter of 7~8
  - Could be a PageRank effect of SEO



(b) YahooWeb



Graph structure in the web
Broder et al., 2000

Discussion

- Can you represent your research problem as a graph?
  - Should you?
- Would it be useful to know how fast information can propagate through your graph?
- Can you breakdown your approach into a map/reduce paradigm?

Graphs are pretty, look at them more!